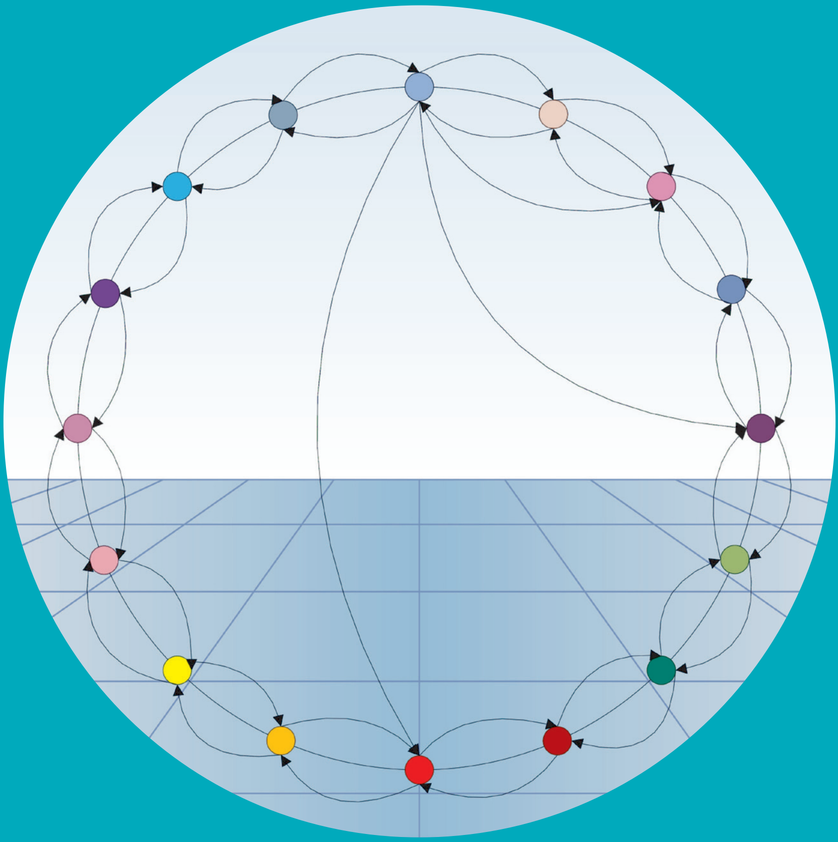
Communications and Networking

Framework Architecture for Decentralized Communications

Jouni Mäenpää



Aalto University

DOCTORAL DISSERTATIONS

Aalto University publication series
DOCTORAL DISSERTATIONS 62/2013

Framework Architecture for Decentralized Communications

Jouni Mäenpää

A doctoral dissertation completed for the degree of Doctor of Science (Technology) to be defended, with the permission of the Aalto University School of Electrical Engineering, at a public examination held at the lecture hall S1 of the school on 30 May 2013 at 12 o'clock, noon.

Aalto University
Aalto School of Electrical Engineering
Communications and Networking

Aalto University publication series
DOCTORAL DISSERTATIONS 62/2013

© Jouni Mäenpää

ISBN 978-952-60-5120-8 (printed)
ISBN 978-952-60-5121-5 (pdf)
ISSN-L 1799-4934
ISSN 1799-4934 (printed)
ISSN 1799-4942 (pdf)
http://urn.fi/URN:ISBN:978-952-60-5121-5

Unigrafia Oy
Helsinki 2013

Finland

NORDIC ECOLABEL
441 697
Printed matter

Aalto University

Abstract

Aalto University, P.O. Box 11000, FI-00076 Aalto www.aalto.fi

Author
Jouni Mäenpää

Name of the doctoral dissertation
Framework Architecture for Decentralized Communications

Publisher Aalto School of Electrical Engineering

Unit Communications and Networking

Series Aalto University publication series DOCTORAL DISSERTATIONS 62/2013

Field of research Networking Technology

Manuscript submitted 17 December 2012 **Date of the defence** 30 May 2013

Permission to publish granted (date) 28 February 2013 **Language** English

☐ **Monograph** ☒ **Article dissertation (summary + original articles)**

Abstract

Peer-to-Peer (P2P) systems represent a paradigm shift from the traditional client/server architecture. Over the past decade, P2P technologies have proven themselves as a viable option for providing services in the Internet. This success has resulted in initiatives to develop standards-based P2P protocols and services. One of the major initiatives in this area is Peer-to-Peer Session Initiation Protocol (P2PSIP), a suite of communication protocols that enable the Session Initiation Protocol (SIP) to decentralize its functions. P2PSIP is being standardized in the Internet Engineering Task Force (IETF). This dissertation presents a framework architecture for decentralized communications that is built around P2PSIP and the set of technologies it uses, including the REsource LOcation And Discovery (RELOAD) P2P signaling protocol, Chord Distributed Hash Table (DHT) algorithm, and the Interactive Connectivity Establishment (ICE) Network Address Translator (NAT) traversal solution.

The framework presented in this dissertation is a set of reusable and modular software components that can be used in a flexible manner either individually or in different combinations to support the needs of a broad set of applications and use cases. Due to its flexibility and modularity, the framework is not an integrated architecture, tightly coupled set of components, or a purpose-built software platform whose components cannot function individually or are not interchangeable. The framework and all of its components were implemented as a part of the work on this dissertation.

In the dissertation, the performance of the implementation of the framework and its components is evaluated using real-world prototypes and simulators. The focus is on evaluating the performance of DHT maintenance routines, ICE-based NAT traversal, the operations the framework provides to applications, and the performance of the implementation of the framework in mobile environments. Based on the performance analysis, missing features and performance bottlenecks are identified. The performance bottlenecks are addressed and the missing features are added by designing new components to complete the framework. These components include self-tuning, service discovery, M2M communication, and session setup delay optimization components.

The contributions of this dissertation can be divided into three categories. First, the delays associated with using the services and operations provided by the framework are analyzed and optimized. Second, the overlay network platform that the framework provides is evaluated and extended. Finally, the framework is applied to new use cases. The overall result of the work is a framework architecture for decentralized communications that is scalable, adaptive, generic, modular, based on emerging standards, and has high performance.

Keywords P2PSIP, SIP, RELOAD, ICE, P2P, M2M

ISBN (printed) 978-952-60-5120-8 **ISBN (pdf)** 978-952-60-5121-5

ISSN-L 1799-4934 **ISSN (printed)** 1799-4934 **ISSN (pdf)** 1799-4942

Location of publisher Espoo **Location of printing** Helsinki **Year** 2013

Pages 241 **urn** http://urn.fi/URN:ISBN:978-952-60-5121-5

Aalto-yliopisto

Tiivistelmä

Aalto-yliopisto, PL 11000, 00076 Aalto www.aalto.fi

Tekijä
Jouni Mäenpää

Väitöskirjan nimi
Viestinnän hajautuksen mahdollistava viitekehysarkkitehtuuri

Julkaisija Sähkötekniikan korkeakoulu

Yksikkö Tietoliikenne- ja tietoverkkotekniikan laitos

Sarja Aalto University publication series DOCTORAL DISSERTATIONS 62/2013

Tutkimusala Tietoverkkotekniikka

Käsikirjoituksen pvm 17.12.2012 **Väitöspäivä** 30.05.2013

Julkaisuluvan myöntämispäivä 28.02.2013 **Kieli** Englanti

☐ **Monografia** ☒ **Yhdistelmäväitöskirja (yhteenveto-osa + erillisartikkelit)**

Tiivistelmä

Vertaisverkkoarkkitehtuurit poikkeavat perinteisistä, keskitettyihin palvelimiin pohjautuvista arkkitehtuureista. Viimeisimmän vuosikymmenen aikana vertaisverkot ovat osoittautuneet toimivaksi tavaksi toteuttaa palveluja. Tämä menestys on herättänyt kiinnostuksen standardoida vertaisverkkoyhteyskäytäntöjä ja -palveluita. Tärkein näistä aloitteista on P2PSIP (Peer-to-Peer Session Initiation Protocol). P2PSIP on joukko yhteyskäytäntöjä, joita standardoidaan IETF (Internet Engineering Task Force) -organisaatiossa. Tämä työ esittelee P2PSIP:n käyttämien teknologioiden ympärille rakennetun viitekehysarkkitehtuurin, joka mahdollistaa viestinnän hajautuksen. Teknologiat, joita työssä käytetään ovat RELOAD (REsource LOcation And Discovery), hajautetut tiivistetaulualgoritmit (DHT) ja ICE (Interactive Connectivity Establishment) -osoitteenmuuntajien läpäisymenetelmä.

Työssä esitetty viitekehys koostuu joukosta itsenäisiä komponentteja, joita voidaan käyttää joko yksitellen tai ryhmissä tukemaan erilaisten käyttötapausten tarpeita. Viitekehys on suunniteltu joustavaksi; se ei ole jäykkä yhtenäinen arkkitehtuuri eikä joukko toisistaan riippuvia komponentteja. Se ei myöskään ole yhtä tarkoitusta varten rakennettu ohjelmistoalusta jonka komponentit eivät voi toimia itsenäisesti ja jotka eivät ole korvattavissa. Viitekehys ja sen komponentit toteutettiin osana tätä työtä.

Tämä työ arvioi yllä mainitun viitekehyksen toteutuksen suorituskykyä käyttäen työn osana kehitettyjä prototyyppejä ja simulaattoria. Työ keskittyy arvioimaan DHT-algoritmien ylläpitorutiineja, ICE-menetelmää, viitekehyksen sovelluksille tarjoamien toimintojen suorituskykyä ja viitekehyksen toteutuksen suorituskykyä langattomissa verkoissa. Suorituskykyanalyysin pohjalta työ nostaa esille tekijöitä, jotka voivat muodostua pullonkauloiksi suorituskyvylle ja tunnistaa uusia hyödyllisiä ominaisuuksia. Työssä edellä mainitut pullonkaulat ratkaistaan ja uudet ominaisuudet lisätään suunnittelemalla uusia osia, jotka täydentävät viitekehyksen. Nämä uudet osat ovat hajautettujen tiivistetaulualgoritmien itseviritysmenetelmä, palveluiden hakumenetelmä, laitteiden välinen (M2M) hajautettu viestintämenetelmä ja istunnonaloitusviiveiden optimointimenetelmä.

Työn tulokset voidaan jakaa kolmeen osa-alueeseen. Näistä ensimmäinen on viitekehyksen tarjoamiin palveluihin ja toimintoihin liittyvien viiveiden arviointi ja optimointi. Toinen osa-alue on viitekehyksen käyttämän sovelluskerroksen verkkoalustan suorituskyvyn arviointi ja alustan laajentaminen. Kolmas osa-alue on uusien käyttötapausten kehittäminen. Työn lopputulos on viestinnän hajautuksen mahdollistava viitekehysarkkitehtuuri, joka on skaalautuva, muuttuviin olosuhteisiin sopeutuva, sovellusriippumaton, moduuleista koottu, alustaviin standardeihin pohjautuva ja omaa korkean suorituskyvyn.

Avainsanat P2PSIP, SIP, RELOAD, ICE, P2P, M2M

ISBN (painettu) 978-952-60-5120-8 **ISBN (pdf)** 978-952-60-5121-5

ISSN-L 1799-4934 **ISSN (painettu)** 1799-4934 **ISSN (pdf)** 1799-4942

Julkaisupaikka Espoo **Painopaikka** Helsinki **Vuosi** 2013

Sivumäärä 241 **urn** http://urn.fi/URN:ISBN:978-952-60-5121-5

Preface

This dissertation presents a framework architecture for decentralized communications. The framework is based on the author's work on Peer-to-Peer Session Initiation Protocol (P2PSIP) and the set of enabling technologies it uses, including Distributed Hash Table (DHT) algorithms, REsource LOcation And Discovery (RELOAD), and Interactive Connectivity Establishment (ICE). The work was carried out at Ericsson Finland between the years 2007 and 2013.

I would like to thank my managers at Ericsson for providing me the opportunity to carry out the research for this dissertation and for supporting me in different phases of the process, including especially Gonzalo Camarillo, Johan Torsner, Raimo Vuopionperä, Viktor Berggren, Harri Reiman, Hans Hermansson, and Daniel Enström. I am grateful to the co-authors of my publications, especially Gonzalo Camarillo for his invaluable comments and guidance, and Jaime Jiménez for all the idea-generating discussions. I would also like to thank my colleagues at Ericsson with whom I worked in the projects in which the research for this dissertation was carried out.

Professor Raimo Kantola supervised this dissertation and all of my studies at Aalto University. I would like to thank him for his invaluable advice and comments during the process of writing this dissertation and during my studies. I would also like to thank the pre-examiners of this dissertation, György Dán and Mika Ylianttila for their comments that helped to improve the quality of the dissertation.

The research for this thesis was carried out in the context of four TEKES projects: Decentralized Inter-personal Communication (DECICOM), ICT SHOK Future Internet, Device Interoperability EcosysteM (DIEM), and Massive-Scale Machine-to-Machine Service (MAMMotH). I would like to thank my fellow researchers with whom I worked in these projects. I

would also like to thank Ericsson and TEKES for supporting the work on the publications of this dissertation.

Finally, I would like to express my deepest gratitude to Johanna, Nella, my mother, and the rest of my family for their love, patience, and support.

Nummela, March 26, 2013,

Jouni Mäenpää

Contents

List of Publications

This thesis consists of an overview and of the following publications which are referred to in the text by their Roman numerals.

I Jouni Mäenpää and Gonzalo Camarillo. Study on Maintenance Operations in a Chord-based Peer-to-Peer Session Initiation Protocol Overlay Network. In *2009 IEEE International Symposium on Parallel and Distributed Processing (IPDPS '09), Seventh International Workshop on Hot Topics in Peer-to-Peer Systems (Hot-P2P)*, Rome, Italy, pp. 1-9, May 2009.

II Jouni Mäenpää and Gonzalo Camarillo. Analysis of Delays in a Peer-to-Peer Session Initiation Protocol Overlay Network. In *7th IEEE Consumer Communications and Networking Conference (CCNC)*, Las Vegas, USA, pp. 1-6, January 2010.

III Jouni Mäenpää and Jaime Jiménez Bolonio. Performance of REsource LOcation And Discovery (RELOAD) on Mobile Phones. In *2010 IEEE Wireless Communications and Networking Conference (WCNC)*, Sydney, Australia, pp. 1-6, April 2010.

IV Jouni Mäenpää and Gonzalo Camarillo. Estimating Operating Conditions in a Peer-to-Peer Session Initiation Protocol Overlay Network. In *2010 IEEE International Symposium on Parallel and Distributed Processing (IPDPS '10), Eight International Workshop on Hot Topics in Peer-to-Peer Systems (Hot-P2P)*, Atlanta, USA, pp. 1-6, April 2010.

V Jouni Mäenpää, Veera Andersson, Ari Keränen and Gonzalo Camarillo.

Impact of Network Address Translator Traversal on Delays in Peer-to-Peer Session Initiation Protocol. In *2010 IEEE Global Telecommunications Conference (GLOBECOM 2010)*, Miami, USA, pp. 1-6, December 2010.

VI Jouni Mäenpää. Performance evaluation of Recursive Distributed Rendezvous based service discovery for Peer-to-Peer Session Initiation Protocol. *Elsevier Journal on Computer Networks*, Volume 56, Issue 5, pp. 1612-1626, March 2012.

VII Jouni Mäenpää, Jaime Jiménez Bolonio and Salvatore Loreto. Using RELOAD and CoAP for wide area sensor and actuator networking. Accepted for publication in *EURASIP Journal on Wireless Communications and Networking*, Volume 2012, Number 1, pp. 121, March 2012.

VIII Jouni Mäenpää. Reducing P2PSIP Session Setup Delays. In *2013 IEEE International Conference on Computing, Networking and Communications (ICNC 2013)*, San Diego, USA, pp. 1-6, January 2013.

Author's Contribution

Publication I: "Study on Maintenance Operations in a Chord-based Peer-to-Peer Session Initiation Protocol Overlay Network"

The author of this dissertation fully implemented the prototype used in the publication, including implementations of the P2PP peer protocol and the Chord DHT. He also developed scripts for running the prototype in the PlanetLab network and for collecting the measurements results. The author carried out the PlanetLab measurements and analyzed their results. He also wrote the paper.

Publication II: "Analysis of Delays in a Peer-to-Peer Session Initiation Protocol Overlay Network"

The author of this dissertation fully produced the software implementation used in the publication. This included integrating a SIP stack to the P2PSIP prototype that the author had previously implemented and adding other new functionality whose performance was analyzed in the paper. The author also performed and prepared the PlanetLab measurements, collected the results of the measurements, analyzed the results, and wrote the paper.

Publication III: "Performance of REsource LOcation And Discovery (RELOAD) on Mobile Phones"

The author of this dissertation had the main idea for this publication. Additionally, the author of the dissertation wrote the paper, contributed to the prototype implementation, contributed to running the experiments, and performed the statistical analysis of the results.

Publication IV: "Estimating Operating Conditions in a Peer-to-Peer Session Initiation Protocol Overlay Network"

The author of this dissertation had the original idea for the new operating condition estimation mechanisms presented in this publication. Further, the author of the dissertation wrote the paper, fully implemented the new P2PSIP simulator used to carry out the experiments, produced the software implementation of the new mechanisms, prepared and carried out the simulations, and performed the statistical analysis of the results.

Publication V: "Impact of Network Address Translator Traversal on Delays in Peer-to-Peer Session Initiation Protocol"

The author of this dissertation was the main editor of this paper. His contribution consisted of providing the main idea for the paper, implementing most of the prototype used in the measurements, porting the P2PSIP prototype to J2ME, running the experiments in PlanetLab and on mobile phones, collecting the measurement results, and performing the statistical analysis of the results.

Publication VI: "Performance evaluation of Recursive Distributed Rendezvous based service discovery for Peer-to-Peer Session Initiation Protocol"

The author of this dissertation was the sole author of this paper and did all the work associated with it. He had the original idea for the paper, produced all the software required by the simulations, executed the simulations, and carried out the analysis of the simulation results.

Publication VII: "Using RELOAD and CoAP for wide area sensor and actuator networking"

The author of this dissertation had the original idea for the CoAP usage for RELOAD presented in this publication. Further, the author of the dissertation wrote the software implementation used in the simulations. This included implementing the CoAP protocol, integrating it into the simulator, and modifying the simulator to support simulated sensor devices and the WASAN architecture proposed in the paper. The author also prepared and carried out the simulations, performed the analysis of the results, and wrote the paper.

Publication VIII: "Reducing P2PSIP Session Setup Delays"

The author of this dissertation was the sole author of the paper and did all the work associated with it. He had the original idea for the mechanisms presented in the paper, produced the software implementation, prepared and carried out the simulations, and performed the statistical analysis of the results.

List of Abbreviations

3G	Third Generation
6LoWPAN	IPv6 over Low power Wireless Personal Area Networks
ACK	ACKnowledgement
ADSL	Asymmetric Digital Subscriber Line
ALEX	Address List Extension
ALG	Application-Layer Gateway
ALM	Application-Level Multicast
ALTO	Application-Layer Traffic Optimization
AODV	Ad hoc On-Demand Distance Vector
AoR	Address of Record
APDMF	Address and Port Dependent Mapping and Filtering
API	Application Programming Interface
AS	Autonomous System
ASAP	AS Aware peer relay Protocol
ASP	Address Settlement by Peer-to-Peer
BEX	Base Exchange
BRAVIS	BRAndenburg VIdeo conferencing System
C/S	Client/Server
CAN	Content Addressable Network
CAPEX	Capital Expenditure

CoAP	Constrained Application Protocol
CON	Confirmable
CoRE	Constrained RESTful Environments
CPU	Central Processing Unit
CRUD	Create, Read, Update, Delete
CSN	Chord for Sensor Networks
CSP	Chord Secure Proxy
DHT	Distributed Hash Table
DNS	Domain Name Service
DNS-SD	DNS Service Discovery
DSDV	Destination-Sequenced Distance Vector
DTLS	Datagram Transport Layer Security
EIMF	Endpoint Independent Mapping and Filtering
FQDN	Fully Qualified Domain Name
GHT	Geographic Hash Table
GSM	Global System for Mobile Communications
GW	Gateway
HIP	Host Identity Protocol
HIP BONE	HIP Based Overlay Networking Environment
HIP-HOP	Distributed Transport Function in P2PSIP Using HIP for Multi-Hop Overlay Routing
HSDPA	High-Speed Downlink Packet Access
HSPA	High-Speed Packet Access
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
ICE	Interactive Connectivity Establishment
ICS	Internet Coordinate System

ID	Identifier
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IMS	IP Multimedia Subsystem
IP	Internet Protocol
IPSec	IP Security
ISDN	Integrated Services Digital Network
ISUP	ISDN User Part
J2ME	Java Micro Edition
J2SE	Java Standard Edition
JR	Join Rate
JVM	Java Virtual Machine
LN	Local Node
LR	Leave Rate
LTE	Long-Term Evolution
M2M	Machine-to-Machine
M2PP	Mobile Peer-to-Peer Protocol
MCN	Monitoring and Control Node
MUX	Multiplexing
NAT	Network Address Translation
NON	Non-Confirmable
NS	Network Size
OPEX	Operational Expenditure
P2P	Peer-to-Peer
P2PP	Peer-to-Peer Protocol
P2PSIP	Peer-to-Peer Session Initiation Protocol

PBX	Private Branch Exchange
PC	Personal Computer
PN	Proxy Node
PnPAP	Plug-and-Play Application Platform
PNS	Proximity Neighbor Selection
PSTN	Public Switched Telephone Network
RAI	Real-Time Application and Infrastructure
RAN	Radio Access Network
ReDiR	Recursive Distributed Rendezvous
RELOAD	REsource LOcation And Discovery
REST	Representational State Transfer
RFC	Request For Comments
RPC	Remote Procedure Call
RST	Reset
RTP	Real-Time Transport Protocol
RTT	Round-Trip Time
SDP	Session Description Protocol
SEP	Service Extensible P2P Peer Protocol
SHA-1	Secure Hash Algorithm One
SigComp	Signaling Compression
SIP	Session Initiation Protocol
SLA	Service Level Agreement
SLP	Service Location Protocol
SOM	Self-Organizing Map
SSDP	Simple Service Discovery Protocol
STUN	Session Traversal Utilities for NAT

SvO	SIP via Overlay
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TTL	Time-To-Live
TURN	Traversal Using Relays around NAT
UA	User Agent
UAC	User Agent Client
UAS	User Agent Server
UDP	User Datagram Protocol
UDVM	Universal Decompressor Virtual Machine
UNSAF	UNilateral Self-Address Fixing
UPnP	Universal Plug and Play
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VCNF	Video Conference Network Foundation
VCP	Virtual Cord Protocol
VIPR	Verification Involving PSTN Reachability
VoIP	Voice over IP
VRR	Virtual Ring Routing
W3C	World Wide Web Consortium
WASAN	Wide Area Sensor and Actuator Network
WebRTC	Web Real-Time Communication
WG	Working Group
WN	Wide-area Node
WSN	Wireless Sensor Network
XML	eXtensible Markup Language
XPP	eXtensible Peer Protocol

1. Introduction

During the past decade, Peer-to-Peer (P2P) networking has emerged as a viable paradigm for providing services in the Internet. Well-known examples of successful P2P applications include file sharing, streaming, and real-time communication. The benefits that P2P technologies offer to applications include among other things self-organization, scalability, robustness, and reduced capital expenditures (capex) and operational expenditures (opex).

The tremendous success of P2P communication services, the prime example of which is Skype, gave rise to an initiative to create a standardized P2P communication service in the IETF. This initiative resulted in the creation of the P2PSIP Working Group (WG) in the Real-Time Applications and Infrastructure (RAI) area of the IETF. The mission of the P2PSIP WG became to create a general-purpose P2P signaling protocol. The protocol that the WG adopted as a working group item is called REsource LOcation And Discovery (RELOAD) [98]. RELOAD provides a generic overlay networking service that can be used also by other application protocols than SIP.

This dissertation presents a framework architecture for decentralized communications that is built around P2PSIP and the set of enabling technologies it uses, including RELOAD, Interactive Connectivity Establishment (ICE), the Chord Distributed Hash Table (DHT) algorithm, and the Session Initiation Protocol (SIP). The framework has been designed, and its performance analyzed, in the publications of this thesis.

The work on the framework began by the author of this dissertation implementing an early real-world prototype of a P2PSIP overlay network. While running the prototype in the global PlanetLab network [170], one of the author's first findings was that P2P algorithms are difficult to configure and suffer from scalability problems. To address these problems, the author ran a large set of measurements using the prototype to gain an understanding of how the system should be configured to make it robust. The findings in this area are presented in Publication I.

One challenge for a P2P system is that due to the need to route messages through several intermediate hops in the overlay network, the performance of the system is inherently lower than that of a client/server system. To gain an understanding of various delays in a P2PSIP system, including the call setup delay, the author of this dissertation ran delay measurements in PlanetLab using the P2PSIP prototype. Further, the author compared P2PSIP delays to those of client/server SIP. The results of these measurements are reported in Publication II.

A P2P system also typically consumes more resources such as Central Processing Unit (CPU) power, memory, and bandwidth on devices running it than a client/server system. Consequently, running a P2P system on mobile devices and networks is especially challenging. In Publication III, we studied the feasibility of using P2PSIP in mobile environments by running the P2PSIP prototype on low-end mobile phones.

To further address the problems associated with scalability and configuring overlay algorithms, the author of this dissertation developed novel mechanisms to make a P2PSIP system self-tuning, that is, able to monitor the operating conditions of the overlay network and adapt to them. These mechanisms are presented in Publication IV.

Any P2P system has to be able to cope with Network Address Translators (NATs). In Publication V, we study how Interactive Connectivity Establishment (ICE) based NAT traversal affects the performance of P2PSIP. One of the findings is that the use of ICE results in a considerable increase in the P2PSIP session setup delay.

The use of ICE in a P2PSIP overlay requires the nodes to be capable of discovering TURN relays from the overlay in a scalable way. To address the problem of discovery of TURN servers and other type of service providers, the author of this dissertation proposed a service discovery mechanism for P2PSIP in Publication VI. The author also developed optimizations to make service discovery more adaptive.

The P2P mechanisms that the P2PSIP working group develops, including RELOAD, are intended to be generic and not specific to a single application or application protocol. To expand RELOAD beyond the P2PSIP use case, we proposed in Publication VII an architecture for distributed Machine-to-Machine (M2M) networks that is based on a novel Constrained

Application Protocol (CoAP) usage for RELOAD that we designed.

As discussed above, the findings in Publication V and Publication VII showed that session setup delays can be unacceptably high for application usages of RELOAD. To address this issue, the author of this dissertation developed novel mechanisms for bringing the session setup delay down. These mechanisms and an analysis of their performance is the topic of Publication VIII.

The results obtained during the work on this dissertation were contributed to IETF standardization in the form of Internet drafts. At the time of writing this dissertation, two of these Internet drafts had become official P2PSIP working group items and had gone through the IETF working group last call procedure. One of the Internet drafts was still an early proposal in the working group.

Together, the results, mechanisms, and architectures presented in the publications of this dissertation form a generic and scalable framework architecture for decentralized communications.

1.1 Research Methodology

This dissertation presents new measurement results, mechanisms, and architectures that are utilized to develop a framework architecture for decentralized communications. The approach that was followed while working on the publications of this dissertation was to first produce a software implementation of the framework and its components and then analyze their performance. The performance evaluation was done either through real-life measurements or simulations. Thus, the main methods that were used were system design and performance analysis.

As a part of the work, the entire framework and its components were implemented. This was necessary since the components were not readily available either because we were the first ones to propose, implement, and contribute the components to standardization or because the standardization of the components was still ongoing while the work on this dissertation was being carried out.

The performance of the implementation of the framework and its components was verified through measurements and simulations. Whenever possible, the results were obtained by running the prototypes in the real Internet. Throughout the work, the PlanetLab [170] network was used extensively. When it was necessary to experiment with larger-scale systems that were infeasible to run in real networks, a simulator was used. A major part of the work associated with this dissertation consisted of implementing the prototypes and the simulator. The simulator that the author built and used has been validated in Publication VI. All of the results presented in this dissertation have been reported in scientific publications. Many of them have also been provided as input to the P2PSIP standardization process.

As more results were obtained and more experience was gained in running the implementation of the framework, several additional components from which the framework could benefit in order to better fulfill the requirements that will be listed in Chapter 2 were identified. After the missing components had been identified, they were designed and implemented, and their performance was evaluated.

In Publications I, II, III, and V, prototyping and performance evaluation through PlanetLab measurements was used as the main method. In Publications IV, VI, VII, and VIII, the main method was prototyping and performance evaluation through simulations.

The main result of the research presented in this dissertation is a framework architecture for decentralized communications that is scalable, adaptive, generic, modular, based on emerging standards, and has high performance.

1.2 Contribution to Knowledge

The main contributions to knowledge that this dissertation makes are as follows:

- The dissertation analyzes the impact of ICE on P2PSIP session setup delays, shows that these delays are unacceptably high, and proposes novel mechanisms to reduce the delays.

- The dissertation analyzes the performance and scalability of the ReDiR service discovery mechanism. Also a novel mechanism is developed to self-tune ReDiR parameters.

- A new distributed M2M architecture is proposed. The architecture is based on a novel CoAP usage for RELOAD that was designed as a part of the work on this dissertation. The proposed architecture is evaluated through simulations and using a real-world prototype.

- Through simulations and measurements in the real Internet, new results are produced on the performance of maintenance operations in P2PSIP overlays, on delays associated with various operations that a P2PSIP overlay provides to applications, and on the performance of P2PSIP in mobile networks and devices.

- It is shown that existing mechanisms for estimating operating conditions in a DHT-based overlay network are not accurate enough. To address this issue, new operating condition estimation mechanisms are designed and their performance analyzed. These mechanisms can be used to self-tune the parameters of DHT algorithms.

1.3 Structure of the Thesis

The remainder of the thesis is structured as follows.

Chapter 2 describes the research goal of this thesis and lists the requirements that the framework presented in the thesis should fulfill.

Chapter 3 provides background information on the technologies utilized in this thesis, including P2P technologies, communication protocols, NAT traversal, and M2M communication. In this chapter, also previous research that is relevant for the components of the framework is summarized.

Chapter 4 presents the proposed framework architecture for decentralized communications. The chapter begins with an overview of the framework and its components. Next, in Section 4.2, the setup that was used for evaluating the performance of the implementation of the framework is described. The remaining sections of the chapter present the components of the framework one by one. Section 4.3 summarizes our work on analyzing the impact of churn and maintenance operations on the performance of P2PSIP. In Section 4.4, the focus is on the performance of the basic API that the framework uses between the overlay networking layer and the application protocols. Section 4.5 is dedicated to our work on analyzing the impact of ICE on the performance of P2PSIP. Section 4.6 deals with session setup and ICE optimizations that the author of this dissertation developed for P2PSIP. Section 4.7 addresses issues related to adaptivity and self-tuning in P2PSIP overlays. Section 4.8 presents the work that the author of this dissertation did on service discovery in P2PSIP overlays. Section 4.9 presents a summary of the work we carried out to analyze the performance of the P2PSIP peer protocol in mobile environments. Section 4.10 describes our architecture for using RELOAD to decentralize M2M communication.

Finally, Chapter 5 presents the conclusions of this thesis and proposes topics for future research.

2. Research Goal

The research goal of this dissertation is to design a framework architecture for decentralized communications. The framework architecture is built around P2PSIP and the set of enabling technologies that it uses. This chapter will define what is meant by such a framework and list the requirements that the framework fulfills.

2.1 Framework Architecture for Decentralized Communications

In this dissertation, a framework architecture for decentralized communications refers to a flexible software framework consisting of a number of reusable and modular components organized in different layers. The framework does not represent a tightly coupled set of components or an integrated architecture; the flexible and modular design of the framework allows different applications to utilize the framework in different ways by using the components either individually or in different combinations. At the core of the framework is an overlay networking layer that exposes an API to application usages on the application support layer. The application usages are components utilizing the overlay networking layer in order to be able to run in decentralized mode. The framework will be described in more detail in Chapter 4.

The framework is built around the basic P2PSIP components, including RELOAD, ICE, and the Chord DHT. The framework consists of these components and a number of novel components that were designed as a part of the work on this dissertation.

2.2 Requirements for the Framework

In this section, the requirements for the framework will be described. The overall goal in the publications of this dissertation was to build a framework that is adaptive, scalable, generic, modular, based on current or emerging standards, and has high performance.

2.2.1 Adaptiveness

The most important challenge for P2P systems is the problem of churn, that is, the constant process of peer arrival and departure. To make structured overlay algorithms resistant to churn, they must be configured appropriately. One of the early findings in the research carried out for this dissertation was that a major shortcoming of structured overlay algorithms like the Chord DHT is that in order to configure the algorithm, the expected churn rate and network size need to be known in advance, even before the system is made operational. These estimates are used to configure the algorithm in a static fashion. The challenge with this approach is that if the operating conditions that the system faces during its lifetime do not match the expected operating conditions, the result is sub-optimal performance or even network partitioning. Therefore, the first requirement for the framework is that it needs to incorporate mechanisms that are able to adapt the system to changing operating conditions through self-tuning behavior.

2.2.2 Scalability

Another requirement for a framework architecture for decentralized communications is that it needs to be able to scale from small-scale use cases such as a VoIP network within a small company to large-scale use cases such as a distributed M2M system consisting of potentially hundreds of thousands of embedded devices. One important aspect of scalability is the adaptive behavior discussed in the previous subsection. However, also mechanisms beyond adaptive behavior are needed, including the ability to keep traffic levels low, and to discover and provide services in the overlay in a manner that distributes load evenly.

2.2.3 Generality

The framework needs to be generic enough so that it can be used to decentralize a wide range of application protocols and services. Therefore, the operations (i.e., the API) that the different layers of the framework offer to other layers must not be specific to or optimized for a single application. Further, the set of operations should be complete enough to fulfill the needs of applications ranging from M2M communication to real-time multimedia communication.

Another aspect of generality is that an implementation of the framework needs to be able to run on heterogeneous devices and networks; it should be able to run on devices ranging from embedded devices and mobile phones to desktop computers and servers, and also be able to accommodate even constrained devices such as sensors and actuators.

2.2.4 Modularity

The framework needs to have an extensible and modular internal architecture that enables it to utilize among other things different application protocols, overlay algorithms, and P2P signaling protocols. This is necessary since different environments in which the framework is used may place different requirements on its components. An example is the overlay algorithm; a small static overlay consisting of perhaps only tens of nodes does not require the same level of complexity from the overlay algorithm as a dynamic system having perhaps tens of thousands of nodes.

2.2.5 Based on Emerging and Approved Standards

To achieve wide adoption and interoperability, the framework should be based on accepted or emerging open standards. This makes it possible for nodes running independent implementations of the components to interoperate with each other and even participate in the same overlay network.

2.2.6 High Performance

The framework should be designed for high performance. It should offer mechanisms that can be activated to meet the performance requirements of different classes of applications. Examples of such requirements include low failure rate of operations, low traffic load, and low session setup time.

3. Background

This Chapter introduces the technologies that the framework architecture for decentralized communications presented in this dissertation builds on. These include P2P technologies, SIP, NAT traversal, P2PSIP, and M2M communication.

3.1 Peer-to-Peer Technologies

P2P systems represent a paradigm shift from the traditional client/server model, in which a participating node can either act as a server or as a client but cannot offer both capabilities at the same time. Typically, a client/server system consists of one high performance system, the server, and several mostly lower performance systems, which are the clients. The server is the only provider of content and services; clients only request content or services without sharing any of their own resources [191]. In contrast, in a P2P system, a participating node is capable of acting at the same time as a server as well as a client. The purpose of such a system is to employ distributed resources of the nodes to perform a selected function in a decentralized manner [153]. The distributed resources can include computing power, data, network bandwidth, and presence (e.g., of computers or users). The function can be distributed computing, data or content sharing, communication and collaboration, or platform services. The decentralization may apply to algorithms, data, or both.

The term peer-to-peer is widely associated with P2P file sharing applications due to the predominantly negative attention such systems have received in the general media. P2P file sharing applications started to gain momentum after the legal challenges that the Napster service, which popularized file sharing, experienced. Napster's Achilles' heel was its centralized register of file locations; although the files themselves were stored at individual users, Napster relied on a centralized register for storing the locations of files [38]. The centralized register allowed the Recording Industry Association of America (RIAA) to enforce a filter at the Napster site. Because of the central point of failure that the centralized register represented for Napster, the next generation of P2P file sharing applications such as Gnutella sought to replace central registers with a distributed one.

Besides file sharing, another example of a successful P2P application is real-time P2P communication, which is the focus of this dissertation. The prime example of a successful P2P communication application is Skype. Skype was created by the developers of the KaZaa P2P file sharing application in 2003. It was the first VoIP and instant messaging client based on P2P technology. The network that Skype uses is a P2P overlay network implemented using a protocol that is proprietary and encrypted. Skype functions such as login, NAT and firewall traversal, call establishment, media transfer, codecs, and conferencing have been investigated in [13]. It has been shown in [74] that the nature of the Skype system differs fundamentally from other P2P systems such as file sharing and music and video distribution.

Yet another example of a popular class of P2P applications is P2P video streaming. The self-scaling property of P2P architectures makes them especially attractive for the delivery of video streams. Self-scaling refers to the fact that every new peer that is added to the system also brings additional capacity; any peer receiving a video stream can also forward it to further peers. A survey of popular P2P streaming applications is presented in [73, 166]. These include, among others, Joost, Octoshape, PPLive, Zattoo, PPStream, SopCast, TVants, TVUPlayer, Veoh TV, PeerCast, and Coolstreaming.

Besides self-scaling, other characteristics of P2P networks include self-organization, symmetric communication, and distributed control [183]. A self-organizing system automatically adapts to the arrival, failure, and departure of nodes [189], a phenomenon that is also known as churn. Communication is symmetric in nature since peers act both as clients and servers. Control is distributed since there is no central entity controlling the system and no central index or directory. The benefits of P2P systems and the challenges that they face are discussed in the subsections below.

3.1.1 Benefits

The benefits of P2P systems include low barrier of entry, self-scaling, low operating cost [217], robustness, easy configuration [46], and easy maintenance [193]. P2P architectures provide a low barrier of entry for service providers. This is because little hardware or network investment is needed to launch a P2P application as the architecture relies on resources of the participating peers to provide the service. Such easy and cheap deployability makes P2P systems attractive for providing low-cost services [216]. The self-scaling property was already discussed in the context of P2P video streaming above. A P2P architecture reduces operational expenditures compared to a centralized architecture since fewer entities have to be installed and operated [23]. The systems are robust since they have no or less central points of failure than client/server systems. P2P systems also employ mechanisms such as data replication, redundant routing table entries, larger neighbor sets, and hierarchy to improve reliability. P2P systems are easier to configure than centralized systems due to their self-configuration properties, including especially the ability of the peers to organize themselves in a structured overlay network. The self-configuration and self-organization properties of the systems also mean that they are easier to maintain than client/server systems.

3.1.2 Challenges

The challenges affecting P2P systems include churn, heterogeneity, load balancing, connectivity, and various security issues. Churn, that is, the continuous process of node arrival and departure, causes a number of negative effects on P2P systems, including latency growth and network partitioning [180].

Although P2P algorithms are symmetric, meaning that all peers play the same role, the devices running the P2P software can be highly heterogeneous [207], ranging from mobile phones in narrowband wireless access networks to powerful servers having broadband Internet connectivity. Heterogeneity of peer capabilities is also one but certainly not the only factor causing the problem of load balancing. Load imbalance may also follow from the way DHT algorithms distribute objects randomly among peers, non-uniform distribution of objects and peers in the identifier space, heterogeneity in object loads, continuous insertions and deletions of objects, skewed object arrival patterns, and churn [70]. The problem of load balancing in DHT-based P2P systems has been studied extensively in the literature. The proposed mechanisms can be divided into four different categories [85]. The first category consists of mechanisms based on the concept of virtual servers. This approach is used as a basis for many load balancing algorithms, including [205, 175, 49, 70, 72]. In this approach, a single physical node may host several virtual nodes, also known as virtual servers. These virtual nodes are participating independently in the same overlay network. The benefit of the use of virtual servers is that it makes the number of objects stored per node more uniform. A slight modification to the virtual server approach is to allow only one of the virtual servers of a node to be active at any given time [104]. The second class of load balancing mechanisms takes the approach of directly controlling the location of objects in the overlay. One example of this approach is the application of the power of two choices paradigm [10, 154] to balance load [31]. This approach uses multiple hash functions per object. When storing an object into the overlay, these hash functions are used to calculate alternative keys for the object. After calculating the alternative keys, the load experienced by the peers responsible for these keys is retrieved and the object is stored in the least loaded peer. The third category of load balancing mechanisms controls the location of nodes in the overlay. Karger and Ruhl [104] propose a mechanism in which underloaded nodes migrate to the portions of the address space occupied by too many objects. Another similar mechanism is proposed in [182]. The fourth and final category of load balancing algorithms attempts to balance the address space between the nodes in the overlay network. The majority of algorithms in this category, such as [141, 112, 18], attempt to minimize the variation between partition sizes, meaning that they try to ensure that the nodes are responsible for equally sized partitions of the total identifier space of the overlay. In [89], we propose a mechanism that takes the partition sizes of nodes into account when selecting entries to the routing table of a DHT algorithm.

P2P systems also face connectivity challenges due to the ubiquitous deployment of NATs and firewalls in the present-day Internet [202]. NATs cause problems for P2P applications since peers located behind a NAT device normally do not have permanently visible public addresses and ports on the Internet to which incoming TCP and UDP connections from other peers can be directed. NAT traversal will be discussed in more detail in Section 3.3.

Security

The security issues associated with P2P systems have received a great deal of attention from the research community. From the viewpoint of this dissertation, the most relevant security issues are those of structured overlays. Chopra et al. [46] discuss security problems of structured overlays for real-time communication. These include attacks on the routing of queries, targeted denial of service attacks, and attacks on data integrity [46, 192]. In [193], security challenges for P2PSIP are discussed. The challenges include attacks on the Node-ID mapping scheme, attacks on overlay routing, malicious bootstrap nodes, identity enforcement without a trusted central authority, free riding, lack of anonymity, provision of reliable and secure emergency services, lawful interception, and spam prevention. Reliable real-time P2P communication also requires each user to be unique and discoverable without false negatives [26].

Many of the security problems that P2P systems face are caused by the presence of malicious nodes in the overlay. Such nodes may misroute, corrupt, or drop messages and routing information [42]. They may also attempt to steal the identity of other nodes and corrupt or delete resource objects that these nodes are responsible for. Malicious nodes that are able to choose Node-IDs can compromise the integrity of the overlay without the need to control a particularly large fraction of the nodes. Such nodes may also run targeted attacks on selected victims. The most efficient way to secure the assignment of Node-IDs is to delegate the task to a trusted central authority that assigns Node-IDs and certificates [42].

Even if Node-ID assignment has been secured, an attacker may still be able to obtain a large number of legitimate Node-IDs. This is known as the Sybil attack [55]. Solutions to Sybil attacks include among others charging money for certificates and binding Node-IDs to real-world identities [42].

Besides secure Node-ID assignment, other requirements for secure routing in structured overlays include secure routing table maintenance, and secure message forwarding [42]. Secure routing table maintenance ensures that the fraction of malicious nodes in the routing tables of correct nodes does not exceed the fraction of malicious nodes in the entire overlay. Secure message forwarding ensures that at least one copy of a message sent to a key reaches the correct peer responsible for the key with a high probability.

Free-riding represents a problem for P2P systems since P2P systems typically rely on altruistic behavior in the form of voluntary contribution of resources from peers. However, in the absence of central control, some peers may choose to free-ride rather than to contribute their resources to the system [62]. This has generated a large body of research that attempts to design incentive mechanisms for P2P systems to overcome the free-riding problem [63].

Lawful interception of P2PSIP communications is problematic due to there being no central entity through which signaling will pass. Other challenges include changing participants and varying data responsibility. Potential solutions to this problem are analyzed in [194]. However, the conclusion is that providing a general solution for lawful interception is technically highly challenging and therefore represents an open research problem.

Despite of the large body of research on security of P2P systems, fully distributed security is still an ongoing research problem [192]. For this reason, the RELOAD P2P signaling protocol that P2PSIP uses relies on a central authority to verify credentials, and to assign Node-IDs and certificates. Although dependence on a centralized authority goes against the ideal of a fully distributed system, its use is unavoidable due to the lack of a reliable enough distributed security mechanism. RELOAD uses certificates to sign each message and each resource record stored in the overlay. Further, RELOAD also uses secure TLS or DTLS transport for connections between nodes in the overlay. The benefit of RELOAD's security model is that it addresses a large number of security problems that a large-scale, publicly accessible DHT system faces [26].

3.1.3 Overlay Networks

The collection of connections between peers in a P2P system is called a P2P overlay network. An overlay network is virtual, application-level network topology built on top of the physical network. There are two main approaches to building P2P overlay networks: structured and unstructured [4, 183, 46]. Unstructured P2P overlay networks are constructed in an uncontrolled fashion; a peer that joins the system can connect to any of the existing peers. The advantage of unstructured overlay networks is that they are flexible when it comes to lookup operations, that is, they can support arbitrary search queries such as wildcard searches. The disadvantage of unstructured overlays is that they cannot provide performance guarantees. This is because even if the system contains the resource being searched for, no guarantee can be given that the data can be located due to the unstructured nature of the system. This inability to avoid false negatives is one of the most important disadvantages of unstructured systems from the viewpoint of real-time P2P communication [26]. Unstructured overlays are also inefficient since they typically need to rely on flooding or random walks as the main search mechanism.

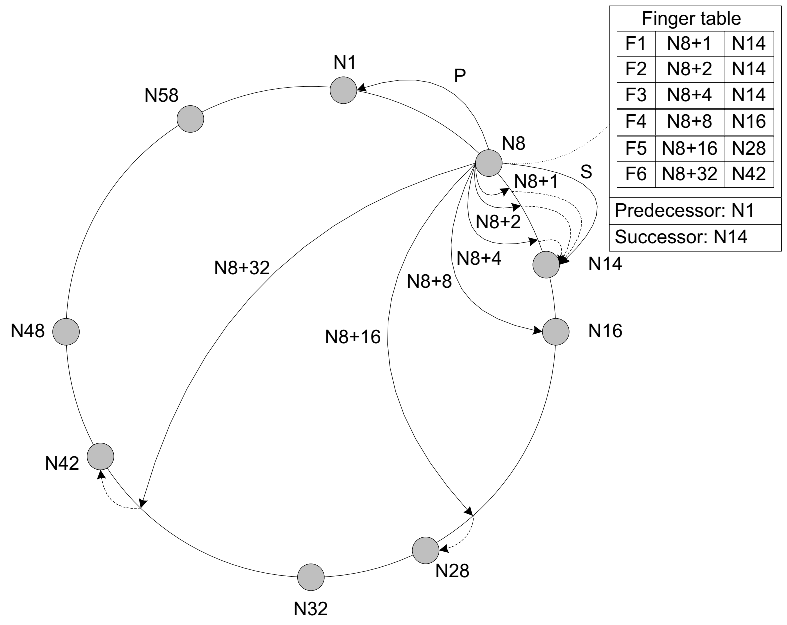
Structured overlay algorithms emerged to address the limitations of unstructured algorithms. In structured overlay networks, the identifier of a peer defines its place in the topology of the overlay network, indicating to which other peers it should connect. The most significant advantage of structured overlay networks is that they can provide performance guarantees. They have efficient key-based routing and are able to locate even rare objects. They have also been shown to provide efficiency and scalability in a wide range of applications [41]. Although structured overlay algorithms cannot readily support arbitrary searches, mechanisms for adding such support have been designed [41]. The main class of structured overlay networks are the Distributed Hash Table (DHT) based systems. The most well-known DHT algorithms include Chord [205], Pastry [189], Kademlia [151], Tapestry [220], and Content Addressable Network (CAN) [176].

Although in this dissertation, P2P overlay networks are divided into structured and unstructured systems, also other more fine-grained taxonomies can be used as discussed in [34]. However, the division into structured and unstructured systems is sufficient for the purposes of this dissertation. The focus of this dissertation is solely on structured overlay networks due to their better suitability for real-time P2P communication, which is also demonstrated by the P2PSIP working group's choice to use a DHT algorithm, namely Chord, as the mandatory-to-implement overlay algorithm for RELOAD.

3.1.4 Chord

Chord [205] is one of the most studied structured overlay algorithms. It is a DHT algorithm that uses consistent hashing to build a structured overlay network out of the participating peers. Consistent hashing assigns each peer and resource record an m-bit identifier using SHA-1 [58] as the base hash function. The keys are ordered on an identifier circle of size 2^m, which is called the Chord ring.

On the Chord ring, such as the one shown in Figure 3.1, each peer main-



Finger table

F1	N8+1	N14
F2	N8+2	N14
F3	N8+4	N14
F4	N8+8	N16
F5	N8+16	N28
F6	N8+32	N42

Predecessor: N1

Successor: N14

Figure 3.1. Chord ring

tains a routing table consisting of a finger table, successor list, and predecessor list. In an N-node network, the finger tables of peers contain information about $O(\log N)$ other peers. The successor list contains the peer's immediate successors on the Chord ring. A typical approach is to set the size of the successor list to $O(\log N)$. Although the original Chord algorithm maintains information about only one predecessor, some Chord implementations such as the one specified in the RELOAD specification [98], also maintain a predecessor list to increase the stability of the overlay; if a peer maintains only one predecessor and loses it, the peer cannot make reliable routing decisions until it learns the identity of a new predecessor. Depending on the frequency of stabilization operations, this time may be long enough to result in an increase in the number of failed routing decisions.

Figure 3.1 shows an example of a Chord ring that uses 6-bit identifiers (i.e., m=6). The use of 6-bit identifiers results in an identifier space whose size is 64. In the figure, the routing table of Node 8 (*N8*) is depicted. The routing table consists of the finger table, a predecessor list, and a successor list. The sizes of the successor and predecessor lists are one, whereas the size of the finger table is six entries. In the figure, the single predecessor pointer is denoted by P, and the successor pointer by S. The i^{th} entry in the finger table of *N8* contains the identity of the first node that succeeds *N8* by at least 2^{i-1} on the Chord ring. As an example, the 6th entry in the finger table contains the first peer that succeeds *N8* by at least 32. This peer is *N42*.

As already mentioned, the Chord DHT algorithm is of special relevance to P2PSIP since the P2PSIP working group has specified it as mandatory to implement.

Maintenance

A major challenge with which DHTs such as Chord need to deal with is churn. One metric of churn is node session time, which is defined as the time between the node joining the network and subsequently leaving it. The negative impacts of churn on DHTs include latency growth, suboptimal routing decisions, failed lookups, inconsistent lookup results, data inconsistency, increased bandwidth usage, and network partitioning [180, 71].

The impact of churn on P2P systems has been studied extensively in the literature. Stutzbach and Rejaie [206] study churn in P2P file sharing networks, including Kad (which is used by the eMule P2P file sharing software), Gnutella, and BitTorrent. The work identifies several properties of churn in file sharing networks. One of the findings is that a large portion of the active peers in the network are highly stable whereas the rest of the peers have fairly low session times. In [180], it is shown that in a file sharing system, the session times of nodes can be as low as a few minutes.

Guha and Jain [74] study churn in Skype's super peer network. The findings indicate that there is very little churn in the super peer network. Further, the super peers demonstrate diurnal behavior, which causes median session times to be of several hours, the median being 5.5 hours. Skype usage peaks during normal working hours and is significantly reduced at night. For super peers, Skype usage is correlated with normal working times. All in all, user behavior in the Skype network was found to be rather different from P2P file sharing networks.

Li et al. [129] compare the performance of different DHTs under churn. The studied DHTs include Tapestry, Chord, Kelips [78], and Kademlia.

The results indicate that the DHTs can achieve similar performance under churn if their parameters are tuned sufficiently well. The parameters having the greatest effect on DHT performance under churn are maintenance interval and identifier base (as an example, the default identifier base of Chord is 2). The effect of base depends on the size of the network, whereas the effect of the maintenance interval depends on the average session time of churning nodes.

Kassinen et al. [106] analyze the impact of churn on a simulated Kademlia-based Peer-to-Peer Protocol (P2PP) overlay network that is experiencing churn. The results suggest that churn has a clear impact on lookup success ratios; under the churn rates studied, the lookup success rate varied between 30% (2000-peer overlay experiencing heavy churn) and 70% (200-peer overlay experiencing lighter churn).

Ou et al. [161] study the impacts of different churn models on the performance of a simulated 200-peer Kademlia-based P2P network. The studied churn models include exponential distribution, Pareto distribution, and Weibull distribution. The results indicate that churn model does not have a significant impact on the results of the simulations.

Gummadi et al. [76] study how the geometry of DHTs affects their performance in the areas of static resilience and proximity routing. Static resilience refers to how well a DHT can route under churn even before the maintenance routine has had a chance to update the routing table as a response to peer departures and arrivals. Proximity refers to the total latency of the DHT routing paths and their local convergence. The main finding is that the ring geometry, which is used for instance by Chord, achieves the best resilience and proximity performance.

There are two approaches to handling churn in a DHT: reactive recovery and periodic recovery [180]. In reactive recovery, a peer reacts to the failure or departure of a neighboring node by immediately starting to search for a replacement. A significant downside with reactive recovery is that it can lead to a positive feedback cycle that overloads the network [180]. In periodic recovery, a peer recovers from neighbor failure at a fixed, periodic rate. Periodic recovery has been shown to improve performance of a DHT under churn by allowing the system to avoid positive feedback cycles [180].

The strategy that Chord uses for dealing with churn is periodic recovery, also known as periodic stabilization or periodic maintenance. The periodic maintenance routine is run in order to keep the contents of the finger table, successor list, and predecessor list up to date with the topology of the overlay that changes constantly as a result of churn. As a part of the maintenance routine, a peer synchronizes its predecessor list with its first predecessor and its successor list with its first successor, and incorporates new peers into its finger table.

The main challenge with periodic maintenance is choosing an appropriate maintenance rate. On the one hand, if the maintenance rate is too low, a peer cannot recover fast enough from the loss of its neighbors. On the other hand, if the maintenance rate is too high, the result is wasted bandwidth, energy, and potential network overload. To address this challenge and also other challenges associated with configuring DHTs, various mechanisms have been developed. These mechanisms are discussed in the next subsection.

Adaptive Behavior

The traditional approach to configuring DHTs is to do it in a static fashion, that is, to use fixed values for parameters such as the maintenance interval and the size of the routing table (which in the case of Chord includes setting the sizes of the predecessor list, successor list, and finger table). The problem with this approach is that in the absence of perfect information about the future, it is very difficult to choose values that would remain appropriate in all possible operating conditions that an overlay network faces during its lifetime. Therefore, rather than using fixed values to configure a DHT, it would be more efficient if the system can determine its operating conditions and adapt to them by dynamically adjusting its configuration parameters.

The techniques that an overlay network can use to adjust its parameters to match the prevailing operating conditions are referred to as self-tuning. A typical approach to self-tuning is to estimate three parameters: network size N, join rate λ, and leave rate μ. The network size estimate N is used to adjust the size of the routing table. The maintenance rate is set based on the estimates for N, λ, and μ.

The approaches to estimating operating conditions can be divided into two coarse categories: active mechanisms and passive mechanisms. Passive mechanisms do not add extra messages to the overlay. Instead, all the information used in the estimation process consists of information that is locally available to the node computing the estimates. In contrast, active mechanisms for operating condition estimation add extra traffic to the overlay. Active approaches can be divided into three categories [127]. The first category is probabilistic polling and randomized report techniques, which probe the overlay in a probabilistic way and infer properties of the system from the received replies. In these techniques, the initiating node broadcasts a message to all nodes in the overlay. The second category is made up of epidemic, also known as gossip-based, algorithms that gather information from the system by constantly exchanging information between peers in a random manner. The third class of techniques relies on random walks in the overlay.

Mahajan et al. present a self-tuning mechanism for the Pastry DHT in [139]. In this mechanism, the density of Node-IDs in the Pastry leaf set is used to estimate N. The value of μ is estimated using observed node failures in the leaf set and routing table. The mechanism to estimate N and μ are passive in nature. The join rate λ is not estimated.

Ghinita and Teo [69] present an adaptive maintenance framework for Chord. The density of Node-IDs in Chord's successor list is used to estimate N. The leave rate μ is estimated by maintaining a failure history of observed node failures. The node join rate λ is estimated using the online times of nodes in the routing table. Online time refers to the difference between the current time and the time when the peer joined the overlay. The presented estimation mechanisms are passive.

The appropriate rate for Chord stabilization operations is studied in [132]. The main finding is that a Chord network that is in a ring-like state stays in the ring-like state as long as nodes send $\Omega(\log^2 N)$ messages in the time it takes for N new nodes to join or $N/2$ existing nodes to leave the system. This guideline makes it possible to calculate an appropriate value for the Chord maintenance rate provided that estimates for N, μ, and λ are available.

In [138], we propose how to add self-tuning behavior to the Chord variant that RELOAD uses. The approach is passive. The size of the network is estimated using the density of node-IDs in the successor and predecessor lists. Join rate is estimated using the online times of peers in the routing table following the approach in [69]. Leave rate estimation is based on keeping track of peer failures in the routing table using the algorithm presented in [139]. For determining the maintenance rate, the guidelines in [132] are used.

Shafaat et al. [197] present a gossip-based aggregation-style network size estimation mechanism. The mechanism is active since it adds traffic to the overlay by performing a random walk in it. The network size estimate is produced by using Node-ID distances collected during the random walk.

In [19], an active algorithm that takes a snapshot of a running Chord overlay is presented. The snapshot is used for determining the current state of the overlay. The algorithm operates by dividing the overlay into subparts, performing a measurement for each subpart utilizing token passing, and sending the measurement result back to a central collecting point. The snapshot algorithm can be used to estimate among other things the network size and the churn rate.

Kostoulas et al. [124, 173] propose two size estimation algorithms for structured overlay networks. The first is an active approach called Hops Sampling. This approach is similar to [197] in that it relies on gossip requests. The hop count values in the returned gossip responses are used to produce an estimate of the network size. The second size estimation algorithm is a passive approach called Interval Density. This approach is similar to that in [139] since it uses the density of the identifier space to estimate N.

Binzenhofer and Leibnitz [20] introduce a mechanism to estimate churn in structured overlay networks. As in [139, 69], the estimate is based on the number of changes a peer observes in its neighborhood set. In addition to producing a local estimate, nodes also share observed changes with their direct neighbors. The mechanism makes the assumption that peers that leave the system will rejoin the system at a later time; the estimate is calculated using information about the duration of the periods during which peers stay online and offline.

In [91], a passive approach that estimates network size from local information is presented. The accuracy of the mechanism is rather limited as the estimate it produces is within the range $\frac{N}{2}...N^2$.

A passive mechanism that estimates N in a Chord overlay using the density of identifiers on the successor list and the distance of finger pointers from their ideal positions is presented in [21]. The accuracy of the estimator is shown to be roughly between $\frac{N}{2}...2 \times N$ through simulations in which no churn was present. A uniform distribution of Node-IDs in the identifier space was assumed while evaluating the accuracy.

Massoulié et al. [147] present two random-walk based active mechanisms for size estimation in overlay networks. These mechanisms are called *random tour* and *sample and collide*. The random tour method is based on statistics collected during a random walk along the overlay. The sample and collide method is inspired by the birthday paradox [17] and is based on counting the number of random samples gathered until a target number of redundant samples are obtained.

Bawa et al. [17] propose a few flooding and random walk based active approaches to network size estimation. The cost of the flooding based approaches is high, $O(logN)$ as they need to probe a large number of nodes in the overlay. The *BdayParadox* algorithm is based on the birthday paradox and random walks. The birthday paradox states that for $\sqrt{N}$ independent samples from a population of size *N*, the probability that a pair of samples will have the same value is at least 0.5. Using this approach, the cost of the size estimation algorithm is reduced to $O(\sqrt{N}logN)$, which is still very high. The final mechanism is *RandomIncWalk*, which is based on performing random increasing walks. The cost of this approach grows logarithmically with *N* but its performance has only been verified in random graph network topologies.

To summarize, mechanisms for estimating the operating conditions of a P2P overlay network can be classified to two coarse categories: passive mechanisms and active mechanisms. The general downside of active mechanisms is the high communication cost that they incur due to the fact that they add extra messages to the overlay either through broadcasting, gossiping, or random walks. Passive mechanisms have better scalability since they rely only on information that is locally available for a peer, such as the density of node-IDs in the routing table, online times of peers in the routing table, and the number of failures observed. However, the accuracy of existing passive mechanisms may not always be sufficient, as we show in Publication IV.

3.1.5 Service Discovery and Registration

The ability to discover services from a P2P overlay network is crucial for many P2P applications. One example of a service that is necessary to nearly all P2P systems operating in the Internet is a relay service that may be needed due to the presence of the most restrictive types of NATs. In a P2PSIP network, other possible services include a voice mail service, IMS or Public Switched Telephone Network (PSTN) gateway service, conference focus service, and transcoding service.

Since service discovery is a term used in many different contexts, this subsection begins with the definition of what service discovery means in the context of structured overlay networks. Well-known service discovery protocols such as the Service Location Protocol (SLP) [79], Simple Service Discovery Protocol (SSDP) for Universal Plug and Play (UPnP) [209] and Service Discovery Protocol (SDP) for Bluetooth [8] are designed for local area networks. These solutions address a different problem than that of finding services within a structured P2P overlay network. Further, these conventional service discovery protocols rely on broadcast protocols or central server repositories and thus do not scale to P2P networks. Therefore, a number of service discovery algorithms optimized for P2P overlay networks have been proposed.

In this dissertation, service discovery refers to the process through which a node participating in an overlay network can find providers of a given service from the same overlay network. In contrast, service registration refers to the process that a node follows to register itself as a service provider for a given service in the overlay. The problem of discovering services within a single P2P overlay network should not be confused with global or wide area service discovery, in which a device connected to the Internet attempts to discover services offered by any other device in the Internet [30]. Global service discovery using P2P technologies is discussed for instance in [43]. This approach uses a top tier universal overlay network interconnecting various lower tier overlay networks.

A naive way to perform service discovery and registration in a structured overlay network is to store the Node-IDs of all providers of a service under a well-known key uniquely identifying the service, such as a SHA-1 hash over the well-known string *"turn-service"*. The problem with this approach is that it does not scale, since in a large system, the approach can cause a high lookup and storage load on the unfortunate peer that is responsible for the well-known key.

Service discovery algorithms for P2P systems can be classified into two coarse categories: proximal and non-proximal algorithms. Proximal algorithms attempt to discover nearby service providers and are typically optimized to a specific use case such as relay discovery for VoIP or P2P streaming. In contrast, non-proximal algorithms do not take proximity into account but are more generic in nature. Proximal algorithms incur a higher cost than non-proximal ones due to the need to maintain information about communication latencies between nodes.

A proximal node discovery algorithm is proposed in [158] for a P2P streaming system. The algorithm relies on Internet Coordinate Systems

(ICS) [133]. The algorithm partitions the coordinate space and stores location information of service nodes in a DHT. One disadvantage of the algorithm is that the coordinates need to be maintained, which adds extra traffic to the overlay. In addition, the algorithm uses fixed parameters such as the number of levels and partitions, which may hinder scalability.

Ren at al. propose an Autonomous System (AS) Aware peer-relay Protocol (ASAP) in [178]. ASAP uses an annotated graph to select relay nodes for P2P communication. ASAP relies on the existence of powerful peers called cluster surrogates that have sufficient resources for maintaining the AS graph. The cluster surrogates are also responsible for maintaining a cluster set, which consists of a set of nearby nodes. The maintenance of clusters and the AS graph add extra traffic to the overlay.

Dowling et al. [56] propose the use of a gradient topology [190] for discovering relay servers in a P2P network. In the gradient topology, peers use a local utility to adapt their connections to other peers. The peers with the highest utility value, which is calculated based on available bandwidth and session time, are clustered in the center of the topology. The architecture divides peers into three categories: NATed peers, normal super peers, and super peers capable of providing an ICE service. Only super peers are connected using the gradient topology; NATed peers are clients of the gradient topology. The downside of this approach is that it defines a new overlay network topology and is thus not suitable for DHT-based systems.

A distributed relay selection technique is proposed in [15]. The approach is to construct a two tier P2P network. All peers in the top tier provide routing and relay services. The nodes in the lower tier connect to nearby (in terms of network latency) peers in the top tier. A peer in the lower tier performs relay discovery by asking for a list of relays from a peer in the top tier. The downside of this approach is that it distributes the cost of maintaining the P2P system unevenly among the publicly reachable (top tier) and NATed (lower tier) peers.

Zhou et al. [224] present a service advertisement and discovery model for call services in a P2P overlay. In the model, service providers store information about their services in the overlay. Peers using the services discover service providers using DHT lookups. Information about the services and their attributes is encoded in the key under which the record of the service is stored in the overlay. The downside of this approach is that it places an upper limit on the services in the overlay and also on the attributes of the services. Further, the proposed key assignment scheme does not distribute the records uniformly in the overlay and is incompatible with protocols like RELOAD due to the need to encode information in the keys.

Service discovery in a structured overlay could also be implemented through a broadcast algorithm, such as the one specified in [59]. This algorithm constructs a spanning tree rooted at the broadcast initiator. Each node in the tree forwards the broadcast message to all or subset of its fingers. The downside of broadcast is that its cost can become prohibitive especially in large P2P systems.

The RELOAD specification defines a simplistic TURN-specific service discovery mechanism [98]. The algorithm requires as input an estimate of the percentage of peers in the overlay that are capable of acting as TURN servers. In the algorithm, each TURN service provider stores a number of records equal to the reciprocal of the estimate in the overlay. Peers that need to find a TURN server generate lookups to random Node-IDs until they find a TURN relay. One downside of the algorithm is the estimate of the percentage of TURN servers that it requires. This estimate needs to be fairly accurate or otherwise the process of finding a TURN server becomes expensive (in terms of messages required and the latency of the TURN server discovery process).

The P2PSIP working group has also adopted a more generic service discovery algorithm. This algorithm, which we have specified in [137], is based on the Recursive Distributed Rendezvous (ReDiR) service discovery mechanism proposed in [181]. The benefits of ReDiR include, in addition to its generic nature, that it does not rely on the existence of super peers and does not require an estimate of the density of service providers. In addition, ReDiR does not rely on flooding or broadcast as the lookup mechanism and does not use fixed parameters such as the number of levels or partitions. The way ReDiR operates should also result in a balanced load of storing the service provider records. An overview of ReDiR is given in the subsection below.

ReDiR

ReDiR implements service discovery by building a tree structure of the peers that provide a given service. ReDiR trees are service-specific; each service has its own ReDiR tree. The tree structure is embedded tree node by tree node into the RELOAD overlay by using RELOAD Store and Fetch

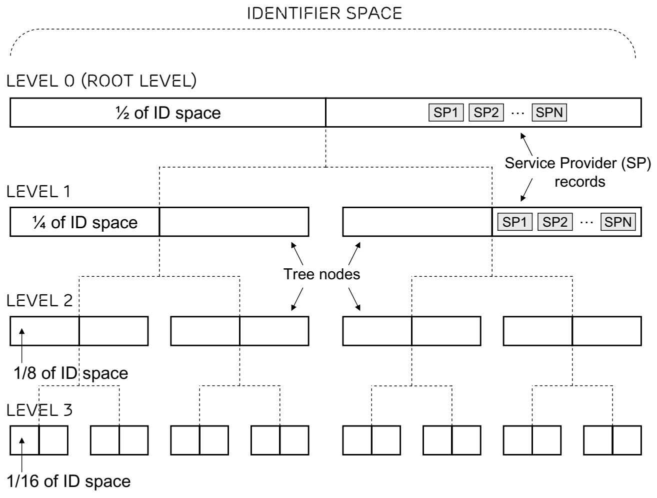


Figure 3.2. ReDiR tree

operations. Each tree node in the ReDiR tree contains one or more node-IDs of nodes providing a particular service. The ReDiR tree is illustrated in Figure 3.2. The tree has multiple levels. Each tree node belongs to a particular level. The root of the tree contains a single node at level 0. The child nodes of the root are at level 1, and so forth. The ReDiR tree has a branching factor b, which is 2 in the example shown in Figure 3.2. At every level i in the tree, there are at most b^i nodes. The nodes at every level are labeled from left to right, such that a pair *(i, j)* identifies the *j*th node from the left at level i. The tree is embedded into the DHT by storing the values of tree node *(i, j)* at key *H(namespace, i, j)*, where H is the SHA-1 hash function, and namespace is a string that identifies the service being provided. An example of a namespace is *"turn-server"*. Each ReDiR tree node is stored as a resource record in the RELOAD overlay. The RELOAD data model that these resource records use is *dictionary*, that is, a set of values indexed by a key. The dictionary keys are node-IDs of service providers. Each dictionary value (i.e., service provider record) contains among other things a RELOAD destination list that specifies how to reach the service provider in the overlay. The service provider record of a given service provider is typically stored in multiple tree nodes. The exact amount of copies stored depends on the density of existing service provider records in the tree.

Each level in the ReDiR tree spans the whole identifier space of the DHT.

At each level, the ID space is divided among the tree nodes. Each tree node is further divided into b intervals. When storing a service provider record (i.e., performing a service registration) with identifier k at a given level l, the record is stored in the interval within whose range k falls. A ReDiR service lookup is carried out by selecting a key (e.g., by generating a random key or using the node-ID of the node initiating the service lookup) and searching for the closest successor of the key from the ReDiR tree.

The primary reason why ReDiR embeds a tree structure in the overlay is to be able to provide service discovery in a scalable manner. As was discussed in Section 3.1.5, a simple service discovery mechanism relying on well-known keys and the regular put/get API that a DHT provides does not scale for popular services. The benefit of ReDiR's tree structure is that it distributes the load among peers that store information about service providers far better than a simple mechanism relying on well-known keys. This is because the part of the ReDiR tree in which a peer wishing to discover a service performs service lookups is determined by the search key. Since ReDiR makes it possible to use random search keys that are distributed uniformly in the identifier space, the result is, at least in theory, a balanced distribution of service lookups among the peers participating in the ReDiR tree. Another benefit of the use of the ReDiR tree and random search keys is that the service users are, at least in theory, distributed evenly among the service providers. The performance of ReDiR is analyzed in Publication VI.

3.2 Session Initiation Protocol

At the time of writing this dissertation, SIP [187] was the only application protocol usage for RELOAD that the P2PSIP WG of the IETF had formally adopted. SIP is the main IETF-specified session control protocol. SIP is being widely used for the establishment, modification, and termination of voice and video over IP, instant messaging, and presence sessions in the Internet. SIP has also been adopted as the call control protocol by the IP Multimedia Subsystem (IMS) [1].

SIP is an application-layer protocol that uses textual encoding. The textual encoding of SIP messages makes the protocol easier to debug for developers. SIP is based on the Hypertext Transfer Protocol (HTTP) [65]. It uses an HTTP-like request/response transaction model. Each transac-

Table 3.1. Examples of SIP methods

Method	Purpose
ACK	Acknowledges the reception of a final response to an INVITE request
BYE	Terminates a session
CANCEL	Cancels the previous request sent by a client
INFO	Carries session-related control information
INVITE	Establishes a session
MESSAGE	Allows the transfer of instant messages
NOTIFY	Notifies a SIP node about an event that has occurred
OPTIONS	Queries a server about its capabilities
PUBLISH	Publishes event state
REGISTER	Informs a proxy server about a binding between an AoR and a contact address
SUBSCRIBE	Requests asynchronous notifications of an event
UPDATE	Updates the parameters of a session

tion consists of a request and at least one response. SIP defines two types of responses: final and provisional. A final response always terminates a SIP transaction. In contrast, the purpose of a provisional response is merely to indicate progress; an example is an indication that the called terminal is ringing. Another key SIP concept is that of a dialog. A SIP dialog is a persistent peer-to-peer signaling relationship between two SIP user agents. As an example, an INVITE dialog is initiated by the SIP INVITE request, whose purpose is to establish a session. An INVITE dialog is terminated by a SIP BYE request, whose purpose is to tear down a session. SIP requests that are exchanged within a SIP dialog are referred to as mid-dialog requests. Further examples of SIP requests are listed in Table 3.1

SIP messages can contain one or more message bodies. For multimedia sessions, the most common body type is a session description, which describes the details of a multimedia session such as IP addresses, ports, and audio and video codecs to be used in the session. In SIP, multimedia sessions are described using the Session Description Protocol (SDP) [80].

The network elements that SIP defines include User Agents (UAs), registrars, proxies, and redirect servers. There are two types of UAs: User Agent Clients (UAC) and User Agent Servers (UAS). A UAC is a logical entity that creates a request and receives a response. In contrast, a UAS is a logical entity that receives a request and generates a response to it. The UAC and UAS roles last only for the duration of a single transaction.

A SIP registrar is a server that maintains information about the locations of SIP users. The location of the user is typically the IP address and port where the user is currently reachable. SIP UAs inform the registrar about locations of the users using SIP REGISTER requests. SIP proxies take care of routing of SIP messages. Proxies can be either stateless or stateful. A stateless proxy does not maintain any state information for a transaction. Instead, it simply forwards every request it receives downstream and every response it receives upstream. In contrast, a stateful proxy maintains client and server transaction state machines and thus takes care of among other things message retransmissions. The difference between a SIP proxy and a SIP redirect server is that a redirect server does not perform routing of SIP messages. Instead, it answers the requests it receives with a redirect response that directs the UAC to contact an alternative URI that points closer to the UAS.

Over the years, a considerable number of SIP extensions have been standardized. Some of the extensions that are used in this dissertation include SIP extensions for Instant Messaging (IM) [37], presence [185], event notification [184], and compression [33].

3.2.1 Signaling Compression

In contrast to SIP, traditional call control protocols such as the Integrated Services Digital Network (ISDN) User Part (ISUP) [60] use binary encoding for their signaling messages. The cost of the textual encoding that SIP uses is that SIP messages are typically multiple, even on the order of one hundred times larger than binary encoded messages. The large size of SIP messages is a challenge especially for narrowband links, where it leads to increased call setup delays.

The mechanism that the IETF has designed to address this problem is Signaling Compression (SigComp) [172]. SigComp is a solution to reduce the size of application protocol messages through data compression. In the protocol stack, SigComp represents a new layer between the application layer and the underlying transport; compressed application protocol messages are carried in the payload of SigComp messages. SigComp can make use of a variety of compression algorithms. This is possible since the decompression algorithm needed to decompress the payload is included in the header of the SigComp message in the form of bytecode. This bytecode is executed in the receiving end by running it in a virtual machine called the Universal Decompressor Virtual Machine (UDVM) that SigComp defines.

SIP extensions that can be used to signal that SigComp compression is desired for SIP messages are specified in RFC 3486 [33].

SigComp has been extended to improve its efficiency. RFC 3485 [68] defines a SIP and SDP static dictionary for SigComp that contains character sequences that occur commonly in SIP messages and their SDP bodies. RFC 3321 [81] defines further extensions, including dynamic compression and shared compression. The idea in these mechanisms is to utilize information from previously sent and received messages while compressing further application protocol messages. The author of this dissertation has studied the performance of SigComp and its extensions in [134]. In Publication VIII, the author uses SigComp and its extensions to reduce P2PSIP session setup delays.

3.3 NAT Traversal

Network Address Translation (NAT) is a method by which IP addresses are mapped from one address realm to another in order to provide transparent routing to end hosts [203]. There exist different types of NAT devices. Especially the way NATs assign IP addresses and port numbers to outgoing sessions differs across NATs [7]. In endpoint independent mapping, the NAT reuses the same port mapping for subsequent packets sent from the same internal IP address and port to any external IP address and port. In address dependent mapping, the NAT reuses the port mapping only for the same external IP address, regardless of the external port. Finally, in address and port dependent mapping, the NAT reuses the port mapping only for subsequent packets sent to the same external IP address and port.

Also the way NATs perform packet filtering for packets originating from external endpoints differs across NAT devices [7]. In endpoint-independent filtering, the NAT forwards packets to an internal IP address and port regardless of the external IP address and port source. In address-dependent filtering, the NAT forwards packets to an internal IP address and port only if the internal endpoint has previously sent packets to the external source IP address. Finally, in address and port dependent filtering, the NAT forwards packets from an external host only if the internal endpoint has previously sent packets to the external host's IP address and port.

There are many techniques for making applications such as real-time multimedia communication work across NAT devices. Application Level Gateways (ALGs) [203] allow transparent connectivity between applications running on two hosts in different address realms. ALGs may achieve this through different means, including interacting with NAT to set up state, using NAT state information, and modifying application specific payload of protocol messages. UNilateral Self-Address Fixing (UNSAF) [52] is a process in which an originating endpoint attempts to determine or fix the address and port by which it is known to another endpoint for instance to be able to use the address data in a protocol exchange, or to advertise a public address from which it will receive connections. Session Traversal Utilities for NAT (STUN) [186] is one way to perform the UNSAF function. Another example of an UNSAF mechanism is Teredo [94], which tunnels IPv6 over UDP/IPv4.

The NAT traversal solution that has been adopted by P2PSIP is Interactive Connectivity Establishment (ICE) [188]. ICE is a NAT traversal technique that has been designed by the IETF. In ICE, hosts that wish to establish a direct connection first gather a set of candidate addresses that can potentially be used for communication. There are three main types of candidate addresses. A host candidate is the transport address (i.e., IP address and port) of a local network interface. A server reflexive candidate is an address that has been assigned to a host by a NAT. A relayed candidate is a transport address that has been allocated for a host at a relay server for the purpose of relaying traffic destined to and originating from the host. ICE only uses a relay server as a last resort. The need for using a relay occurs when both of the hosts wishing to establish a connection are located behind the most restrictive types of NATs, that is, NATs using address and port dependent mapping and filtering behavior.

Having gathered ICE candidate addresses, the hosts run connectivity checks to test the connectivity between pairs of candidate addresses. The checks are done in priority order in such a way that host candidates have the highest and relayed candidates the lowest priority.

ICE makes use of the Session Traversal Utilities for NAT (STUN) [186] protocol and its extension, Traversal Using Relays around NAT (TURN) [140]. STUN is a client/server protocol. It defines two types of transactions: request/response transactions and indication transactions. The difference between the two types of transactions is that unlike request/response transactions, indication transactions consist only of a single message from one STUN agent (i.e., STUN client or server) to another; an indication

Table 3.2. STUN and TURN methods

Method	Purpose
Allocate	Create an allocation on a relay server
Binding	Used for learning server reflexive address, for connectivity checks, and for keepalives
ChannelBind	Used for creating a channel to a remote peer
CreatePermission	Install a permission allowing a remote peer to send data to the client
Data	Used for sending data from a server to a client
Refresh	Used for keeping an allocation alive
Send	Used for sending data from a client to a server

transaction does not include a response. The STUN RFC defines a single method called Binding. A Binding request is sent from a STUN client to a STUN server. ICE uses the Binding method to learn the server reflexive address of a STUN client and to implement connectivity checks and keepalives.

TURN is an extension to STUN. TURN allows hosts to control the operation of a relay server and to exchange packets with its peers using the relay. TURN extends STUN with a number of new methods, including the Allocate, Refresh, Send, Data, CreatePermission, and ChannelBind methods. The Allocate method is used for creating an allocation on a relay server. An allocation is a data structure on the server containing among other things the relayed address of the client. A TURN client can use the allocation for sending data to its peers. The client uses the Send method for sending data to the server. The Data method is used for sending data from the server to the client. The Refresh method is used by the TURN client for keeping the allocation alive. Using the CreatePermission method, a client can install a permission on the server for a given remote IP address. This permission allows a remote peer to send data to the client through the allocation. The ChannelBind method is used by the channel mode of TURN. The channel mode is an alternative to the use of Send and Data indications. In the channel mode, the ChannelBind method is used to request the creation of a channel between the client and a remote peer. Once the channel has been created, ChannelData messages are used instead of Send and Data indications to exchange data. The benefit of ChannelData messages is that due to the existence of the channel state on the relay server, they require less overhead (4 bytes) than Send and Data indications (which carry 36 bytes of overhead). The different

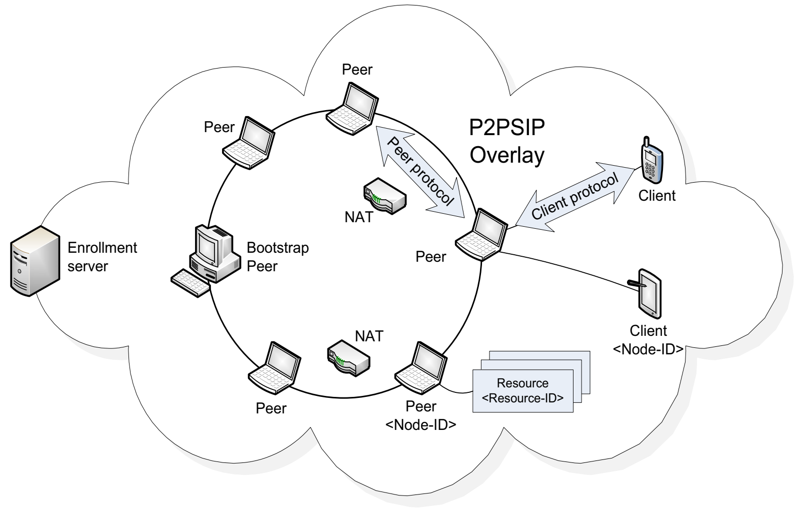


Figure 3.3. P2PSIP overlay

STUN and TURN methods are summarized in Table 3.2.

3.4 Peer-to-Peer Session Initiation Protocol

P2PSIP refers to a use case of SIP in which the SIP protocol is used in a P2P environment. The fundamental idea in P2PSIP is to replace the centralized proxy-registrar servers that SIP uses with a decentralized P2P overlay network. The first publications to propose a P2P architecture for SIP were [201, 25].

The architecture of a P2PSIP overlay is illustrated in Figure 3.3. The figure uses the terminology specified in [27]. In the figure, P2PSIP peers create an overlay network that in the figure uses a ring topology. The overlay has one or more bootstrap peers. Bootstrap peers are publicly reachable and thus capable of serving as the first points of contact in the overlay for peers wishing to join it. The protocol that peers use to maintain the overlay network, establish connections, and perform lookup and storage operations is called the peer protocol. In addition to peers, the P2PSIP overlay also contains nodes called clients. Clients are not part of the overlay ring typically because they have insufficient resources (e.g., bandwidth or battery power) to do so. Since they are not part of the overlay ring, clients do not provide routing and storage services to other nodes. Instead, clients use the services that the overlay network provides by connecting to a peer. The protocol that a client uses to communicate with a peer is called the client protocol. Both peers and clients are identified by node-IDs. The data that are stored in the overlay are referred to as resources. Each resource record is identified by a resource-ID. The overlay has a single centralized entity called the enrollment server, which is responsible for assigning certificates and node-IDs. Finally, some nodes may be located behind NAT devices that hinder the establishment of connections.

The P2PSIP peer protocol is being standardized in the P2PSIP working group of the IETF. In the early stages of P2PSIP standardization, there were several competing proposals for the peer protocol, including REsource LOcation And Discovery (RELOAD) [98], Peer-to-Peer Protocol (P2PP) [14], Address Settlement by Peer to Peer (ASP) [100], Extensible Peer Protocol (XPP) [143], Service Extensible P2P Peer Protocol (SEP) [102], and Distributed Transport Function in P2PSIP Using HIP for Multi-Hop Overlay Routing (HIP-HOP) [48]. Eventually, the working group converged into a single peer protocol which merged the RELOAD, ASP, and P2PP proposals. The resulting protocol inherited the name RELOAD. RELOAD is used as both the peer protocol and the client protocol in a P2PSIP overlay network.

P2PSIP can be used in several potential use cases [28]. Perhaps the best known of these is the one in which P2PSIP is used to provide a public P2P VoIP service. This use case is best exemplified by services such as Skype. In the open global P2P VoIP network use case, there is no single service provider responsible for the VoIP service. Instead, any user can join and leave the network freely and anyone can implement the software required to participate in the overlay network. This use case is made possible by the use of open standards. Another envisioned use case is to use P2PSIP-powered P2P presence to enable for instance instant content sharing between multimedia consumer electronics devices. Small organizations not having centralized IT could use P2PSIP to implement a decentralized serverless small-scale IP Private Branch Exchange (PBX) system. It has also been proposed that P2PSIP could be used for connecting together a farm of SIP proxies in a transparent way to pass resources between them with as little configuration as possible [215, 218]. A

P2PSIP system could also be used to construct ad-hoc communication systems in a zero-configuration fashion without the need to depend on Internet connectivity. Such a system could be used in isolated wireless ad-hoc networks [135] with no or limited connection to the Internet for instance in public events [165], emergencies [149], and battlefields. Huguenin et al. [171] specify how to use RELOAD for inter-domain SIP federation as a component of a solution called Verification Involving PSTN Reachability (VIPR) [12]. VIPR provides a fully distributed inter-domain routing for phone numbers. Yet another use case for P2PSIP is proposed in Publication VII, in which we describe how to use RELOAD to distribute M2M communication.

One challenge when analyzing the literature on real-time P2P communication is that the term P2PSIP is used to refer to a variety of different solutions. Especially in the early work on real-time P2P communication, the term P2PSIP referred to a system using SIP as the peer protocol, meaning that all the operations in the DHT were implemented using SIP in contrast to using RELOAD or one of its early versions. Also, some systems are referred to as P2PSIP although they use a proprietary P2P signaling protocol, do not use the SIP protocol for session control, or do not implement NAT traversal. In this dissertation, the term P2PSIP is used to refer to the system being specified in the P2PSIP working group of the IETF, that is, a system that uses RELOAD or one of its early versions (i.e., RELOAD, P2PP, or ASP) as the peer protocol, ICE as the NAT traversal solution, Chord as the DHT algorithm, and SIP as the session control protocol. Other SIP-based P2P solutions are referred to as *distributed SIP*.

3.4.1 RELOAD

As was discussed in the previous subsection, the P2PSIP working group has adopted RELOAD [98] as the P2PSIP peer protocol. RELOAD is a P2P signaling protocol that provides a generic overlay network service for application protocols. RELOAD supports operations such as data lookup, data storage, and message routing. The most important features of RELOAD include its security framework, usage model, NAT traversal, high performance routing, and pluggable overlay algorithms.

In RELOAD, every node has one or more public key certificates assigned by a central server [98]. Besides certificates, the central server also assigns node-IDs. RELOAD messages are sent over TLS [54] and DTLS [179] protocols. The certificates assigned by the central server are utilized in TLS and DTLS handshake phases. Further, every RELOAD message a node sends and every object a node stores in the overlay is signed with the node's private key. For storing the certificates in the overlay, RELOAD defines a certificate store usage, which makes it possible to fetch certificates from the overlay. The certificate store usage avoids the need to include certificates in each RELOAD message.

Although RELOAD is designed to support a P2PSIP overlay network, also other applications can utilize it by defining new usages. This is enabled by RELOAD's usage model. Each new RELOAD usage needs to specify among other things the content and format of the data structures it stores in the overlay. Several RELOAD usages have been proposed. The SIP usage for RELOAD, which will be discussed in more detail in Section 3.4.3, is specified in [99]. Other usages include the CoAP usage for RELOAD that we have proposed in [103], the service discovery usage that we have proposed in [137], RELOAD usage for distributed conference control [116], SNMP usage for RELOAD [167], usage for shared resources [117], and usage for PSTN verification [171].

RELOAD uses ICE [188] (see Section 3.3) to establish connections across NATs. When two nodes wish to set up a direct connection, they first exchange ICE candidate addresses via the overlay using a RELOAD message called Attach. ICE will then run connectivity checks between pairs of candidate addresses to discover a working path between the nodes.

Although RELOAD specifies Chord as the mandatory to implement DHT, it can also be extended to support other overlay algorithms. This is enabled by RELOAD's topology plugin model. A topology plugin is responsible for implementing a specific overlay algorithm. At the time of writing this dissertation, Chord was the only topology plugin that had been specified for RELOAD.

The most important RELOAD messages are summarized in Table 3.3. As was already discussed above, the RELOAD Attach message is used to establish direct connections for RELOAD signaling between two peers. Attach messages carry ICE candidate addresses in their payload. The purpose of the AppAttach is similar to Attach with the difference that AppAttach is used to establish direct connections for application-layer protocols such as SIP. The Join and Leave messages are used to join and leave the overlay, respectively. The Fetch and Store messages are used to retrieve data from and store data in the overlay. The Update message is used to support DHT maintenance operations, such as periodically

Table 3.3. Examples of RELOAD messages

Message	Purpose
AppAttach	Establish a direct connection with another node for application layer messages
Attach	Establish a direct connection with another node for RELOAD signaling
Fetch	Used to retrieve a data element stored under a given resource-ID
Join	Used by a new peer to join the overlay
Leave	Used to indicate that a node is exiting the overlay
Ping	Used for instance for populating the finger table
Store	Store data in the overlay
Update	Used for overlay maintenance

exchanging neighborhood information among neighboring peers. Finally, the Ping message is used to test connectivity and for instance to find new entries to the Chord finger table.

3.4.2 P2PP

P2PP [14] was one of the predecessors of RELOAD during the P2PSIP peer protocol standardization process. The reason why a separate subsection is devoted to P2PP is that the author's first P2PSIP implementation and thus also the first publications of this dissertation were using the P2PP protocol. The main reason why the author chose to base his early P2PSIP implementation on P2PP back in 2007 was that when the implementation was started, P2PP was the only peer protocol proposal that was considered mature enough to be implementable.

RELOAD and P2PP are very similar to each other, which is not surprising considering that P2PP was merged with RELOAD during the standardization process in the P2PSIP working group. Their common properties include for instance the ability to support also other application usages than P2PSIP, the use of ICE for NAT traversal, use of a secure transport, the ability to support multiple different overlay algorithms, messages using binary encoding, support for both peers and clients, reliance on central enrollment and authentication servers, and the use of certificates and public/private key cryptography.

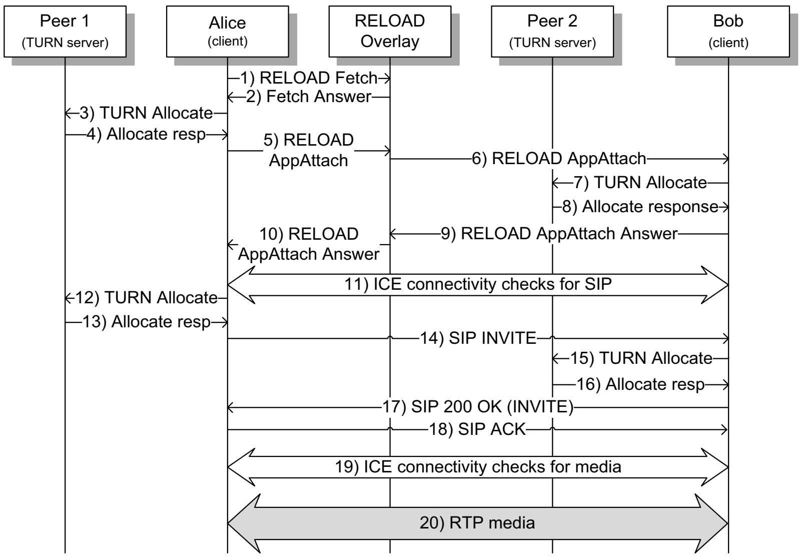


Figure 3.4. P2PSIP session setup

3.4.3 SIP Usage for RELOAD

The SIP Usage for RELOAD specification [99] defines how to use a RELOAD overlay network to provide the functionality of SIP proxy and registrar servers. The main operations defined by the usage are SIP registration and session setup.

To perform a registration operation, a SIP UA uses RELOAD to store a mapping from a SHA-1 hash of its SIP Address of Record (AoR) to its node-ID in the overlay. The process of establishing a SIP session is illustrated in Figure 3.4. In the figure, Alice wishes to set up a VoIP session with Bob. Both Alice and Bob are acting as clients in the P2PSIP overlay. To set up the session, Alice first computes a SHA-1 hash over Bob's SIP AoR. Next, Alice fetches Bob's node-ID from the RELOAD overlay in step 1 using a RELOAD Fetch request destined to the SHA-1 hash. The Fetch answer containing Bob's node-ID is returned in step 2. In steps 3-4, Alice gathers ICE candidates for the purpose of establishing a direct connection for SIP signaling between her and Bob. ICE candidate gathering is carried out by sending a TURN Allocate request to Peer 1 who is acting as a TURN server for Alice. In steps 5-6, Alice sends her ICE candidate addresses to Bob in a RELOAD AppAttach request that is destined to Bob's node-ID. In steps 7-8, Bob gathers his ICE candidate addresses and returns them to Alice in steps 9-10 inside an AppAttach answer. In step 11, Alice and Bob run ICE connectivity checks to establish a direct connection between them for SIP signaling across NATs. Once ICE has established the direct connection, Alice next gathers ICE candidate addresses for the RTP media stream in steps 12-13. These candidates are included in the SDP offer that is sent to Bob in a SIP INVITE request in step 14. After Bob has received the SIP INVITE, he gathers his ICE candidates for the RTP stream in steps 15-16. Bob returns these candidates to Alice in an SDP answer included in the payload of a SIP 200 OK final response that is sent to Alice in step 17. In step 18, Alice acknowledges the reception of the 200 OK response using a SIP ACK message. In step 19, Alice and Bob execute ICE connectivity checks for the RTP media stream. When the connectivity checks have finished, RTP packets start flowing between Alice and Bob in step 20.

3.4.4 P2PSIP Delays

Delays associated with operations such as user registration, lookup, session setup, and user de-registration are inherently longer in a P2PSIP system compared to client/server SIP, as we have shown in Publication II. The major reason for this is the difference in the cost of routing a message from its source to its destination: in a DHT, this cost is $O(logN)$ hops, whereas in a client/server system, it is $O(1)$. From the viewpoint of the session setup delay, a further reason is the two ICE negotiations that P2PSIP performs when setting up a session, as we have shown in Publication V. This subsection will summarize other work on analyzing delays in P2P systems.

Meyer and Portmann [152] present the results of using OpenDHT [181] as a distributed lookup service for SIP. OpenDHT is a free, public DHT service that runs in PlanetLab. In the evaluation, no P2PSIP protocols were used and the SIP nodes did not take part in the overlay but instead used OpenDHT via eXtensible Markup Language (XML) [24] based Remote Procedure Calls (RPCs). A NAT-free environment was assumed. The results indicate that the call setup delays of even a large-scale distributed SIP deployment are acceptable.

Tian et al. [208] compare the performance of DHTs in a distributed SIP use case through OverSim [16] simulations. The compared DHTs include Kademlia, Chord, and Bamboo. No P2PSIP peer protocol is used and a NAT-free environment is assumed. No details are revealed on the network setup used in the simulations. The results indicate that while Kademlia and Bamboo have advantages in performance compared to Chord, the total amount of data stored in their routing tables is larger and their maintenance generates more traffic. Kademlia and Bamboo were also observed to consume more CPU power, which combined with the higher bandwidth usage, may be problematic for mobile devices.

Harjula et al. [83] compare the performance of the Chord and Kademlia DHTs in a distributed SIP use case through simulations. The simulated overlays were not experiencing any churn. A NAT-free environment was assumed. The results indicate that Kademlia can achieve a lower lookup hop count than Chord. However, Chord has better scalability than Kademlia. The paper concludes that Kademlia is a good choice in small and medium size overlays where lookup cost needs to be minimized. However, Chord becomes a better choice in larger overlays.

In [82], the performance of a mobile hierarchical distributed SIP system in which only super nodes participate in the overlay network is analyzed. The results are obtained from an analytical model. They indicate that distributed SIP and client/server SIP achieve similar performance at least in a cellular network due to the dominating role of the delay associated with the first wireless hop. However, when it comes to message hop counts, distributed SIP has clearly higher costs.

In [97], the performance of a distributed SIP system that uses a variant of the CAN DHT algorithm is evaluated through small-scale measurements in which the DHT contained only 2-5 peers. A NAT-free environment was assumed. The peer protocol used in the evaluation was XPP. The conclusion of the evaluation is that distributed SIP is a feasible solution in small office or home environment since the cost in terms of increased delay is moderate at least in a small network.

Zheng et al. [219] study call setup delays in a distributed SIP system. The results were obtained using an analytical model of a Chord network where there are no NATs and firewalls. No P2PSIP protocols were used. Further, iterative routing rather than the recursive routing model that P2PSIP has adopted, was used. The results indicate that session setup delay of distributed SIP has a high variance and that lookup latency is the dominant component, taking 60-80% of the total delay. However, these results are not applicable to P2PSIP due to the differences in the protocol and routing mode used and the absence of ICE and NATs in the evaluation.

The lookup delays in a Chord DHT have naturally been studied also in non-P2PSIP use cases. Binzenhofer and Tran-Gia [22] study delays in a

Chord-based file sharing system. The results were obtained from an analytical model. The main findings include that the search delay increases rapidly as a function of the network size when the size of a small Chord overlay increases. However, despite of the rapid initial increase, the delay still stays moderate even in very large peer populations. Also the 95th, 99th, and 99.99th percentiles of the search delay are analyzed. The results indicate that the percentiles can be on a significantly higher level than the average. The high delay variation makes it more difficult to guarantee Service Level Agreements (SLAs). The path length of Chord lookup operations is investigated in [205] through simulations. The results indicate that the mean and 99th percentile path length increases logarithmically as a function of network size and that the mean path length is roughly $\frac{1}{2}log_2N$.

A common shortcoming for all the research above is that a NAT-free environment is assumed. This greatly reduces the accuracy of the results since multiple components of P2PSIP delays are omitted, including the RELOAD Attach and AppAttach transactions, ICE candidate gathering, and ICE connectivity checks for signaling and media. Further limitations include one or more of the following: (i) some other DHT algorithm than Chord, which is the mandatory-to-implement DHT for P2PSIP, is used, (ii) some other routing mode than symmetric recursive routing, which has been adopted by RELOAD, is used, (iii) the results have been obtained through the use of an analytical model or simulations rather than real-life measurements in a DHT running in the global Internet, or (iv) neither the RELOAD protocol nor its predecessors are used.

3.4.5 Reducing SIP and P2PSIP Delays

In Publication V, we show that P2PSIP session setup delays can be unacceptably high, especially when the hosts wishing to establish a session are located behind P2P-unfriendly NATs. Therefore, mechanisms to bring the session setup delay down to acceptable levels are necessary. Such mechanisms can target different components of the P2PSIP session setup delay, including (1) the delays caused by NAT traversal procedures, (2) delays caused by SIP related procedures, and (3) delays caused by DHT routing and lookups.

Reducing Delays Related to NAT Traversal

As the author of this dissertation shows in Publication VIII, the most efficient way to reduce the P2PSIP session setup delay is to remove one of the two ICE negotiations that P2PSIP session setup includes. In [35], we describe how to achieve this by using the Host Identity Protocol based Overlay Network Environment (HIP BONE) [36] architecture together with the HIP BONE instance specification for RELOAD that we have defined in [113]. The main idea in this approach is to use the Host Identity Protocol (HIP) [157] to perform connection management for P2PSIP. The benefit of the approach is that it allows the multiplexing of SIP and RTP over a single connection established using HIP, thus removing one of the two ICE negotiations performed when setting up P2PSIP sessions. The downsides of the approach include the implementation cost required to integrate P2PSIP and HIP and the fact that the approach is not compatible with RELOAD.

Wacker et al. [213] propose a NAT traversal mechanism for P2P networks. Another similar mechanism is proposed in [174]. These mechanisms are alternatives to ICE. They work as follows: in the first step, the peers wishing to establish a direct connection determine the types of NATs behind which they are located. Next, the peers exchange information about their NAT types across the overlay. Finally, the peers choose the appropriate means for establishing a connection based on the combination of their NAT types. As an example, if both hosts are located behind the most restrictive types of NATs, the peers will establish the connection via a relay. The benefit of this approach is that it reduces the connection establishment delay compared to ICE since it does not need to check connectivity between multiple different candidate address types. However, the approach has a significant shortcoming; the process of a peer trying to determine the type of the NAT behind which it is located is known to be error-prone [188]. This is especially due to the fact that some NATs have non-deterministic behavior, that is, the same NAT can exhibit different behavior under different conditions [7]. Further, the approach is not robust in complex NAT topologies where a peer can be behind multiple levels of NATs.

Another way to reduce NAT traversal related delays is to modify the NAT devices. One example of such an approach is the use of Domain Name System (DNS) extensions to NATs [204]. This approach is based on the use of DNS Application Level Gateways (DNS ALGs), which translate ⟨*Fully Qualified Domain Name (FQDN), private address*⟩ mappings in DNS payloads into ⟨*FQDN, external address*⟩ mappings and vice versa using state information available on NATs. As an example, let us assume that an external host X wishes to establish a session with a private host A located behind a NAT in a domain *private.net*. As the first step, X performs a DNS lookup query for the FQDN of host A, which is *A.private.net*. The DNS query is eventually routed to the DNS server of the domain *private.net*. On its way, the DNS query transits the NAT of the domain *A.private.net*. This NAT has DNS ALG functionality. The DNS server of the domain *private.net* replies to the DNS query with the private address of host A. This reply will transit the NAT on its way towards X. Upon receiving the reply, the DNS ALG of the NAT modifies the payload of the DNS reply by replacing the private IP address in the payload with an external address. Further, the DNS ALG also requests the NAT to setup a temporary binding for host A with the external address. Once X receives the DNS reply, it will initiate a session to A's external assigned address. When it receives the datagrams, the NAT translates the external address to host A's private address. The benefit of the DNS ALG approach is that it removes the need for technologies such as ICE. The downside is naturally that since this approach requires changes to existing NAT devices, it cannot solve the problems in the present day networks.

Reducing the SIP Session Setup Delay

Another way to bring P2PSIP session setup delay down is to focus on optimizing the components of the delay associated with SIP signaling.

Baldi et al. [11] propose an alternative to ICE called Address List Extension (ALEX). ALEX is a non-standard SIP extension that provides dual stack, multihoming, and NAT traversal support for signaling and media flows. ALEX attempts to reduce SIP delays by setting up a direct communication channel for SIP between the communicating SIP UAs for mid-dialog and non-dialog-related SIP messages, and by making use of the information obtained during the ALEX negotiation for the SIP channel when negotiating the media channels. The benefit of ALEX is that, according to the results presented in [11], it can reduce SIP session setup delays on the average by roughly 23%. The downside of ALEX is that it is a non-standard approach. Further, ALEX is not applicable to P2PSIP since it assumes the use of proxy servers for the initial SIP signaling.

As was already discussed in Section 3.2.1, SIP session setup delays can also be reduced by using SigComp compression. This approach reduces the delay of transmitting SIP messages especially over narrowband links. In Section 4.6, the work of the author of this dissertation on applying SigComp to P2PSIP will be summarized.

Reducing DHT Lookup Delays

P2PSIP session setup delays can also be reduced by reducing the cost of DHT lookup operations. The cost (i.e., path length) of a DHT lookup operation is $O(logN)$ hops. A straightforward way to attack the delay of lookup operations is to shorten the path length. One-hop DHTs [77], as their name suggests, bring the path length down to *O(1)*. This is possible since the routing table of every peer in a one-hop DHT contains every other peer in the overlay. The high cost of maintaining a complete routing table makes one-hop DHTs primarily attractive in small-scale or low-churn systems [130]. Further, in environments with NATs, one-hop DHTs need to maintain a full mesh of connections between every peer in the overlay.

Another technique for reducing lookup latency is to choose geographically close nodes as routing table entries. This technique is know as Proximity Neighbor Selection (PNS) [50]. The cost of using PNS is that the DHT design must include an algorithm to search for nearby nodes. Such algorithms typically rely on RTT measurements to select nearby routing table entries. These measurements create additional traffic in the overlay. As an example, it is shown in [180], that a 42% decrease in lookup latency requires a 40% increase in the traffic load of the DHT.

Li et al. [131] apply a proximity-based approach to reduce DHT lookup delays in a distributed SIP system. The proposed locality-aware distributed SIP system uses the Pastry DHT and skip graphs [5]. In the solution, geographical location information is embedded in node-IDs and resource-IDs in the form a a locality prefix. The downside of this approach is that it does not work with the centralized security model of RELOAD and also results in non-uniformly distributed node-IDs, which results in performance degradation. Performance evaluation of the proposed locality-aware system is carried out using simulations assuming a NAT-free environment. The results are reported using hop counts; no delay values are reported. The results confirm that proximity awareness can reduce lookup delays.

Also parallel lookups for a given key can reduce the lookup delay. Cheng et al. [45] propose the use of two parallel lookups in a Bi-Chord [101] ring that maintains two finger tables, one in the clockwise direction and another one in the counter-clockwise direction. The mechanism works by sending one lookup in the clockwise direction and one in the counter-clockwise direction. The mechanism can reduce the lookup path length from $\frac{1}{2}log_2(N)$ to roughly $\frac{1}{2}log_2(\frac{N}{2})$. However, this improvement comes at the cost of the need to maintain two finger tables, which increases the maintenance cost of the overlay considerably.

Other improvements proposed in [222] include the use of the semi-recursive routing mode and cache entry records. A cache entry record is a local database that a peer uses to record its communication history. The database contains the node-IDs, IP addresses, and ports of other peers with whom the peer has established sessions in the past. The downside of semi-recursive routing is that it only works when there are no NATs present in the overlay network. The same is true for the cache entry record; it can only achieve the suggested $O(logN)$ optimization in a NAT-free overlay.

Yet another way to reduce the cost of lookups in structured overlay networks is to use caching of resource records in the overlay. Darlagiannis et al. [53] propose a mechanism that caches popular resource records in intermediate peers along the paths towards the responsible peers. The mechanism can cut the routing cost of lookup operations by 50% compared to the original cost. One challenge associated with caching is choosing a maximum lifetime for cached information. In addition, caching can only reduce delays if the information being cached is popular. This renders caching less effective in the P2PSIP use case, since contact records of users have very uniform popularity when compared to for instance a file sharing use case.

Also the choice of the routing mode impacts the DHT lookup delay. In iterative routing, a peer receiving a message does not forward the message to a peer closer to the target key. Instead, the peer replies to the initiator with a message carrying the identifier of the closer peer. In recursive routing, a peer receiving a lookup request always forwards the request to a peer that is closer to the search key. This continues until the lookup reaches the peer responsible for the search key. The response from the responsible peer is routed back along the reverse of the path that the request followed. Forward-only routing is similar to the recursive routing mode with the key difference that the response is not routed back along the reverse of the path that the request followed. Instead, the response is routed across the overlay to the identifier of the initiator of the request. Finally, in semi-recursive routing, the request is routed recursively but the response is sent directly back to the initiator over a single routing hop. The analysis in [26] shows that the semi-recursive routing mode has the lowest cost, whereas the recursive routing mode typically used by DHTs has the second lowest cost. Thus, the cost of DHT lookups can be reduced by adopting the semi-recursive mode. However, as was already discussed earlier in this subsection, semi-recursive routing works only if there are no NATs between the peers initiating and terminating the request.

DHTs with hierarchical structure [67] can also reduce lookup delays. Ou et al. [163], propose the use of a hierarchical overlay network consisting of multiple layers of sub-overlays. In the system, the lowest level sub-overlay has the most peers while the top level sub-overlay has the least peers. Peers in higher level sub-overlays duplicate all the resource records stored by peers in lower-level sub-overlays. The results of the theoretical performance analysis in [163] suggest that the approach can reduce lookup latencies although the path length is higher than for a flat DHT. However, the lower lookup latency comes at the cost of higher complexity of the system and high load for the peers in the higher-level sub-overlays. Koskela et al. [123] compare the performance of hierarchical and flat architectures for the Chord and Kademlia DHTs in the context of a P2P community management service. The results indicate that a hierarchical structure can achieve lower hop counts, and thus also lower delays, than a flat architecture.

3.4.6 Mobile P2PSIP

Mobile networks and devices pose special challenges for P2P applications, including consumption of resources (e.g., battery capacity [110], CPU [162], bandwidth [111], and memory), device mobility, the churn that device mobility creates [146], and the heterogeneity of the device population [84].

Matuszewski and Kokkonen present a distributed SIP architecture for mobile communities in [150, 120]. The architecture uses OpenDHT [181] as the overlay network instead of Chord and RELOAD. In the architecture, mobile devices do not participate in the overlay network but instead act as clients using XML-RPC over HTTP as the client protocol. ICE is not used. Instead, the mobile devices always use a relay server to send and receive SIP signaling and media. The architecture is evaluated through measurements in the OpenDHT overlay network. The results of the measurements suggest that registration and call setup delays do not impose any significant restrictions on the implementation of a serverless mobile

VoIP service.

A mobile middleware system called Plug-and-Play Application Platform (PnPAP) is presented in [86, 82, 92, 108, 216]. In [82], a version of PnPAP that uses distributed SIP for interconnection is introduced. The interconnection overlay uses a hierarchical architecture where mobile nodes connect to super peers. A NAT-free network is assumed. The performance of the system is analyzed using an analytical model. The results suggest that a hierarchical architecture has lower lookup and registration latencies than a fully distributed architecture in which also mobile devices participate in the overlay.

Another hierarchical distributed SIP architecture for mobile environments is proposed in [145]. The idea in this architecture is to improve the performance of distributed SIP by dividing nodes to different overlay networks based on their mobility behaviors. These separate overlay network domains are connected using an interconnection overlay. Each of the interconnected domains has a super peer which is responsible for participating in the interconnection overlay. Traffic destined to or originating from another domain needs to be routed via the super peer. No performance evaluation is carried out, but the suggested benefit of the architecture is that it avoids the impact of peers with high churn on the whole system by isolating such peers in their own sub-overlay network. The cost of the approach is the need to maintain two levels of overlay networks and the need for the super peers to route the traffic of all peers in their domain.

Cheng et al. [44] propose the use of an unstructured overlay algorithm called Unstructured P2P SIP (UP2P SIP) for connecting SIP nodes in a mobile environment. The use of an unstructured overlay is in contrast to RELOAD, which uses a structured overlay (i.e., Chord) as the mandatory-to-implement overlay algorithm. UP2P SIP uses flooding as the search mechanism. In the UP2P SIP architecture, users form P2P links with their friends instead of random nodes. The UP2P SIP architecture is evaluated through simulations in a stable overlay. The simulation results suggest that the UP2P SIP architecture outperforms a DHT-based system in terms of call setup delay and maintenance cost. However, the performance of the system is not evaluated in high-churn and large-scale scenarios. The downside of the use of an unstructured overlay is that unlike a structured overlay, an unstructured system cannot provide performance guarantees due to the fact that the system cannot avoid false negatives [26], as was discussed in Section 3.1.3.

The feasibility of a communication-oriented mobile P2P system is evaluated in [162] using a system called Mobile Peer-to-Peer Protocol (M2PP). M2PP uses Kademlia as the DHT algorithm and P2PP as the peer protocol. The performance of M2PP is evaluated through simulations that assume a NAT-free environment. No security features such as encryption and signatures were used. The results indicate that the CPU load and bandwidth usage of the system are acceptable for mobile devices. In [107], the same M2PP prototype is used to study hop count and retransmission count values in a mobile P2PSIP system. However, the delays of P2P operations were not studied.

The energy consumption on a mobile DHT client is studied in a BitTorrent system in [110]. The results indicate that the battery of a mobile device is depleted in only a couple of hours. The studied use case is mobile P2P file sharing; mobile P2P communication was not studied.

The battery life of mobile peers in Kademlia-based P2PSIP system is studied in [109]. However, the results are based on calculations and not simulations or measurements. Another study on the performance of a Kademlia-based mobile P2PSIP system is presented in [111]. Also these results are based on an analytical model. Kassinen et al. [105] study the battery life in a Kademlia-based P2P overlay network. The battery life is evaluated both in 3G and WLAN access networks. In the measurements, mobile peers join a simulated overlay network. The results indicate that when using 3G access, the battery of a mobile peer is drained in three hours. When using WLAN access, the battery is drained on the average in 8 hours.

A common property for the mobile P2P research summarized above is that the RELOAD and ICE protocols are not used. Further, either (i) the Chord DHT, which has been chosen as mandatory to implement by the P2PSIP working group, is not used, (ii) the use case is not P2P real-time communication, (iii) the impact of security features such as public key cryptography are not analyzed, (iv) the studies are based on analytical models or simulations rather than measurements on real mobile networks, (v) mobile devices are not participating in the overlay, or (vi) a full range of performance metrics (delays, message sizes, and the use of CPU, memory, bandwidth, battery) is not studied. Section 4.9 will summarize our work on the performance of mobile P2PSIP.

3.4.7 Other P2PSIP Research

The subsections above gave an overview of P2PSIP research in areas related to the publications of this dissertation, including DHT maintenance, delay analysis, mobile P2PSIP, self-tuning, NAT traversal, service discovery, and optimization of delays. This subsection will summarize P2PSIP research in other areas.

Interconnectivity to other communication networks such as PSTN is important to any new real-time person-to-person communication service. Interconnectivity between IMS and P2PSIP has been studied in [142, 88, 144, 221]. The solution for interconnection presented in [88, 221] is to use a gateway that acts as a peer in the P2PSIP network and as an application server on the IMS side. In [142, 144], the approach is to rely on proxy peers that can exchange SIP messages with public domains and provide this function as a service to the P2PSIP overlay. In addition, relay agents are used to relay media streams to endpoints in other domains.

A technology evolution analysis framework is built for mobile peer-to-peer communications in [90]. Three evolution paths are studied: Internet-driven, telecom-driven and proprietary. These paths are represented by P2PSIP, IMS, and Skype. The paper concludes that P2PSIP can serve as an alternative to existing networks in situations where lower costs are desired or when existing networks are not available. The main obstacle for P2PSIP is its technical immaturity. This is a problem that the publications making up this dissertation attempt to address.

RELOAD-based distributed conference control solution is presented in [118]. The idea is to define a new RELOAD usage for separating the conference URI from any specific conference focus entity and instead map it to multiple RELOAD peers. This usage allows the creation of multimedia conferences without dedicated server infrastructure. Another video conferencing system based on distributed SIP is presented in [57]. This system is called Video Conference Network Foundation (VCNF). VCNF uses Chord to store and retrieve user and group information and SIP to establish media sessions. Klauck and Kirsche [115] present yet another P2P videoconferencing system called BRAndenburg VIdeo conferencing System (BRAVIS) that uses P2PP for rendezvous and SIP for session establishment.

Buford and Kolberg [29] have proposed a RELOAD usage for Application Layer Multicast (ALM). In ALM, which is also known as end system multicast [93], end hosts in an overlay network implement all multicast functionality, including membership management and packet replication. ALM requires no infrastructure support. Therefore, it is in contrast with IP level multicast that implements multicast in the network level and thus requires support in network routers. The RELOAD usage for ALM defines new RELOAD data types, message types, and new ALM topology plugins based on Scribe [40] and P2PCast [160]. These mechanisms enable ALM in RELOAD-based overlays.

Koskela [121] presents preliminary work on a HIP-based distributed SIP system. The system is used for experimenting with security mechanisms for distributed real-time communication. The same authors present a related HIP-based SPAM prevention system for distributed SIP in [122]. The use of HIP for P2P connections enables strong authentication and confidentiality, mobility, multihoming, and NAT traversal. The performance of the system is evaluated through measurements. The results indicate that the initial HIP handshake, which is known as the Base Exchange (BEX), introduces a considerable additional delay component to call setup times. However, the impact of the IP Security (IPSec) encryption that HIP uses is minor when sending data over the connection established using HIP. Zheng and Oleshchuk [223] propose another architecture for improving the security of distributed SIP. The architecture is based on the use of centralized secure proxy servers called Chord Secure Proxies (CSPs). CSPs provide security services such as data confidentiality, integrity, and availability during P2PSIP session setup. A CSP is a provisioned, secure and trusted application server that acts as an intermediary between two peers wishing to set up a session. The conclusion of the paper is that a CSP based system provides better security than the current P2PSIP system. The main cost of the CSP approach is the need for the operator of the overlay to provide and operate the centralized CSP servers.

Also the topic of event subscriptions and notifications in a P2PSIP overlay network has generated research interest. A P2PSIP event notification architecture that replaces traditional SIP presence servers with a P2P overlay network is proposed in [164]. The proposed architecture uses ALM to distribute event notifications within the overlay. Wang et al. [214] propose another event notification mechanism for P2PSIP. In this proposal, direct subscribe/notify relationships between peers are used in contrast to ALM. Le et al. [126] propose a solution where a P2PSIP peer acting as a presence server uses routing cost information from an Application-Layer Traffic Optimization (ALTO) [195] server to build a minimum spanning tree among watchers (i.e., peers that have subscribed to a particular event) with itself as the root. Event notifications are distributed among the watchers using the minimum spanning tree. The performance of the solution is analyzed through simulations. The results indicate that the solution reduces the load of the presence server, keeps delays at acceptable levels, and reduces the amount of data traffic.

3.5 Machine-to-Machine Communication

The term Machine-to-Machine communication is a generalization of the term telemetry, which is about performing remote measurements. M2M communication implies autonomic communication between non-human-operated machines [211]. It is used for instance to transmit fuel consumption data from trucks, weather related information from a weather station, or to track items in a vending machine. In telemetry applications, remote access to information on devices is most often achieved through cellular radio networks such as Global System for Mobile Communications (GSM), 3G, or Long-Term Evolution (LTE).

One specific form of M2M communication is wireless sensor networking. A Wireless Sensor Network (WSN) is formed by wireless sensors that co-operatively monitor a physical environment [212]. Using their wireless radios, the sensor nodes communicate not only with each other but also with a base station. Not all of the sensors in a WSN may be able to directly communicate with the base station. Therefore, the sensors also help each other to relay information towards the base station. This allows them to disseminate their sensor data to remote processing, visualization, analysis and storage systems. WSNs are used among other things for wild fire tracking, animal observation, agriculture management, and industrial monitoring to name only a few examples. WSNs use short-range wireless radio technologies for communication. During the recent years, these technologies have been moving away from proprietary to standards-based solutions. The primary radio standard for WSNs is IEEE 802.15.4 [95], which specifies the physical layer and media access control for WSNs. IEEE 802.15.4 based WSNs are capable of carrying IPv6 packets by applying the encapsulation and header compression mechanisms defined by the IPv6 over Low power Wireless Personal Area Networks (6LoWPAN) working group of the IETF [155].

A full M2M system consists of not only the sensors, actuators, and communications equipment in the field. Other elements of the system include centralized management applications, resource directories, databases, and software for analyzing the input and making decisions based on the analysis [125]. Such systems are sometimes referred to as M2M service enablement systems or M2M middleware.

In Publication VII, we investigate Wide Area Sensor and Actuator Networks (WASANs). WASANs refer to autonomous wide area M2M communication systems where the interconnected sensors and actuators cooperate with each other to replace the need for implementing central management, processing, and application logic in servers and data centers. Since such systems are deployed in the wide area, they rely on cellular connectivity. In Publication VII, we propose how to apply a RELOAD-based P2P architecture for WASANs. In addition, in [136], the author of this dissertation applies Self-Organizing Maps (SOMs) [119] to the problem of data classification in WASANs.

The expansion of IP connectivity to WSNs and other forms of M2M communication is only the beginning. The Internet is also extending into common everyday objects ranging from consumer electronics devices and kitchen appliances to toys and even clothing. This vision is known as the Internet of Things, a concept that was put forward by Kevin Ashton in 2002 [148]. The Internet of Things goes beyond the data and services that can be provided by an isolated WSN or embedded M2M system. Rather, it provides access to connected smart objects that in the not-so-distant future will become ubiquitous in our surroundings. In the Internet of Things, these smart objects become addressable, self-configuring, and seamlessly integrated to information networks.

A vision closely related to the Internet of Things is the Web of Things. In the Web of Things, smart objects in the physical world become integrated not only with computer networks but also with the World Wide Web. In this architecture, real world objects become RESTful resources and thus integratable with the existing Web [75]. A central protocol for the Web of Things is the Constrained Application Protocol (CoAP). REST and CoAP will be discussed in more detail in the subsections below.

3.5.1 Representational State Transfer

Representational State Transfer (REST) is an architectural style for designing web applications. The term REST was introduced and defined by Roy Fielding in 2000 in his doctoral dissertation [66]. REST provides a set of architectural principles that focus on a system's resources, including how resource states are addressed and transferred over HTTP. In a RESTful service, resources are identified by a Uniform Resource Locator (URL). REST defines four operations, which are referred to by the acronym CRUD. This acronym comes from the words Create, Read, Update, and Delete. These four operations map to HTTP's POST, GET, PUT, and DELETE methods. To create a resource on the server, clients use the POST method. GET is used to retrieve a resource, whereas PUT is used to change the state of a resource or update it. Finally, DELETE is used to remove or delete a resource.

The major benefit of REST is that it is stateless; a client includes within the HTTP headers and body of a request all the parameters, context, and data needed by the server-side component to generate a response. The statelessness on the server side that this approach enables improves performance and scalability, and simplifies the design and implementation of server-side components.

3.5.2 Constrained Application Protocol

The Constrained Application Protocol (CoAP) [198] is being specified in the Constrained RESTful Environments (CoRE) working group of the IETF. CoAP is a generic web protocol for constrained environments. It realizes the REST architecture for the most constrained nodes. CoAP can be used not only between nodes on the same constrained network but also between constrained nodes and nodes on the Internet. Nodes on the Internet that do not support CoAP but instead the Hypertext Transfer Protocol (HTTP) can interoperate with CoAP nodes through HTTP-CoAP proxies [39]. CoAP can also be used between devices in different constrained networks interconnected by an internet. The application areas of CoAP include different forms of M2M communication.

CoAP provides a request/response interaction model between application endpoints, supports built-in resource discovery, and includes key web concepts such as Uniform Resource Identifiers (URIs) and content-types. CoAP uses UDP as the transport protocol. CoAP meets the specialized requirements of constrained environments such as low overhead, simplicity, and ability to deal with sleeping nodes. The main example of operating environments CoAP targets is 6LoWPANs. However, CoAP also operates over traditional IP networks. CoAP has been extended to enable observation relationships between clients and resources [87].

To discover sensors and resources, CoAP may use either web linking [200], central resource directory [199], or DNS Service Discovery (DNS-SD) [210]. In Publication VII, we describe an additional discovery mechanism for CoAP. This mechanism is based on the use of a RELOAD P2P overlay network as an alternative to centralized services such as DNS-SD and resource directories.

CoAP defines four types of messages: Confirmable (CON), Non-confirmable (NON), Acknowledgement (ACK), and Reset (RST). Confirmable messages are sent reliably, whereas Non-confirmable messages are not. The ACK message acknowledges the reception of a Confirmable message. The RST message indicates that the recipient is not able to process a CON or NON message. The CoAP requests that can be carried in CON and NON messages include GET, PUT, POST, and DELETE. The GET method retrieves information, whereas the POST method requests that the representation enclosed in the request should be processed. The PUT method requests that a resource should be updated or created with the enclosed representation. Finally, the DELETE method requests the deletion of the resource identified by the request URI of the request.

3.5.3 P2P Architectures for M2M Communication

Up to this point, P2P technologies and M2M communication have been discussed as two isolated topics. However, there have also been research efforts investigating ways to merge these two paradigms.

DHTs have been proposed as a solution to data management and routing problems in WSNs due to their ability to perform an efficient lookup in a scalable and reliable manner [64]. However, there are also some challenges when deploying DHTs over WSNs [64]. First, DHTs do not consider the physical structure of the underlying network. Therefore, an overlay hop can have a high cost in terms of physical hops in the underlying network. Second, the maintenance traffic required by DHTs can cause network instability. It also consumes additional energy.

Geographic Hash Tables (GHTs) [177] are inspired by DHT systems. GHTs hash keys into geographic locations. In GHTs, data items are stored on the sensor node geographically nearest to the hash of its key. GHTs are built as overlays on top of existing ad-hoc routing protocols such as the Ad hoc On-Demand Distance Vector (AODV) protocol [168].

In [9], Awad et al. propose a DHT-like routing, storage, and lookup service for WSNs called the Virtual Cord Protocol (VCP). VCP places all nodes and data on a virtual cord-like topology using hashing. Routing is performed by using the virtual cord to find a path to a destination. Additionally, local neighborhood information is used for greedy routing.

The Virtual Ring Routing (VRR) protocol [32] is also inspired by DHTs. VRR organizes the nodes into a virtual ring, on which each node maintains a set of virtual neighbors that are used for routing. In the same way as in the Chord DHT, the virtual neighbors of a node n are those nodes whose identifiers are the closest successors and predecessors of the identifier of n in the identifier space of the virtual ring.

ScatterPastry [2] uses an extended Pastry DHT for overlay routing over WSNs. ScatterPastry can be either implemented as an overlay on top of an existing multi-hop ad-hoc routing protocol such as the Destination-Sequenced Distance Vector (DSDV) protocol [169], or alternatively, integrated within the network layer.

Muneeb et al. [3] have suggested using the Chord DHT over a WSN. The proposed protocol, called Chord for Sensor Networks (CSN), is a network layer protocol that organizes sensors on a logical ring. On the ring, the clockwise successor of each node n is the node that is geographically closest to n. In addition, each sensor node maintains a finger table containing $O(logN)$ other nodes. The cost of lookups is $O(logN)$. The sensors in CSN operate in low-power mode unless a query from the application is made. The performance of CSN is evaluated through simulations. The results of the simulations indicate that CSN can efficiently locate data in a WSN and that it can also scale to large-scale WSNs.

Further examples on the use of DHTs or DHT-like mechanisms over WSNs can be found in [64].

What is common to all of the solutions above is that they use a DHT or a DHT-like mechanism to solve the problem of routing within a single WSN. Another way to use P2P technologies in the context of WSNs would be to use a DHT for routing between different WSNs. Section 4.10 describes our work on such an approach.

4. Framework Architecture for Decentralized Communications

This chapter begins with an overview of the framework architecture for decentralized communications that this dissertation proposes. Next, the setup for the measurements and simulations that were used to evaluate the performance of the implementation of the framework and its components will be described. The remaining sections of the chapter will summarize one by one the publications of this dissertation.

4.1 Overview of the Framework

Figure 4.1 shows the P2PSIP architecture that was adopted by the P2PSIP working group already at the early stages of P2PSIP standardization. This architecture served as the starting point for the work in this dissertation. The architecture consists of components organized in different layers. The components on the top layer of the figure include the application protocols that are being run on top of the P2PSIP overlay network service. The only application protocol usage that the P2PSIP working group had defined at the time of writing this dissertation was SIP. The application protocols interface the underlying overlay networking layer through an API offering among other things data retrieval and storage operations. The overlay networking layer contains two components: the peer protocol and a DHT algorithm. Finally, the bottom layer, which is called the connection establishment layer, contains one component: the ICE protocol. ICE takes care of connection establishment for the peer protocol and SIP.

The publications making up this dissertation extend the P2PSIP architecture shown in Figure 4.1 into a more complete framework architecture for decentralized communications by proposing several new components that were added to the architecture. The resulting extended framework is

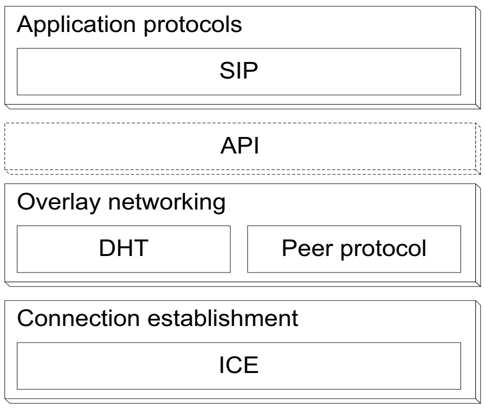


Figure 4.1. P2PSIP architecture

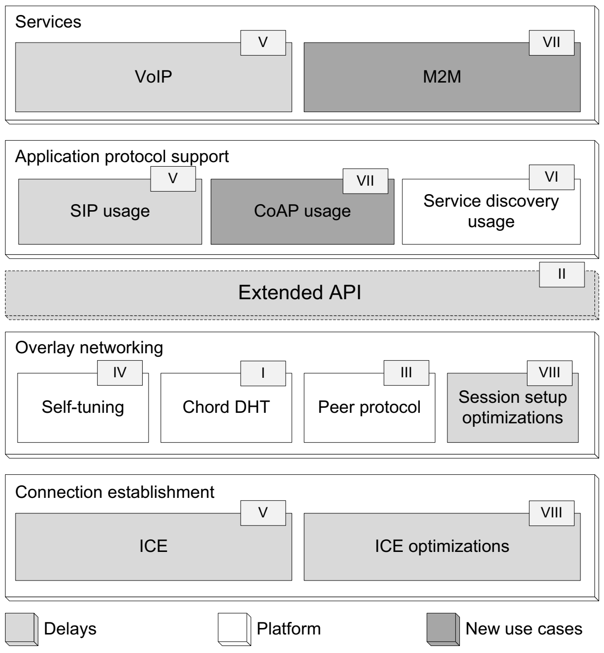


Figure 4.2. Framework architecture for decentralized communications

shown in Figure 4.2.

As a part of the work on this dissertation, all of the components of the framework were implemented. Some of the components, namely those corresponding to the components of the P2PSIP architecture shown in Figure 4.1, were implemented following existing or emerging standards (SIP, ICE, and peer protocol), or previous research (the Chord DHT). The performance of these components was analyzed through measurements and simulations. The rest of the components shown in Figure 4.2 are components that were designed, and in many cases, contributed to standardization as a part of the work associated with this dissertation. For each component of the framework, the figure also indicates the number of the publication focusing on that specific component. In the figure, the components have been divided into three categories using different colors. The light grey color represents work that was carried out to analyze and optimize delays in the framework. This includes the work on ICE, ICE optimizations, session setup delay optimizations, SIP usage, distributed VoIP service, and the extended API. The white color represents work on various aspects of the overlay networking framework. This includes the work on self-tuning, maintaining the Chord DHT, performance of the peer protocol, and the service discovery usage. Finally, the dark grey color represents work on new use cases, including the CoAP usage for RELOAD and the distributed M2M service. The subsections below will provide an overview of the components on different layers of the figure starting from the top layer, which is the Services layer.

4.1.1 Services

The top layer in Figure 4.2 contains the services that were, as a part of the work on this dissertation, implemented on top of the application protocol usages located on the underlying application protocol support layer. The services that were implemented include a distributed VoIP service that utilizes the SIP application protocol usage and a distributed M2M communication service that utilizes a novel CoAP application protocol usage that we defined. The performance of the distributed VoIP service is analyzed in Publication V, whereas the distributed M2M communication service is the topic of Publication VII.

4.1.2 Application Protocol Support

The second highest layer in Figure 4.2 is the application protocol support layer. This layer is responsible for defining how application protocols interface with the underlying overlay networking layer. The application protocol support layer consists of different protocol usages. The SIP usage provides the means to use SIP on top of the overlay networking layer. SIP is used as the session establishment, instant messaging, and presence protocol for the distributed VoIP service described above. When running SIP on top of the platform, the SIP usage for RELOAD [99] specification is followed. The performance of the SIP usage has been studied in Publication V.

The main enabling protocol for the distributed M2M communication service is CoAP. In order to be able to run CoAP on top of the platform, we designed the CoAP usage for RELOAD [103]. The CoAP usage for RELOAD is the topic of Publication VII.

The final usage on the application protocol support layer is the service discovery usage. This component enables applications to discover services from and register services in the overlay network. One service that is necessary for any distributed system is a relay service that is required due to the presence of NATs. In Publication VI, the author of this dissertation describes a service discovery usage for RELOAD that is based on ReDiR. The use case that Publication VI focuses on is a scalable distributed TURN relay service. Other examples of services that need to be discovered from the overlay network include a presence service, transcoding service, gateway service to client/server SIP networks, voice mail service, and a HTTP/CoAP proxy service.

4.1.3 Extended API

Below the application protocol layer are the operations, that is, the API that the framework provides to the application protocols. The basic set of operations includes lookup, store, join, leave, and connection establishment. We analyze the performance of these operations in Publication II.

The lookup and storage operations are used to implement data retrieval and storage. In the case of SIP, these operations can be used to implement a distributed SIP registrar service. A SIP endpoint uses the store operation to insert a mapping between its SIP URI and node-ID in the overlay. Another SIP endpoint can then use the lookup operation for rendezvous.

The join and leave operations allow an application to join the overlay network as either a peer or a client, and leave the overlay network.

The connection establishment operation provides an application a means to establish a direct connection to another endpoint participating in the overlay across NATs. The connections are established by exchanging ICE candidates in the peer protocol messages across the overlay and by running ICE to discover a working path between the endpoints. Once the P2P signaling protocol has established the connection, it hands the connection over to the application, which can then use the connection to exchange data or application protocol messages with the remote endpoint. The connection establishment operation is discussed in Publication V.

In addition to the basic set of the five operations described above, the publications of this dissertation extend the API with a set of new operations. These include service discovery, service registration, and data tunneling.

The service discovery and registration operations are discussed in Publication VI. The service discovery operation allows an application to discover a service provider from the overlay, whereas the service registration operation allows the application to register as the provider of a service in the overlay.

The data tunneling operation is used to send a small amount of data across the overlay without the need to establish a direct connection between the sender and the receiver. The data is carried in the payload of a P2P signaling protocol message that is in this dissertation referred to as the Tunnel message. The tunnel operation is used extensively by the CoAP usage for RELOAD that is described in Publication VII.

4.1.4 Overlay Networking

Underneath the API layer sits the overlay networking layer. The components on this layer include the Chord DHT algorithm, peer protocol, self-tuning, and session setup optimization components. These components will be discussed in more detail in the subsections below.

Chord DHT

A DHT algorithm implements the topology of the overlay network. In the publications of this thesis, the Chord DHT algorithm has been used since the P2PSIP working group has adopted it as the mandatory-to-implement DHT algorithm. As was discussed in Section 3.1.4, one of the major problems for DHTs is the problem of churn. To efficiently cope with churn, the parameters of the DHT such as the maintenance interval need to be configured appropriately. In Publication I, we have studied the impact of different maintenance intervals and churn rates on the performance of P2PSIP.

Peer Protocol

The overlay networking layer uses the P2PSIP peer protocol to send and receive messages, and to establish connections. During the research for this dissertation, two peer protocols were implemented. In the early phases of the work, the P2PP protocol was used. P2PP was one of candidates for the P2PSIP peer protocol in the early stages of P2PSIP standardization. Later, also the RELOAD protocol was implemented following the decision of the P2PSIP working group to merge P2PP with RELOAD and adopt RELOAD as an official working group item. We have studied the performance of the peer protocol especially in Publication III, in which we focus on the performance of RELOAD in mobile environments.

Self-tuning

As was discussed in 3.1.4, self-tuning is a process through which a DHT algorithm adjusts its parameters to match the operating conditions of the overlay. In the framework, it is the task of the self-tuning component to implement this behavior. The prerequisite for self-tuning is the ability to estimate the operating conditions of the overlay. Publication IV proposes novel mechanisms for operating condition estimation. We have specified self-tuning extensions for RELOAD in [138].

Session Setup Optimizations

The research in Publication V demonstrated that P2PSIP session setup delays can be unacceptably high. To address this issue, the author of this dissertation developed novel mechanisms for reducing the session setup delay in Publication VIII. A summary of these mechanisms will be presented in Section 4.6.

4.1.5 Connection Establishment

Beneath the overlay networking layer sits the connection establishment layer. This layer consists of the ICE and ICE optimization components. The sending and reception of all peer and application protocol messages happens through the API that the connectivity layer exposes to the upper layers.

ICE

The connection establishment layer relies on the ICE framework for establishing direct connections between peers located behind NATs and firewalls. ICE is also used for keepalive signaling. The impact of ICE on

P2PSIP performance has been studied in Publication V.

Besides ICE, the connectivity layer also contains STUN and TURN server implementations. To be able to utilize the distributed STUN and TURN services provided by the overlay network, the ICE component needs to learn the addresses of STUN and TURN servers. This information is passed to the ICE component via the API of the connectivity layer following a TURN service discovery procedure implemented using the service discovery component of the framework.

ICE Optimizations
To further address the problem of high session setup delays that the session setup optimization component (see Section 4.1.4) attempts to alleviate, the author of this dissertation studied the impact of different ICE parameters on the session setup delays in Publication VIII. These optimizations are summarized in Section 4.6.

4.1.6 Summary of the Framework

The most important features of the framework are summarized below.

The self-tuning properties of the framework make it adaptive, scalable, and resistant to churn and changes in the size of the overlay network. Self-tuning addresses the challenges associated with providing DHT based services in the real Internet.

The service discovery and registration component enables the provision of new services in the overlay network. This makes it possible to offer services such as a TURN relay service or an IMS gateway service to the user population in a scalable manner.

The extended API of the overlay networking layer meets the needs of a wide variety of application usages. To verify whether the API is generic enough, three different usages, namely the SIP, CoAP, and service discovery usages, were implemented as a part of the work on this dissertation.

The framework contains optimizations that allow it to meet the performance requirements of different application protocols. Publication VIII proposes optimizations that reduce the session setup and connection establishment delays, whereas Publication II introduces optimizations increasing the reliability of lookup operations.

The design choices, implementation, and performance of the implementation of the framework and its components have been verified through extensive testing in the real Internet, mobile devices and networks, and simulators.

When implemented together in the form of the proposed framework, the new components that were designed as a part of the work on this dissertation enable an overlay networking service to fulfill the requirements that were stated out in Chapter 2.

The sections below will summarize the work on the components of the framework. Publication I (Section 4.3) focuses on the Chord DHT. Publication II (Section 4.4) analyzes the performance of the basic version of the API that the overlay networking layer exposes to application protocol usages. Extensions to the API will be described in later sections. The focus of Publication V (Section 4.5) is on the performance of ICE-based NAT traversal and the SIP usage. Publication VIII (Section 4.6) describes session setup and ICE optimizations. Publication IV (Section 4.7) describes our work on operating condition estimation and self-tuning. Publication VI (Section 4.8) deals with service discovery. Publication III (Section 4.9) studies the performance of the peer protocol. Publication VII (Section 4.10) is about the CoAP usage and the distributed M2M communication service it enables. Finally, in Chapter 5, conclusions will be drawn about the framework, focusing especially on the sum of its parts.

4.2 Setup for Analyzing the Performance of the Implementation of the Framework

4.2.1 Implementing the Framework

In order to make the prototype implementation of the framework able to run on as many devices and operating systems as possible, the framework was implemented in the Java programming language. Java was chosen since it made the effort of porting the implementation to environments such as Java Standard Edition (J2SE) on desktop Linux and Windows, Google Android, Java Micro Edition (J2ME), and embedded Linux systems easier. In experiments on mobile phones, the J2ME and Android versions of the prototype were used. In PlanetLab experiments and in experiments on Windows and Linux desktop computers, the J2SE version was used. In the M2M experiments, the prototype was run on an embedded Linux operating system on top of the CACAO Java Virtual Machine (JVM) for ARM processors.

As mentioned above, PlanetLab was used extensively in the experiments. PlanetLab [170] is a global platform for deploying and evaluating network services and protocols. At the time of writing this dissertation, PlanetLab consisted of over 1000 Internet nodes at more than 500 sites located all over the world. Researches are using PlanetLab to develop among other things new peer-to-peer and distributed storage systems. The main reason for selecting PlanetLab as the platform for the experiments is that this made it possible to run the experiments on a global scale with a high number of nodes.

4.2.2 Simulator

The author of this dissertation also implemented an event-driven, message-level P2PSIP simulator around the framework. To enable running the prototype in a simulated mode, a network abstraction layer that sits below the connection establishment layer was designed. The abstraction layer hides away the fact whether the framework is being run on top of a real or simulated network socket layer. It also enabled the use of the same codebase when running the framework in the simulator and on real devices and networks. The simulator has been validated in Publication VI.

4.2.3 Traffic Model

Many of the publications of this thesis use a similar traffic model. In the experiments, P2P lookup traffic consists of lookups related to VoIP calls and presence. Calls are modeled according to busy hour traffic volumes. The number of busy hour call attempts per user is 2.21. This rate of call attempts was chosen based on the results in [6, 47]: in [6], it is suggested that VoIP users initiate 13 calls per day. Further, [47] states that 17% of calls that a user initiates during a day can be used to represent busy hour traffic. This results in a number of busy hour call attempts equal to 2.21. This figure is used as the mean rate for the arrival of calls, which is modeled as a Poisson process.

For presence the traffic, the number of buddies that a user has in the buddy list of her application is assumed to follow the power law distribution, as reported in [128, 156] with an average of 22. This number of buddies was chosen based on the results in [159]. After having joined the overlay, each user initiates lookups to fetch the contact information of her buddies from the overlay. Since users typically call their friends instead of strangers [44], it is assumed that $\frac{2}{3}$ of the calls are placed to users on the buddy list.

Chord Parameters

The size of Chord's finger table and successor list were set to $logN$, where N is the network size, based on the recommendation in [205]. Also a Chord predecessor list is used to improve stability. The size of the predecessor list is set to $\frac{1}{2} \times logN$ to ensure that the probability that all predecessors fail at the same time is low. When configuring the Chord maintenance interval, the recommendation in [132] that roughly $\Omega(\log^2 N)$ rounds of stabilization should occur in the time it takes for N new peers to join or $\frac{N}{2}$ peers to leave the overlay is followed. To configure the maintenance interval, also the results of Publication I are used in addition to the above-mentioned recommendation.

4.3 Maintenance of P2PSIP Overlays

As was discussed in Section 3.1.4, DHT algorithms are difficult to configure. The primary challenge with configuring them is selecting an appropriate frequency for the periodic maintenance routine that most DHTs run to counter the negative effects of churn on performance. Churn and the frequency of maintenance operations naturally impact also the performance of P2PSIP overlays. This section will summarize the work in Publication I on studying the impact of different churn rates and maintenance intervals on the performance of a real-world P2PSIP overlay network. Publication I was, to the best of the author's knowledge, the first study on the effects of churn and maintenance interval in a real-world P2PSIP overlay network running in the Internet.

In the experiments carried out for Publication I, the performance of a 500-peer P2PSIP overlay running in the PlanetLab network was studied using different churn rates and maintenance intervals. The churn rates that were used correspond to user interarrival and departure times of 5s, 10s, and 30s. In a 500-peer overlay, these correspond to mean session times of 42min, 83min, and 250min. Session time refers to the time between the user joining the overlay and leaving it. The maintenance intervals ranged between 15s and 360s. The performance metrics that were measured include hop count, lookup delay, percentage of failed lookups, percentage of failed self-lookups, percentage of timed out requests, and the ratio between maintenance traffic, lookup traffic, and total P2PP traffic in the overlay. Maintenance traffic refers to the P2PP traffic generated by periodic DHT maintenance operations and maintenance traffic generated by peers joining and leaving the system. Lookup traffic refers to P2PP traffic generated by service-related lookup operations associated with establishing calls and presence subscriptions; maintenance-related lookups are not included in this figure. The total traffic includes all the P2PP traffic exchanged in the overlay, that is, both maintenance traffic and lookup traffic. Note that the experiments focused only on measuring P2PP traffic, the traffic generated by SIP messages needed to set up and tear down calls and presence sessions was not measured.

The findings in Publication I show that the dominant traffic type in a P2PSIP overlay is the traffic generated by DHT maintenance routines. The volume of traffic generated by lookups related to VoIP calls, instant messaging, and presence is much lower. Depending on the churn rate and maintenance interval, 80-98% of the traffic in the overlay is maintenance traffic. Therefore, the percentage of maintenance traffic can be higher than in other use cases of DHTs [61]. One implication of this is that P2P algorithm optimizations that piggyback maintenance information in lookup messages [151] are not efficient in a P2PSIP overlay due to the low amount of lookup traffic.

The results in Publication I also indicate that the total amount of traffic that a single peer processes in a P2PSIP overlay is relatively low. Even at the highest churn rate and shortest maintenance interval, the average incoming and outgoing traffic per peer is on the average only 355 kilobytes per hour. Therefore, the bandwidth usage is low compared to many other classes of P2P applications [114, 196]. This observation suggests that the traffic load of P2PSIP should not pose a problem even for mobile devices participating in a P2PSIP overlay. To verify this observation, the performance of P2PSIP on mobile devices is studied in more detail in Publication III.

The results also demonstrate the impact that churn has on lookup delays. As churn increases, the routing tables of peers become less accurate. This has the impact of increasing the path length (i.e., hop count) of lookups. Too infrequent maintenance operations have the same impact; if the maintenance rate is too low for the prevailing churn rate, routing tables become inaccurate and path length and thus also the lookup delay increases. An inappropriate maintenance rate also results in a dramatic increase in lookup errors; if the maintenance rate is too low, the failure rate can become so high that the lookup service provided by the overlay becomes in practice unusable. A perhaps more surprising finding is that also too frequent maintenance operations result in an increase in lookup failures. The explanation for this is that when the maintenance rate is high, peers that become unresponsive due to temporary overload are removed from the routing tables of other peers rather aggressively. Once the nodes become responsive again, they are inserted back. This instability results in a considerable increase in the amount of failed lookups.

The above-described challenges with selecting an appropriate maintenance rate demonstrate the need for a solution that can adapt (i.e., self-tune) the maintenance rate as a function of churn rate and other changing operating conditions. Section 4.7 will describe our work on mechanisms enabling self-tuning.

The results in Publication I indicate that path error is clearly a more common reason for lookup failures than other errors such as timeouts, exceeded Time-To-Live (TTL) limits, and routing loops. A path error refers to a forwarding error at one of the intermediate nodes forwarding a peer protocol request towards its final destination. Further, reverse path errors are very rare compared to forward path errors. This observation is very relevant for P2PSIP, as reverse path errors have been considered as an obstacle for using the recursive routing mode in P2PSIP.

Yet another conclusion of the results in Publication I is that the lookup failure rate is rather high even when the optimal maintenance interval is used at the lowest churn rate. This suggests that P2PSIP would greatly benefit from additional reliability mechanisms for lookups. To address this issue, the impact of end-to-end lookup retransmissions was studied in Publication II.

The results on P2PSIP lookup delays in Publication I also called for a more detailed analysis of P2PSIP session setup delays. Such an analysis is carried out in Publication II and Publication V.

To summarize, the work in Publication I contributed to the understanding of the impact of churn and maintenance interval on P2PSIP, to the understanding of how to best configure a P2PSIP overlay and how the traffic in a P2PSIP overlay differs from other P2P systems. It also identified areas of research that demanded further attention, including self-tuning, reliability mechanisms, performance of P2PSIP on mobile devices and networks, and a detailed analysis of P2PSIP session setup delays.

4.4 P2PSIP Delays

Due to the $O(logN)$ lookup cost of DHTs, a P2PSIP system has inherently higher delays than a client/server SIP system, where the lookup cost is $O(1)$. In a P2PSIP overlay, the relevant delays from the viewpoint of user experience include the delay of joining the overlay, the session setup delay, and the delay associated with leaving the overlay. The join delay influences the time between the user launching the P2PSIP application and the user being able to initiate the first call. The session setup delay influences the time it takes for the user to establish for instance calls, chat sessions, and presence subscriptions. The leave delay affects the time it takes for the P2PSIP application to close. This is because the application needs to inform other peers of the user's departure from the overlay. In a client/server SIP system, the delay corresponding to the join delay is the registration delay. The delay corresponding to the leave delay is the de-registration delay. All of these delays are studied in Publication II. To the best of the author's knowledge, Publication II was the first study that analyzed the delays in a real-world, global-scale P2PSIP overlay. In addition, also to the best of the author's knowledge, it was the first study that compared the performance of P2PSIP and client/server SIP in the real Internet.

In the experiments carried out in Publication II, the performance of a P2PSIP overlay running in PlanetLab was studied. Three different network sizes, 250, 500 and 1000 peers, and 6 different churn rates corresponding to user interarrival and departure times between 1s and 40s were used. In addition to the P2PSIP measurements, also delays in a client/server SIP system that was facing similar load as the P2PSIP overlay were measured.

The results in Publication II show that the P2PSIP join operation delay can be multiple, 11-20, times higher than the SIP registration delay. The average P2PSIP session setup delay is 2.4-5.0 times higher than the average SIP session setup delay and the average P2PSIP leave operation delay 11-14 times higher than the SIP de-registration delay.

The results also indicate that at high churn rates, the failure rate of P2PSIP lookup operations becomes unacceptably high even though the failure rate remains at zero in a client/server SIP system facing the same

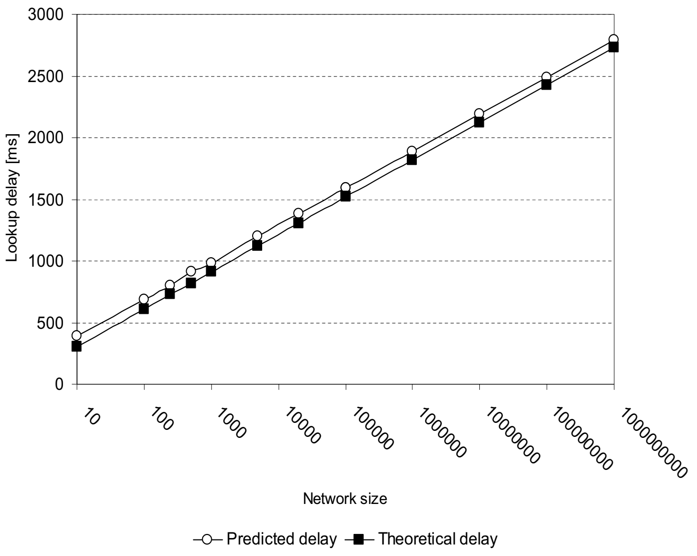


Figure 4.3. Lookup delay as a function of network size

load. In the extreme case, that is, in a 250-peer overlay where the average session time is 4 minutes, over 17% of lookup operations fail. It was also discovered that the dominant reason for a lookup failure is that the lookup gets routed to an unstable peer, that is, to a peer that is in the middle of the process of joining or leaving the overlay. This observation suggests that an efficient way to deal with the majority of lookup failures is to route around unstable peers by repeating the lookup operation immediately after the unstable peer is detected. To enable such detection, we added an explicit notification that informs the peer that originated the request about the unstable peer on the routing path. This strategy is very efficient; it eliminates nearly all of the errors caused by unstable peers in the routing path. As an example, in the above-mentioned case where over 17% of lookups failed, the failure rate was brought down to roughly 2%.

Based on the results on measuring the lookup delay in networks of different sizes, it is possible to predict how the lookup delay will behave when the network size increases further. This prediction is depicted in Figure 4.3, which shows the lookup delay as a function of network size. The figure is based on the results of experiments with a user interarrival and departure time of 5s. The x-axis of the figure uses logarithmic scale. The figure was created by fitting a logarithmic curve to the measurement results of Publication II. In the measurements, a large number

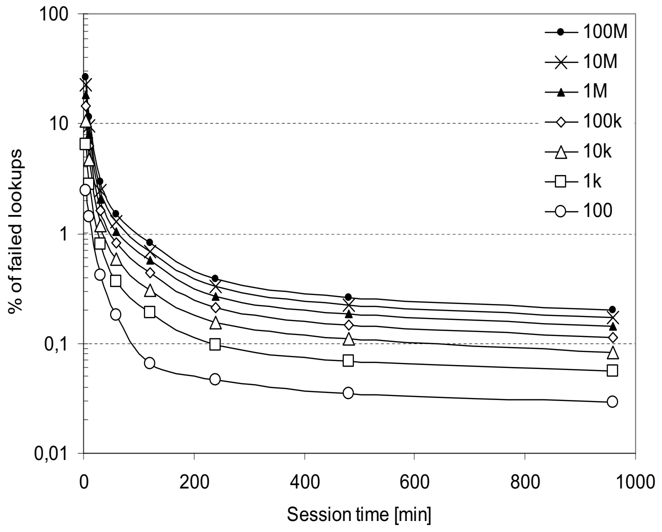


Figure 4.4. Percentage of failed lookups as a function of session time

of delay values were obtained for each network size. The differences between the measured delay values are statistically significant. However, it must be noted that the prediction of delay values in Figure 4.3 should be considered as approximate since the predicted values have been obtained through extrapolation. From the figure, we can observe that the delay appears to grow in a logarithmic fashion and remain reasonable even in a network of 1000 million peers.

In Figure 4.3, also the theoretical average path length of messages in a Chord DHT is depicted. According to [205], the theoretical average path length is $\frac{1}{2} \times log_2(N)$. Based on this information, the average lookup delay can be calculated by multiplying the average path length with the average delay of exchanging messages between two overlay peers, which was obtained from the measurement results of Publication II. The lookup delay predicted using the theoretical average path length is depicted in Figure 4.3. From the figure, one can observe that the two predictions seem to be fairly close to each other. The reason why the delay obtained through extrapolation is slightly higher is explained by the fact that in a real-world DHT, the average path length is higher than the analytical model predicts since temporary instability caused by churn increases the average path length.

Figure 4.4 predicts how the lookup failure rate will behave as a function of the session time. The figure includes session times ranging from five minutes to 16 hours and network sizes ranging from 100 peers to 100 million peers. The y-axis in the figure uses logarithmic scale. In the same way as in the case of Figure 4.3, the prediction for the lookup failure rate was created by fitting a logarithmic curve to the measurement results of Publication II, from which the percentage of failed lookups can be obtained as a function of network size. The differences between the measured failure rates are statistically significant. However, it must be noted that the prediction of lookup failure rates in Figure 4.4 should be considered as approximate since the predicted values have been obtained through extrapolation. From the figure, one can observe that, as can be expected, the lookup failure rate increases as the mean session time decreases. This is because shortening the session time increases the frequency of peer arrivals and departures (i.e., churn). The instability caused by churn results in an increase in the failure rate. One can also see that for any given session time, the percentage of failed lookups grows as a function of network size. As an example, when the session time is 10 minutes, the failure rate in a 100 peer overlay is 1.42% and 11.33% in a 100 million peer overlay. Maintaining a large network in a stable state requires more frequent maintenance operations than maintaining a smaller network experiencing the same mean session time; as was discussed in Section 3.1.4, a Chord network that is in a ring-like state stays in the ring-like state as long as nodes execute $\Omega(\log^2 N)$ rounds of maintenance operations in the time it takes N new nodes to join or $\frac{N}{2}$ nodes to leave the system. In a larger network, peers see node failures, departures, and arrivals more frequently in their routing tables than in a smaller network experiencing the same mean session time. Despite of the higher frequency of maintenance operations in larger overlays, the more frequent changes in routing tables result in more lookup timeouts, path errors, and exceeded TTL limits. This explains the higher failure rates for larger network sizes in Figure 4.4. From the figure, one can also conclude that when the mean session time is extremely low (less than around 10 minutes), the failure rates become so high for all network sizes except for the smallest ones that the lookup service that the overlay provides becomes in practice unusable. In contrast, for mean session times higher than around 60 minutes, which translates to mean peer interarrival and departure times of 1.8s – 14.4s depending on the network size, the lookup failure rate is rather low for all network sizes.

Publication II also derived insights for P2PSIP protocol design. First, due to the high lookup failure rate under heavy churn, additional reliability mechanisms are needed to achieve acceptable performance. Second, P2PSIP also benefits from a mechanism that can quickly recover from unstable peers on the routing path. We proposed a mechanism to address these two concerns and showed that it can considerably reduce the lookup failure rate. Finally, rather than focusing on optimizing only the lookup delay, attention should also be paid to the high P2PSIP join and leave operation delays.

To summarize, the work in Publication II contributed to the understanding of the magnitude of different delays in a P2PSIP overlay and how they compare to the corresponding delays of client/server SIP. Both the impact of churn and network size on the delays was studied. Publication II also analyzed the failure rate of lookups, showed that it can be unacceptably high, and proposed a mechanism that addresses the problem. Further, insights were derived for P2PSIP protocol design. One topic that Publication II left as future work is the impact of NATs on P2PSIP delays. This has been analyzed in Publication V, which will be summarized in Section 4.5.

4.5 P2PSIP and NAT Traversal

Previous research on the performance of P2PSIP has assumed that all peers are publicly reachable (see Section 3.4.4). The same assumption was made in the work on P2PSIP delays in Publication II. To gain an understanding of the impact of NAT traversal on P2PSIP session setup delays, the ICE NAT traversal framework was implemented and integrated into the P2PSIP prototype used in the publications of this dissertation. Next, experiments were carried out in PlanetLab using the prototype. The results of these experiments are reported in Publication V.

To the best of the author's knowledge, Publication V was the first study on the impact of ICE on P2PSIP performance. In the experiments carried out for Publication V, mobile phones and personal computers located behind NATs participated as clients in a 1000-peer P2PSIP overlay network running in PlanetLab. The mobile phones were using Third Generation (3G) High-Speed Downlink Packet Access (HSDPA), whereas the laptops were connected to the Internet through Asymmetric Digital Subscriber Line (ADSL) connections. The focus was on measuring the session setup delays between the mobile phones and laptops. In addition to measuring P2PSIP session setup delays, also session setup delays in a client/server SIP system were measured using the same setup. In the SIP measurements, the P2PSIP overlay was replaced with a centralized SIP proxy-registrar server. This resulted in four different access network and SIP architecture combinations: mobile P2PSIP, wired P2PSIP, mobile client/server SIP, and wired client/server SIP. For each combination, four different scenarios were studied: (1) both nodes are publicly reachable and do not use ICE (*no ICE*), (2) both nodes are publicly reachable and use ICE (*no NATs*), (3) both nodes are behind NATs with Endpoint Independent Mapping and Filtering behavior (*EIMF NATs*), and both nodes are behind NATs with Address and Port Dependent Mapping and Filtering Behavior (*APDMF NATs*). The results of the measurements are summarized below.

The mere act of enabling ICE between two publicly reachable P2PSIP nodes makes the P2PSIP session setup delay multiple times larger compared to the case that ICE is not used. As an example, for the wired P2PSIP scenario, the delay becomes 4.4 times larger. This is because enabling ICE adds several new components to the call setup delay, including ICE candidate gathering for SIP, exchange of ICE candidates for SIP across the overlay, performing the ICE negotiation for SIP, candidate gathering for RTP, and ICE negotiation for RTP.

If the nodes are behind EIMF NATs, the session setup delay increases further. This happens since due to the presence of NATs, ICE connectivity checks between the highest priority candidate pairs (that is, the host candidates) fail. Due to this failure, the ICE negotiation will continue until a pre-defined first time limit called the soft deadline is reached. The purpose of the soft deadline is to give ICE additional time to discover a relay-free path between the nodes. If such paths have been discovered when the soft deadline is reached, the path having the highest priority is selected and ICE is concluded. If no relay-free path has been discovered when the soft deadline is reached, ICE processing will continue.

The highest delays occur when both of the nodes are behind APDMF NATs. In this case, it is not possible to find a relay-free path. Therefore, the ICE negotiation will continue even after the soft deadline is reached. The processing will continue until a point in which all ICE checks have either succeeded or timed out. This point is called the hard deadline. The hard deadline is dictated by the default values of STUN and ICE timers, the number of allowed STUN request transmissions, and the maximum allowed number of connectivity checks. If using the default values, it occurs roughly after 10 seconds from the beginning of the connectivity checks.

The main reason behind high P2PSIP session setup delays is the fact that the ICE negotiation is done twice: first for SIP and later for RTP. Therefore, an efficient strategy for reducing the delay would be to find a way to eliminate one of the negotiations. Section 4.6 will summarize the work that the author of this dissertation has done on such optimizations.

ICE also paces connectivity checks differently for SIP and RTP. For SIP (and for non-RTP sessions in general), the ICE RFC specifies that the minimum interval between STUN transactions is 500ms. For RTP, the minimum is 20ms. The higher interval has the effect of slightly increasing the ICE negotiation delay for SIP. Thus, one way to reduce the ICE delay for SIP would be to use a more aggressive STUN transaction interval. The author of this dissertation has studied such an approach in Publication VIII.

A further factor impacting the delays is the location of the TURN server. In client/server SIP, a provisioned TURN relay of the user's SIP domain is typically used. This relay is usually geographically close to the user. However, in P2PSIP, the TURN server is selected from the P2PSIP overlay randomly without taking geographical proximity into account. This results in higher delays for P2PSIP than client/server SIP. Consequently, P2PSIP would benefit from the ability to find a TURN server that is close to the nodes wishing to establish a connection.

In the mobile P2PSIP and SIP scenarios, all the components of the session setup delay are clearly higher than when fixed ADSL access is used. In a cellular (HSDPA) access network, the individual components of the delays are 1.5-11.8 times higher. In a cellular network, also the fact whether the terminal has a dedicated radio channel allocated when it starts the ICE checks and other signaling procedures impacts the delays.

Publication V also compared the P2PSIP and SIP session setup delays to ITU-T recommendations. ITU E.721 [96] recommends an average delay of 8.0s with a 95th percentile of 11.0s for international calls. Compared to these recommendations, the delays of mobile P2PSIP are never acceptable when ICE is used. When ICE is enabled, the delays of wired P2PSIP are acceptable only when the peers are not located behind NATs. For wired and mobile client/server SIP, the delays are acceptable as long as the hosts are not located behind the most restrictive types of NATs (i.e., APDMF

NATs).

To summarize, Publication V showed that when ICE is used, P2PSIP session setup delays become in many cases unacceptably high. This finding is in stark contrast with previous research that has studied the performance of P2PSIP in NAT-free environments without using ICE. Publication V identified the main reasons causing the delays to be so high and suggested ways to reduce the delays. It also compared the session setup delays of P2PSIP to client/server SIP and showed that also the delays of client/server SIP become unacceptably high when the most restrictive types of NATs are involved. Based on the findings in Publication V, it was evident that future work would be needed to design mechanisms that can reduce the P2PSIP session setup delay. Section 4.6 will summarize the work that the author of this dissertation has done on such mechanisms.

4.6 Reducing P2PSIP Session Setup Delays

Publication V showed that P2PSIP session setup delays can be unacceptably high when the communicating hosts are behind NATs. Thus, there is a need to develop mechanisms to bring the delays down. In this section, the work that the author of this dissertation did in Publication VIII will be summarized. In this work, novel mechanisms that can dramatically reduce P2PSIP session setup delays were developed.

Publication VIII studies nine different scenarios applying various optimizations that the author of this dissertation developed. The optimizations attempt to reduce the P2PSIP session setup delay. The performance of the optimizations is analyzed through simulations using the P2PSIP simulator developed as a part of the work on this dissertation. The size of the simulated overlay was 5000 peers. During the simulations, the overlay was experiencing churn corresponding to peer online time of eight hours. Three different NAT configurations were used. In the *No NATs* configuration, both communicating parties were publicly reachable. In the *EIMF NATs* configuration, the parties were located behind NATs with Endpoint Independent Mapping and Filtering (EIMF) behavior. Finally, in the *APDMF NATs* configuration, the hosts were located behind NATs with Address and Port Dependent Mapping and Filtering (APDMF) behavior. In the simulations, the nodes establishing P2PSIP sessions were using 3G HSPDA cellular radio access.

The baseline scenario is *Unoptimized P2PSIP*. In this scenario, no optimizations are used. This scenario corresponds to the one that was studied in Publication V.

In the *Timers & Joint Candidate Gathering* scenario, the interval, called *Ta*, that ICE uses to pace STUN transactions for non-RTP sessions is modified. Further, ICE candidates are gathered simultaneously for SIP and RTP. The performance improvement that this scenario achieves was found to be rather minor.

The *Reuse ICE Result* scenario reuses the result of the ICE connectivity checks for SIP when prioritizing the ICE candidate pairs for RTP. This strategy turns out to be rather efficient especially in the *APDMF NATs* configuration, reducing the delay by 30%. However, the optimization is less efficient in the other NAT configurations since in them, the ICE negotiation delay is not as dominant a component of the session setup delay as in the *APDMF NATs* configuration.

In the *SIP and RTP Mux* scenario, SIP and RTP are multiplexed on the same port. This strategy eliminates the need to perform one of the ICE negotiations. Depending on the NAT configuration, it reduces the delay by 17-38%.

In the *SIP via Overlay (SvO)* scenario, SIP messages are sent across the overlay encapsulated in RELOAD messages. This strategy turns out to be surprisingly efficient, outperforming even the *SIP and RTP Mux* scenario. It cuts the session setup delay by 36-43% compared to the *Unoptimized P2PSIP* scenario, depending on the NAT configuration.

In the next two scenarios, called *SvO Rc=6* and *SvO Rc=5*, the SvO scenario is still used. However, additionally, the number of STUN request transmissions, which is controlled by the STUN *Rc* parameter, is reduced from 7 to 6 and 5. In the *SvO Rc=5 N=50* scenario, also the maximum number of ICE connectivity checks, *N*, that a host can perform is limited to 50. The *SvO Rc=5 N=50* scenario results in the greatest improvements, reducing the delay by 36-62% compared to the *Unoptimized P2PSIP* scenario.

In the final scenario, *SvO SigComp*, in addition to the use of the *SvO Rc=5 N=50* optimization, SIP messages are compressed using SigComp. This strategy turns out to be the most efficient one, cutting the delay by 41-64% depending on the NAT configuration.

Similar to Publication V, Publication VIII also compared the delays of the best performing *SvO SigComp* scenario to ITU-T recommendations for call setup delays. According to ITU E.721 [96], the average delay should be no more than 8.0s and the 95th percentile no more than 11.0s for international calls. The set of optimizations that *SvO SigComp* uses brings the delays below or down to the 8.0s limit in the *No NATs* and *EIMF NATs* NAT configurations. In the *APDMF NATs* scenario, the delay remains slightly higher than the ITU-T recommendations even if the optimizations bring the delay down by 20 seconds (64%). This is because when the nodes are using cellular access, as they were in the simulations, it is difficult to reduce the delays even lower than this due to the high cost of sending data over the radio interface. However, the situation changes when the nodes are located in fixed (ADSL) access networks; in this scenario, the optimizations reduce the delay of also the *APDMF NATs* scenario down to levels that are in line with the ITU-T recommendations.

Publication VIII also compared the delays that the proposed optimizations achieve to the session setup delays of client/server SIP. The findings include that, perhaps unexpectedly, the *SvO SigComp* scenario produces lower session setup delays in the *APDMF NATs* scenario than unoptimized client/server SIP.

To summarize, Publication VIII developed mechanisms and optimizations that can successfully bring P2PSIP session setup delays down to levels that are acceptable even when compared against the strict ITU-T recommendations for call setup delays.

4.7 Self-tuning

As was discussed in Section 3.1.4, DHT-based systems are non-trivial to configure. Even if the system is correctly configured for one set of operating conditions, this configuration will not remain optimal if the operating conditions change. These challenges were evident in the experiments on the impact of churn and maintenance interval on P2PSIP that were summarized in Section 4.3. Based on these experiments, a DHT would greatly benefit from the ability to adapt its parameters as the operating conditions such as churn and network size change.

A prerequisite for adaptive behavior is the ability to estimate the prevailing operating conditions of a running overlay network. Section 3.1.4 presented an overview of such mechanisms. An ideal mechanism for estimating operating conditions would be able to form an accurate picture of the status of the overlay using only information that is locally available to a peer. Use of only local information, that is, passive approach to operating condition estimation is preferable over active approaches since in the latter, each peer needs to perform frequent measurements in the overlay. The cost of such measurements is that they introduce extra traffic to the system. Due to this reason, we chose to focus on passive approaches in our work on operating condition estimation. This work is documented in Publication IV and will be summarized in this section.

Publication IV starts by evaluating the accuracy of existing passive approaches to operating condition estimation and continues by designing new mechanisms that have improved accuracy. The performance evaluation of the mechanisms is done through simulations. The simulations were carried out using the P2PSIP simulator that the author of this dissertation developed. The maximum size of the simulated overlay was 10000 peers. All peers were assumed to be located behind P2P-friendly NATs (i.e., NATs using endpoint independent mapping and filtering behavior). During the simulated period of time, the overlay was experiencing different levels of churn. In Publication IV, different configurations of estimation mechanisms were studied. These configurations and their performance are summarized below.

In the *Basic* configuration, existing, unoptimized Network Size (NS), Join Rate (JR), and Leave Rate (LR) estimation mechanisms were used. The accuracy of these estimates was found to be rather poor. The leave rate estimation mechanism had the greatest inaccuracy, which was found to be the result of two factors. First, newly joined peers produce very inaccurate leave rate estimates. This is due to the fact that leave rate estimation is done based on node failures that a peer observes in its routing table. Such failures are recorded in a data structure called the failure history. Since a newly joined peer typically has no or very few entries in its failure history, it produces a very inaccurate leave rate estimate. Second, the temporary instability caused by churn causes peers to overestimate the leave rate. This is because peers add too many failures to their failure histories.

In the *LR Optimizations* configuration, several new techniques were applied. First, to improve their leave rate estimates, peers joining the overlay bootstrap their failure histories by downloading the failure history of the admitting peer (i.e., the peer that helps the joining peer in joining the overlay). Second, the identities of failed peers are stored in the failure history to prevent the same peer from being added multiple times to the failure history. This can happen due to the instability caused by churn.

Third, the entries in the failure history were allowed to age, that is, to have a weight inversely proportional to their age in the leave rate estimation algorithm. Fourth, the network size at the time when a given failure was observed was stored in the failure history. This historic value was then used when calculating the leave rate estimate. Our results show that these four mechanisms improve the accuracy of the estimate considerably.

The *Two NS Estimates* configuration focuses on the network size estimate. In addition to using the density of node-IDs to calculate a primary network size estimate, also the distance of finger pointers from their ideal positions is utilized to produce a secondary network size estimate. The final estimate is calculated as a weighted average over the primary and secondary estimates. The improvement this mechanism achieves over the *Basic* configuration was found to be rather small. However, the mechanism can still be useful in situations where using only one estimate is unreliable. This can happen for instance when the neighbor table of a peer does not have enough entries for the node-ID density based size estimation mechanism to produce a reliable estimate.

In the *Estimate Sharing* configuration, peers share their NS, JR, and LR estimates by piggybacking them on overlay stabilization messages. The final estimate is calculated as a weighted average over the shared estimates. This configuration achieves a considerable further improvement compared to the previous configurations.

In the final configuration, called *All Optimizations*, several new mechanisms are introduced. First, peers use the 75th percentile of the received estimates as the final estimate. The 75th percentile is used rather than the median or average since it captures changes faster. Second, the failure detection mechanism is improved so that it handles differently timeouts that occur because of a routing error and timeouts that occur because a peer has left the overlay or crashed. Third, the failure detection mechanism was also improved to differentiate between connections that are terminated due to a peer leaving from the overlay and connections that are terminated because the remote peer is no longer within the appropriate range in the routing table. Fourth, joining peers used the leave rate estimates of the admitting peers in addition to relying on the downloaded failure history. Fifth, failure histories were periodically cleared if the entries in them were very old.

The comparison between the five configurations discussed above showed that the *All Optimizations* configuration produces the most accurate estimates. Compared to the *Basic* configuration, it achieves a 239%, 55%, and 515% improvement for the network size, join rate, and leave rate estimates, respectively.

To summarize, existing passive approaches to operating condition estimation are not accurate enough. Their biggest shortcoming is that they do not perform well under churn. The mechanisms presented in Publication IV achieve a considerable performance improvement compared to previously proposed mechanisms. One important finding is that in order to improve accuracy, it is necessary for peers to share their estimates by piggybacking them to overlay maintenance messages and to use statistical mechanism to process the shared data. The benefit of piggybacking is that it does not add new messages to the overlay.

An important use case for operating condition estimation is self-tuning of DHT parameters. In the case of the Chord DHT, the network size estimate is needed to select an appropriate size for the finger table, successor list, and predecessor list. For choosing an appropriate maintenance interval, all the three estimates, network size, join rate, and leave rate, are needed. In [138], we specify self-tuning extensions for the mandatory-to-implement Chord DHT of RELOAD. These extensions set the size of the finger table to $max(log_2N, 16)$ and the sizes of the predecessor list and the successor list to log_2N. The maintenance interval is set to

$$min(\frac{N}{2 \times \mu \times log_2^2N}, \frac{N}{\lambda \times log_2^2N}) \tag{4.1}$$

,

where N is the network size estimate, μ is the join rate estimate, and λ is the leave rate estimate.

When applying the mechanisms proposed in Publication IV, the network size estimate is accurate within 10.6% of the real network size. This estimate is accurate enough for the self-tuning use case. This is because a logarithm of the estimate is used in the equations that determine the size of the routing table and the maintenance interval. Thus, the effect of a 10.6% inaccuracy is very small. For the join rate and leave rate estimates, the mechanisms in Publication IV detect sudden decreases in the join and leave rates with a delay. However, in the self-tuning use case, this delay is not a problem since it does not compromise the stability of the overlay. This is because there is no harm in having a maintenance rate that is slightly higher than necessary during the period that it takes from the mechanisms to detect the sudden decrease. What is more important is the ability to detect sudden increases in the join and leave rates quickly. If not considering the periods when the join and leave rates suddenly drop to zero, the mechanisms proposed in Publication IV produce a join rate estimate that is accurate within 21.6% of the real join rate. The leave rate estimate is accurate within 25.0% of the real leave rate. The results in Publication IV show that the join rate estimate is very responsive to increases in the join rate. Therefore, the 21.6% average inaccuracy does not create a problem for the self-tuning use case. However, the leave rate estimate reacts to increases in the leave rate with a delay. Therefore, even if the 25% average inaccuracy of the leave rate estimate is in practice not an issue for the self-tuning use case, the slow responsiveness may be an issue if the leave rate increases very rapidly. One strategy for increasing the responsiveness is to adopt the maximum (or alternatively, 95th percentile) of the shared leave rate estimates as the estimate based on which the maintenance interval is determined. This strategy was found to improve the responsiveness considerably in the experiments of Publication IV, although it tends to overestimate the leave rate. Future work on operating condition estimation could focus especially on improving the responsiveness of leave rate estimation to rapid increases in the leave rate.

4.8 Service Discovery and Registration

As was discussed in 3.1.5, P2PSIP nodes need to be able to discover service providers from the overlay. Service discovery needs to be performed in a way that balances the load evenly among the service providers and among nodes storing the records of the service providers. Further, service discovery must not take an excessive amount of time and it must not cause a significant traffic load on the overlay. Achieving these goals is non-trivial in the case of popular services such as a TURN relay service, in which only 10% [51] of the nodes may be capable of providing the service and 100% of the node population needs to be able to discover a TURN relay service provider.

To enable more efficient service discovery in P2PSIP, the author of this dissertation defined a service discovery usage for RELOAD that uses ReDiR in a RELOAD overlay network in Publication VI. To the best of the author's knowledge, this was the first proposal to develop a generic service discovery mechanism for RELOAD. In addition, Publication VI was the first to analyze the performance of ReDiR in a RELOAD overlay network. It also developed a novel model to assist in configuring ReDiR. To the best of the author's knowledge, Publication VI was the first to propose such a model. The work in Publication VI will be summarized below.

In Publication VI, the performance of ReDiR in a P2PSIP overlay network was studied through simulations using the P2PSIP simulator written by the author of this dissertation. Three different network sizes, N=100, N=1000, and N=10000, were used. In the simulations, the impact of ReDiR's branching factor, starting level in the ReDiR tree, service provider density (i.e., the percentage of service providers out of total number of nodes in the overlay), and churn on ReDiR's performance was studied.

The results of the simulations indicate that the ReDiR branching factor b has a significant impact on delays, traffic load, and the load of answering ReDiR Get and Put requests. However, it has a smaller impact on the load of service providers and only a minor impact on the failure rate of service discovery and registration operations.

When it comes to the starting level, it was discovered in Publication VI that the adaptive starting level selection algorithm that ReDiR uses has a negative impact on performance and thus should be disabled if clients perform service lookups only infrequently, which is true in the TURN server discovery use case. The starting level has a considerable impact on ReDiR delays and failure rates. It also impacts the amount of ReDiR traffic and the distribution of the load of answering Get requests among peers storing the ReDiR tree nodes.

Also the size of the overlay network has an impact on ReDiR's performance, including delays, percentage of ReDiR traffic out of total traffic in the overlay, failed ReDiR operations, load of answering ReDiR requests, and the load of service providers. Regardless of the network size, branching factor, starting level and density of service providers, ReDiR fails to distribute the load evenly among service providers.

Service provider density impacts ReDiR's performance as well. Publication VI made the counter-intuitive finding that increasing the density does not always result in better performance; especially very high densities result in reduced performance. This is because when the service provider density is high, the ReDiR tree becomes so densely populated at the starting level and levels close to it that the average length of the upward and downward walks in the ReDiR tree increases. In addition, when the density is low and a high number of service lookups are being carried out, the peers storing ReDiR tree nodes can experience a very high load.

The main impact that churn has on ReDiR performance is increased failure rate of ReDiR operations. Churn also impacts the distribution of Get load and the load of service providers. This is because records stored by leaving peers and clients served by leaving service providers are transferred to some other node in the overlay, which on the average doubles the load of that node.

The main conclusion of the experiments is that ReDiR is fairly difficult to configure. If configured inappropriately, ReDiR can cause a high Get, Put, and storage load on tree nodes at the lowest levels of the ReDiR tree. If the number of service providers is low, the cost of being responsible for a tree node can be high for a peer. The main ReDiR configuration parameters are the branching factor b and the starting level l_{start}. The values of these parameters should be selected so that they provide a good match with the current network size and the density of service providers. Since these values tend to vary over time, it becomes important to be able to adapt the values of b and l_{start}.

To address the issue with selecting optimal values for b and l_{start}, a novel adaptive mechanism was developed in Publication VI. As the first step, the mechanism uses an algorithm that produces an estimate of the number of service providers present in the overlay by observing the average number of service provider records at the starting level of the ReDiR tree. This algorithm was shown to be able to produce an estimate that is accurate within 9% of the real number of service providers.

Once the number of service providers has been estimated, the next step is to determine the optimal values for b and l_{start}. The results of the simulations carried out for Publication VI pointed out two important findings. First, the delay of service lookups is minimized when the majority of service lookups finish at the starting level. Second, to achieve a good distribution of load among peers processing the ReDiR Get requests, one should aim at maximizing the number of service registration operations that do not perform an upward walk in the ReDiR tree. Based on these two findings, Publication VI devised an algorithm that can be used to select b and l_{start} values that maximize the probability that service lookups finish at the starting level and that registration operations do not have an upward walk.

The proposed adaptive mechanism for configuring ReDiR was validated through simulations. The results of the simulations show that the mechanism can accurately determine the best b and l_{start} combination for a given number of service providers.

As a summary, the work in Publication VI contributed to the understanding of the performance of ReDiR in P2PSIP overlays. It also pointed out previously unknown shortcomings in ReDiR. To address the most important shortcoming, that is, the difficulty of configuring ReDiR, a novel mechanism that estimates the number of service providers present in the overlay and configures ReDiR to achieve short service lookup delays and good distribution of load among the peers storing the ReDiR tree was developed. Future work on ReDiR is needed to balance load better among the service providers. One way to achieve better load balance is to remove the records of service providers experiencing a high load from the ReDiR tree and insert the records back when there is capacity available again. This mechanism avoids highly loaded service providers from becoming even more loaded and forces other service providers to contribute their resources instead.

4.9 P2PSIP in Mobile Networks and Devices

As was discussed in Section 3.4.6, mobile networks and devices pose special challenges for P2P applications. To gain an understanding of the performance of RELOAD in such environments, measurements were carried out in Publication III in which a mobile phone was participating in a P2PSIP overlay. The focus was on determining whether mobile terminals can participate as full peers (in contrast to being clients) in a RELOAD overlay. Several metrics were analyzed, including memory usage, CPU load, battery consumption, communication delays, and bandwidth usage. To the best of the author's knowledge, Publication III was the first study to investigate the performance of RELOAD on real mobile networks and devices.

In the measurements, a mobile phone participating as a full peer in a 10000-peer RELOAD overlay network was monitored. Peers other than the mobile peers were simulated peers running on a server. The mean session time of peers was eight hours. The P2PSIP application running on the mobile phone was implemented using the Java Micro Edition (J2ME). The phones were connected to the Internet using a 3G HSDPA connection with 2048 kbit/s downlink and 384 kbit/s uplink bandwidth.

The results in Publication III indicate that the memory usage of a J2ME application using RELOAD does not pose a problem even for low-end mobile phones; the Java heap size was found to vary between 547 and 767 kB. This figure includes all the memory used, including the application itself, the routing table, resource records stored on the phone by other peers, and so forth. It represents only a small fraction of the total memory available on low-end mobile phones that were available at the time when the measurements were carried out.

Also the CPU load caused by the RELOAD application was measured and compared to other mobile applications. The average CPU load that the RELOAD application causes is 43% higher than for instance the load caused by a non-P2P mobile instant messaging client. The main reason for the higher CPU load was observed to be the cryptographic operations (e.g., verification of certificates and signatures, and generation of signatures) that RELOAD performs. Even if the CPU load is higher than for applications like simple games and mobile instant messaging clients, it is still acceptable if looking only at the average CPU load, which was 25.7% in the measurements. However, the variance of the CPU load is very high compared to other mobile applications. This is caused by frequent periods during which the CPU load reaches levels close to or equal to 100%. These peaks are associated with the reception of RELOAD messages and the cryptographic operations that their processing requires. The frequent peaks cause for instance the 85th percentile CPU load to be as high as 94%.

The battery consumption of RELOAD was observed to be rather high; the battery of a mobile device participating as a full peer in a RELOAD overlay was drained in less than five hours. This implies high cost for a mobile phone to act as a full peer. The reason for the high battery usage is that RELOAD messages are exchanged so frequently over the radio interface that the phone has very limited or no opportunities at all to transfer to a low-power state. Instead, the network has to keep a dedicated radio channel allocated for the phone in practice all of the time, which results in maximum battery consumption.

The average size of RELOAD messages was observed to be 819 bytes. The dominant component of RELOAD messages is the security block, which contains certificates and signatures. The exchange of certificates and signatures constitutes 84.1% of the traffic exchanged in the overlay.

During a one-hour period, the mobile phone sent and received 1.1 MB of RELOAD traffic. Thus, the amount of traffic does not appear to constitute a problem considering the bitrates offered by 3G and later cellular radio technologies.

Besides increasing the CPU load, cryptographic operations also increase messaging delays. This is because the wall-clock time associated with signing a RELOAD message was observed to be on the average 1.6s on a mobile phone. The cost of verifying a signature was 0.2s. Thus, one key finding is that RELOAD's cryptographic operations represent a significant cost when it comes to the traffic load, CPU load, and messaging delays. This cost has not been taken into account in previous work, in which security features have not been implemented.

The presence of mobile peers in a RELOAD overlay has a dramatic impact on RELOAD delays. The presence of even a single mobile peer in the routing path of a message can render the delays multiple times larger compared to a case when all of the peers use broadband fixed access. In a mobile-only overlay, the delays are even more dramatic. The results in Publication III indicate that the cost of a DHT lookup in a mobile-only 1000-peer overlay is 14 times higher than in a RELOAD overlay consisting only of peers running on PCs using fixed access.

To summarize, the work in Publication III made contributions to the understanding of the performance of RELOAD on mobile phones and networks, especially when it comes to the cost of cryptographic operations, identification of performance bottlenecks and non-bottlenecks on mobile phones, and the impact that the presence of mobile peers has on the routing cost.

4.10 Distributed M2M Communication

The work that has been described in the previous sections has focused on decentralizing a real-time person-to-person communication service. The only protocol that has been run on top of the overlay network layer of the framework described in Chapter 4.1 has so far been SIP. This section will describe how to add another protocol, CoAP, to the framework. Including CoAP makes it possible to use the framework to implement a distributed M2M communication service. An important motivation for applying distributed algorithms to M2M communication is to be able to cope with the exponential growth in the number of connected devices that is associated with the vision of the Internet of Things. To be able to meet this growth, there is a need for new architectures that are highly scalable and have self-* properties, such as being self-organizing, self-configuring, and self-scalable.

This section will summarize our work on the CoAP usage for RELOAD that we designed in order to be able to run CoAP on top of RELOAD and thereby implement a decentralized M2M architecture. We originally presented the idea of the CoAP usage for RELOAD in Publication VII and subsequently proposed it in the IETF in [103]. To the best of the author's knowledge, Publication VII was the first to propose the combination of the CoAP and RELOAD protocols. In addition to defining the CoAP usage for RELOAD, also a decentralized M2M communication architecture was built around it in Publication VII. The main use case that Publication VII considers is decentralized wide area sensor and actuator networking. As a part of the work, the architecture was implemented and its performance evaluated through simulations and measurements on real hardware and networks.

The CoAP usage for RELOAD provides four basic functions. First, it makes it possible to register CoAP resources in and retrieve them from a RELOAD overlay. Second, it enables rendezvous between CoAP nodes using dedicated ICE negotiated connections for CoAP. Third, it provides an alternative rendezvous mechanism in which CoAP messages are tunneled across the RELOAD overlay. Fourth, it makes it possible to use the RELOAD overlay as a cache for sensor data.

The proposed architecture for wide area sensor and actuator networking is illustrated in Figure 4.5. In the architecture, three different types of nodes, Proxy Nodes (PNs), Wide-area Nodes (WNs), and Gateway Nodes (GWs) create a RELOAD overlay network. All these nodes are devices equipped with cellular radio modules. They have one or more sensors or actuators attached to them. The sensors are used for monitoring the environment, whereas the actuators are used for carrying out actions based on analysis of the sensor data. The nodes use CoAP for M2M communication. The RELOAD overlay is used as a rendezvous, storage, and NAT traversal mechanism for CoAP through the operations provided by the CoAP usage for RELOAD. The difference between WNs and PNs, both of which have a cellular radio interface, is that PNs have also a WSN radio interface through which they participate in a local WSN. The nodes in the WSNs are referred to as Local Nodes (LNs). Since the PNs have

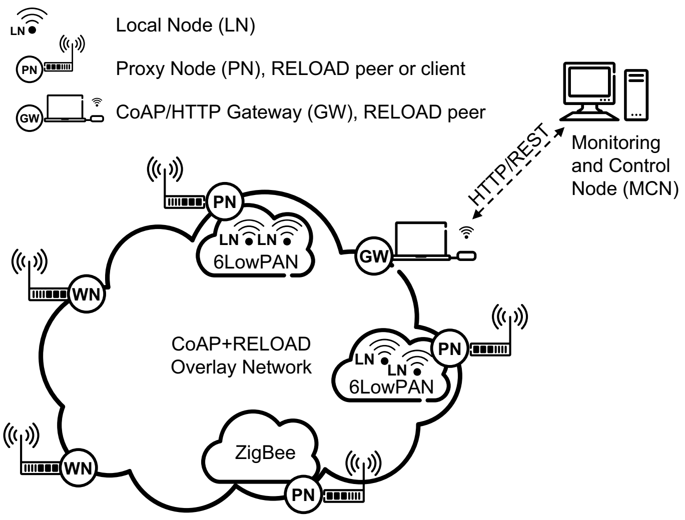

Local Node (LN)
Proxy Node (PN), RELOAD peer or client
CoAP/HTTP Gateway (GW), RELOAD peer
Monitoring and Control Node (MCN)
HTTP/REST
PN
LN
6LowPAN
GW
WN
CoAP+RELOAD Overlay Network
ZigBee

Figure 4.5. Decentralized M2M architecture

both cellular and WSN radio interfaces, and since they are part of the RELOAD overlay network, they can interconnect the CoAP nodes in the LNs with all the other CoAP nodes in the system. Therefore, the RELOAD overlay provides a P2P federation of the separate WSN islands. A GW node, in addition to acting as a peer in the RELOAD overlay, also acts as HTTP/CoAP proxy [39]. Thus, the GW node can provide web applications an HTTP REST interface to the resources in the WSNs interconnected by the RELOAD overlay network. The REST interface can be used for instance by Monitoring and Control Nodes (MCNs) to manipulate the resources in the system.

The proposed architecture was evaluated through simulations. For the simulations, a version of the P2PSIP simulator that included a CoAP implementation was used. The simulated overlay consisted of 2000-10000 PNs, each of which was connected to a WSN with 1-100 LNs. The PNs were assumed to use 3G High-Speed Packet Access (HSPA) access. The simulation assumed a road traffic and road condition monitoring use case, in which WSNs are deployed to monitor local conditions in the Finnish highway network. Each WSN has a PN that participates in the RELOAD overlay. The actuators in a given WSN use not only local information, but also information from other nearby WSNs as input for taking actions. For this, the actuators establish CoAP observation relationships with sensors in other WSNs.

In the simulations, the performance of a RELOAD-based M2M communication architecture was compared to a traditional Client/Server (C/S) architecture, in which all PNs in the local WSNs connect to a central data center. Also the use of dedicated ICE-negotiated connections for CoAP was compared to the use of tunneling of CoAP messages across the overlay. This results in four different scenarios. In scenario (1), called *RELOAD-dedicated*, all the PNs are part of a RELOAD overlay and dedicated ICE-negotiated connections are used for CoAP observation relationships. In scenario (2), called *RELOAD-tunnel*, no dedicated connections are set up for CoAP observation relationships. Instead, all notifications from sensors to their observers are tunneled across the RELOAD overlay in the payload of RELOAD messages. In scenario (3), called *C/S-dedicated*, there is no RELOAD overlay. Instead, a star topology in which all PNs communicate with a central server is used. However, there is still a P2P aspect present as dedicated CoAP observation relationships are established in a P2P manner directly between the sensors and their observers. In scenario (4), *C/S-tunnel*, no dedicated connections are used for CoAP. Instead, all traffic, including CoAP notifications, are sent via the central server.

The results of the simulations show that when dedicated connections are used for CoAP observation relationships, the communication delays of client/server and decentralized systems are on the same level. The exception is the one-time cost associated with establishing a CoAP observation relationship, which is more expensive in the decentralized system due to the need to route RELOAD Fetch and Attach message across the overlay. However, since this is a one-time cost, most use cases should be able to tolerate it. Therefore, in practice, when it comes to CoAP delays, the use of a RELOAD overlay is no more expensive than the use of a central server when dedicated connections are used for CoAP.

When CoAP messages are tunneled, which is an appropriate strategy only when the volume of inter-device communication is low, the delays associated with the decentralized architecture are higher. This is due to the delay of sending CoAP messages over multiple hops across the overlay. Thus, although tunneling CoAP across the overlay is an efficient strategy when sending CoAP data rather infrequently to a remote node, it is not suitable for real-time use cases where the delay between the sender and the receiver needs to be minimized.

When it comes to the traffic load, a client/server architecture generates less traffic when the network is small and the volume of inter-device communication (i.e., CoAP messaging) is low. A client/server system also places a lower load on the Radio Access Network (RAN). This is because in a P2P system, each message goes through the RAN twice, in the RANs of the sending and receiving peers. However, a client/server system scales poorly as the network becomes larger or the volume of inter-device communication increases. This is because the traffic load that the central server needs to handle grows rapidly to levels that need considerable investments in capacity. In contrast, in a decentralized system, the load of the PNs remains low compared to the maximum bitrate that the PN has available even when the network is large and CoAP notifications are sent frequently. An interesting observation is that in all of the scenarios studied, the largest source of traffic in the overlay is STUN keepalive signaling.

Which architecture to recommend depends on various factors. The advantages of the decentralized system include self-organization, low capex and opex, robustness, and scalability. If these advantages are not important for the use case, the size of the system is small, the frequency of CoAP messaging low, or if there is a need to minimize RAN load, then a client/server architecture is a good choice. However, if the above-mentioned advantages are important or as the frequency of CoAP messaging, the size of the system, or the number of CoAP observation relationships grows, a P2P architecture becomes quickly more recommendable.

As a part of the work on Publication VII, also a proof-of-concept prototype of the proposed architecture was built and tested on real networks and hardware. In the prototype, small single-board computers with embedded Linux are used as the PNs. The PNs have both 3G and ZigBee radio interfaces. The PNs connect sensors in a ZigBee WSN to the RELOAD overlay. The prototype uses the same code base as the simulator. This is possible since the PNs run the CACAO Java Virtual Machine for ARM processors. In the experiments, the PNs were participating in a 1000-node RELOAD overlay running in PlanetLab. A set of measurements were run using this setup, focusing on the end-to-end delay between two LNs located in different ZigBee WSNs. The most important finding of these measurements was that communication over ZigBee represents only roughly 10% of the total end-to-end delay; the rest of the delay comes from the communication between the PNs across the overlay.

To summarize, in Publication VII, a new decentralized architecture for M2M networks that is based on a novel CoAP usage for RELOAD was presented. The architecture was implemented both in a simulator and as a prototype running on real networks and hardware. The findings indicate that compared to a client/server architecture, the proposed architecture has clearly better scalability, and that the delays in the architecture are on the same level as in a client/server system when dedicated connections are used for inter-device communication. The types of M2M systems that benefit most from the architecture are large-scale networks having from moderate to high levels of inter-device communication. Future work on decentralized M2M communication could focus for instance on eliminating the need for keepalive signaling through the use of CoAP and RELOAD ALGs.

5. Conclusions

This dissertation has presented work on P2PSIP and the set of technologies that it uses, including DHT algorithms, P2P signaling protocols, NAT traversal mechanisms, and application protocols run on top of P2PSIP. P2PSIP is a set of protocols that enables use cases such as decentralized real-time communication. P2PSIP is based on emerging Internet standards. P2PSIP represents an attractive platform for providing communication services since it shares all the benefits of P2P systems, including scalability, robustness, low capex and opex, and self-organization. However, there are a number of challenges that hinder the wide-scale deployment of P2PSIP. The publications of this dissertation identify some of these challenges and address them by adding new components to the P2PSIP architecture. The result of this work, and the overall contribution of the dissertation, is a framework architecture for decentralized communications that fulfills the requirements that were stated out in Section 2.2 including adaptivity, scalability, generality, modularity, use of emerging standards, and high performance.

The approach that has been followed throughout the work on this dissertation has been to analyze the performance of the implementation of the framework, identify its bottlenecks, and finally design new components to address the identified bottlenecks. The publications of this dissertation have analyzed the impact of churn and DHT maintenance on the performance of P2PSIP, the cost of P2PSIP operations, the impact of NAT traversal on P2PSIP, and the performance of P2PSIP in mobile environments. The new components that were added to the framework include self-tuning, service discovery, M2M communication, and improved session setup mechanisms.

The work on DHT maintenance operations and the impact of churn presented in this dissertation highlighted the need for self-tuning mechanisms that can automatically adjust the parameters of the DHT to changing operating conditions. To support self-tuning behavior, mechanisms to estimate the operating conditions of a running P2PSIP overlay were designed. These include mechanisms to estimate the join rate, leave rate, and the size of the overlay network. It was shown that these mechanisms can clearly outperform existing mechanisms.

The cost of P2PSIP operations were analyzed thoroughly. These costs include the delays associated with joining an overlay, leaving an overlay, and performing lookups and setting up sessions in the overlay. Also the impact of NAT traversal on P2PSIP session setup delays was analyzed. The results indicate that in a P2PSIP overlay, session setup delays can be unacceptably high. To address this problem, mechanisms that can bring the session setup delays down to acceptable levels were designed. These mechanisms include tunneling of signaling data across the overlay, eliminating one of the two ICE negotiations that P2PSIP session setup requires, data compression, and ICE optimizations. When applied together, the mechanisms can reduce the session setup delay by up to two thirds down to levels that are in line with ITU-T recommendations for call setup delays.

A ReDiR-based service discovery component was added to the framework. This component addresses the issue of providing and locating services in a scalable manner in a P2PSIP overlay. The performance of the component was analyzed in a P2PSIP overlay in a TURN server discovery use case. The main result of the analysis was that ReDiR is difficult to configure. To address this issue, mechanisms to estimate the number of service providers in a RELOAD overlay and to select optimal parameters for ReDiR were developed. The performance of these mechanisms was validated, showing that they can accurately determine optimal ReDiR parameters.

The framework was applied to use cases beyond real-time person-to-person communication. The CoAP usage for RELOAD that was designed as a part of the work on this dissertation makes it possible to build a distributed M2M architecture. Such architecture provides a P2P federation of geographically distributed WSN islands. The architecture supports the registration of CoAP resources in a RELOAD overlay, rendezvous between CoAP endpoints through either tunneling across the overlay or the use of dedicated connections, and use of the overlay as a cache for sensor data. The architecture enables autonomous sensor and actuator networks in which intelligence is located in the devices themselves rather than in central data centers. The architecture was implemented and its performance was analyzed both in a simulator and on real hardware and networks. It was shown that the architecture has better scalability than a client/server architecture and that it can achieve delays that are comparable to delays in a client/server system.

The results that were obtained during the work on this dissertation were contributed to P2PSIP standardization. This included specifying a self-tuning extension for the variant of the Chord DHT that RELOAD uses. Also a scalable service discovery extension for RELOAD was specified. These two contributions have the potential to enable the use of RELOAD in large-scale dynamic overlays. Also the work on the CoAP usage for RELOAD was contributed to P2PSIP standardization. This work has the potential to enable the creation of fully decentralized autonomous M2M systems.

The results presented in this dissertation show that the framework can scale from small-scale person-to-person communication use cases to large-scale M2M networks with hundreds of thousands of interconnected devices. The framework can also dynamically adapt itself to changing conditions through self-tuning. This makes the framework robust against churn and even considerable changes in the size of the system. The framework can be implemented on heterogeneous devices and networks. It can also fulfill the needs of very different applications built on top of it, ranging from real-time multimedia communication to the Internet of Things. The framework achieves maximum interoperability through the use of emerging and existing Internet standards. The framework has been designed to be modular and extensible. As a part of the work on this dissertation, a number of new components were incorporated to the framework, providing support for service discovery, service registration, operating condition estimation, self-tuning, M2M communication, data tunneling, and session setup delay optimization. Finally, it has been demonstrated that the framework can, when using mechanisms such as the optimized session setup component, meet the strict performance requirements of applications, including especially the requirements of real-time communication services.

5.1 Future Work

This dissertation presented a framework architecture for decentralized communications. One of the design requirements for the framework was modularity, which makes it easy to extend the framework with new components. One example of a component that could be added to the framework as future work is a distributed event subscription and notification mechanism. Such a mechanism allows users to subscribe to the status of resource-IDs and resources in the overlay network and receive notifications when the resource is created, deleted, or modified. Distributed event subscription and notification would make it possible to implement for instance a scalable presence service in a P2PSIP overlay.

The area of decentralized M2M networks has potential for further research. One interesting challenge is the design of energy-efficient, lightweight, structured overlay algorithms and P2P signaling protocols that are optimized for M2M environments. Another topic warranting future research is alternative approaches to NAT traversal in cellular M2M networks. The expected growth in the number of connected devices in cellular networks may justify the deployment of ALGs that can support M2M protocols such as the CoAP usage for RELOAD.

Despite of several years of research, fully distributed security in P2P networks is still an ongoing research problem. Due to this reason, present-day P2P systems rely on centralized components to implement security. To achieve fully distributed security, further research is needed in areas such as secure distributed assignment of node-IDs, distributed trust among peers, and dealing with a large number of malicious or compromised peers.

This dissertation has among other things described how to implement scalable service discovery for RELOAD. Being able to discover services from and register services in the overlay is only one part of the puzzle. Another part is the ability to efficiently balance the load among peers providing highly popular services such as a TURN relay service. Future work could focus on service discovery optimizations that pay special attention to balancing the load between service providers.

A further challenge for P2P systems is to be able to deal with the problems of *free-riding* and *tragedy of the commons*. Users choosing to free-ride use the services of the P2P system without contributing back any of their own resources. The tragedy of the commons is a dilemma in which users deplete a shared limited resource (e.g., a node responsible for a popular resource or service) even if it is in no-one's long-term interest to do so. To alleviate these problems, future research on incentive mechanisms that are specific to P2PSIP networks is necessary.

Lawful intercept represents an open research problem for decentralized real-time communication systems. Lack of a solution to support lawful intercept in P2PSIP telephony networks has the potential to limit the use cases in which P2PSIP can be deployed. As an example, regulators might choose to impose lawful intercept requirements on public P2PSIP telephony services, thus making their introduction difficult in the absence of decentralized mechanisms for lawful intercept.

A recent standardization activity that has potential to disrupt existing real-time communication services or at least extend their reach to new contexts is Web Real-Time Communication (WebRTC). WebRTC relies on JavaScript APIs for browsers developed by the Word Wide Web Consortium (W3C) and real-time communication protocols designed by the IETF. One interesting additional use case for the framework would be an overlay network topology built using WebRTC-powered P2P data channels between browsers. Such a use case would require looking into possibilities for implementing the framework using JavaScript and the fifth revision of the Hyper-Text Markup Language standard (HTML5).

Bibliography

[1] 3GPP. 2012. IP Multimedia Subsystem (IMS); Stage 2. TS 23.228, 3rd Generation Partnership Project (3GPP).

[2] A.-B. Al-Mamou and H. Labiod. 2007. ScatterPastry: An Overlay Routing Using a DHT over Wireless Sensor Networks. In: Proceedings of the 2007 International Conference on Intelligent Pervasive Computing (IPC), pages 274–279.

[3] M. Ali and Z. A. Uzmi. 2004. CSN: A Network Protocol for Serving Dynamic Queries in Large-Scale Wireless Sensor Networks. In: Proceedings of the Second Annual Conference on Communication Networks and Services Research, pages 165–174.

[4] L. O. Alima, A. Ghodsi, and S. Haridi. 2005. A Framework for Structured Peer-to-Peer Overlay Networks. In: Global Computing, volume 3267 of *Lecture Notes in Computer Science*, pages 223–249. Springer. ISBN 3-540-24101-9.

[5] J. Aspnes and G. Shah. 2003. Skip Graphs. In: Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms, SODA '03, pages 384–393.

[6] B. Athwal, F. C. Harmatzis, and V. P. Tanguturi. 2006. Replacing Centric Voice Services with Hosted VoIP Services: an Application of Real Options Approach. In: Proceedings of the 16th International Telecommunications Society (ITS) European regional conference, pages 37–38.

[7] F. Audet and C. Jennings. 2007. Network Address Translation (NAT) Behavioral Requirements for Unicast UDP. RFC 4787, Internet Engineering Task Force.

[8] S. Avancha, A. Joshi, and T. Finin. 2002. Enhanced Service Discovery in Bluetooth. IEEE Transactions on Computers 35, no. 6, pages 96–99.

[9] A. Awad, C. Sommer, R. German, and F. Dressler. 2008. Virtual Cord Protocol (VCP): A Flexible DHT-like Routing Service for Sensor Networks. In: Proceedings of the 5th IEEE International Conference on Mobile Ad Hoc and Sensor Systems (MASS), pages 133–142.

[10] Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal. 1994. Balanced Allocations (Extended Abstract). In: Proceedings of the 26th Annual ACM Symposium on Theory of Computing (STOC), pages 593–602.

[11] M. Baldi, L. De Marco, F. Risso, and L. Torrero. 2008. Providing End-to-End Connectivity to SIP User Agents Behind NATs. In: Proceedings of the 2008 IEEE International Conference on Communications (ICC), pages 5902–5908.

[12] M. Barnes, C. Jennings, J. Rosenberg, and M. Petit-Huguenin. 2013. Verification Involving PSTN Reachability: Requirements and Architecture Overview. Internet-Draft draft-jennings-vipr-overview-04, Internet Engineering Task Force. Work in Progress.

[13] S. A. Baset and H. G. Schulzrinne. 2006. An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol. In: Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM), pages 1–11.

[14] S. Baset, H. Schulzrinne, and M. Matuszewski. 2007. Peer-to-Peer Protocol (P2PP). Internet-Draft draft-baset-p2psip-p2pp-01, Internet Engineering Task Force. Work in Progress.

[15] S. A. Baset and H. Schulzrinne. 2010. Reliability and Relay Selection in Peer-to-Peer Communication Systems. In: Proceedings of the 2010 International Conference on Principles, Systems and Applications of IP Telecommunications (IPTComm), pages 111–121.

[16] I. Baumgart, B. Heep, and S. Krause. 2007. OverSim: A Flexible Overlay Network Simulation Framework. In: Proceedings of the 2007 IEEE Global Internet Symposium, pages 79–84.

[17] M. Bawa, H. Garcia-Molina, A. Gionis, and R. Motwani. 2003. Estimating Aggregates on a Peer-to-Peer Network. Technical Report 2003-24, Stanford InfoLab.

[18] M. Bienkowski, M. Korzeniowski, and F. Meyer auf der Heide. 2005. Dynamic Load Balancing in Distributed Hash Tables. In: Proceedings of the 4th International Workshop on Peer-to-Peer Systems (IPTPS), pages 217–225.

[19] A. Binzenhöfer, G. Kunzmann, and R. Henjes. 2006. A Scalable Algorithm to Monitor Chord-based P2P Systems at Runtime. In: Proceedings of the 20th International Conference on Parallel and Distributed Processing (IPDPS), pages 375–375.

[20] A. Binzenhöfer and K. Leibnitz. 2007. Estimating Churn in Structured P2P Networks. In: Proceedings of the 20th International Teletraffic Conference on Managing Traffic Performance in Converged Networks (ITC), pages 630–641.

[21] A. Binzenhöfer, D. Staehle, and R. Henjes. 2005. On the Fly Estimation of the Peer Population in a Chord-based P2P System. In: Proceedings of the 19th International Teletraffic Congress (ITC), pages 1827–1936.

[22] A. Binzenhöfer and P. Tran-Gia. 2004. Delay Analysis of a Chord-based Peer-to-Peer File-Sharing System. Technical Report 332, University of Würzburg.

[23] A. Binzenhöfer, K. Tutschku, B. Graben, M. Fiedler, and P. Arlos. 2006. A P2P-Based Framework for Distributed Network Management. In: Wireless Systems and Network Architectures in Next Generation Internet, volume 3883 of *Lecture Notes in Computer Science*, pages 198–210. Springer. ISBN 978-3-540-34025-6.

[24] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau. 2004. Extensible Markup Language (XML) 1.0 (Third Edition). W3C Recommendation, W3C.

[25] D. Bryan, B. Lowekamp, and C. Jennings. 2005. SOSIMPLE: A Serverless, Standards-based, P2P SIP Communication System. In: Proceedings of the 1st International Workshop on Advanced Architectures and Algorithms for Internet Delivery and Applications (AAA-IDEA), pages 42–49.

[26] D. Bryan, B. Lowekamp, and M. Zangrilli. 2008. The Design of a Versatile, Secure P2PSIP Communications Architecture for the Public Internet. In: Proceedings of the 22nd IEEE International Symposium on Parallel and Distributed Processing (IPDPS), pages 1–8.

[27] D. Bryan, P. Matthews, E. Shim, D. Willis, and S. Dawkins. 2011. Concepts and Terminology for Peer to Peer SIP. Internet-Draft draft-ietf-p2psip-concepts-04, Internet Engineering Task Force. Work in Progress.

[28] D. Bryan, E. Shim, and B. Lowekamp. 2007. Use Cases for Peer-to-Peer Session Initiation Protocol (P2P SIP). Internet-Draft draft-bryan-p2psip-usecases-00, Internet Engineering Task Force. Work in Progress.

[29] J. F. Buford and M. Kolberg. 2012. Application Layer Multicast Extensions to RELOAD. Internet-Draft draft-kolberg-sam-baseline-protocol-01, Internet Engineering Task Force. Work in Progress.

[30] J. Buford, H. Yu, and E. K. Lua. 2008. P2P Networking and Applications. Morgan Kaufmann. ISBN 978-0071373401.

[31] J. Byers, J. Considine, and M. Mitzenmacher. 2003. Simple Load Balancing for Distributed Hash Tables. In: Peer-to-Peer Systems II, volume 2735 of *Lecture Notes in Computer Science*, pages 80–87. Springer. ISBN 978-3-540-40724-9.

[32] M. Caesar, M. Castro, E. B. Nightingale, G. O'Shea, and A. Rowstron. 2006. Virtual Ring Routing: Network Routing Inspired by DHTs. In: Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (ACM SIGCOMM), pages 351–362.

[33] G. Camarillo. 2003. Compressing the Session Initiation Protocol (SIP). RFC 3486, Internet Engineering Task Force.

[34] G. Camarillo. 2009. Peer-to-Peer (P2P) Architecture: Definition, Taxonomies, Examples, and Applicability. RFC 5694, Internet Engineering Task Force.

[35] G. Camarillo, J. Mäenpää, A. Keränen, and V. Andersson. 2011. Reducing Delays Related to NAT Traversal in P2PSIP Session Establishments. In: Proceedings of the 2011 IEEE Consumer Communications and Networking Conference (CCNC), pages 549–553.

[36] G. Camarillo, P. Nikander, J. Hautakorpi, A. Keranen, and A. Johnston. 2011. HIP BONE: Host Identity Protocol (HIP) Based Overlay Networking Environment (BONE). RFC 6079, Internet Engineering Task Force.

[37] B. Campbell, J. Rosenberg, H. Schulzrinne, C. Huitema, and D. Gurle. 2002. Session Initiation Protocol (SIP) Extension for Instant Messaging. RFC 3428, Internet Engineering Task Force.

[38] B. Carlsson and R. Gustavsson. 2001. The Rise and Fall of Napster - An Evolutionary Approach. In: Proceedings of the 6th International Computer Science Conference on Active Media Technology (AMT), pages 347–354.

[39] A. Castellani, S. Loreto, A. Rahman, T. Fossati, and E. Dijk. 2012. Best Practices for HTTP-CoAP Mapping Implementation. Internet-Draft draft-castellani-core-http-mapping-05, Internet Engineering Task Force. Work in Progress.

[40] M. Castro, P. Druschel, A. M. Kermarrec, and A. I. Rowstron. 2006. Scribe: a Large-scale and Decentralized Application-level Multicast Infrastructure. IEEE Journal on Selected Areas in Communications 20, no. 8, pages 1489–1499.

[41] M. Castro, M. Costa, and A. Rowstron. 2005. Debunking Some Myths about Structured and Unstructured Overlays. In: Proceedings of the 2nd USENIX Symposium on Networked Systems Design & Implementation (NSDI), pages 85–98.

[42] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. S. Wallach. 2002. Secure Routing for Structured Peer-to-Peer Overlay Networks. SIGOPS Operating Systems Review 36, no. SI, pages 299–314.

[43] M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron. 2002. One Ring to Rule Them All: Service Discovery and Binding in Structured Peer-to-Peer Overlay Networks. In: Proceedings of the 10th ACM SIGOPS European Workshop, pages 140–145.

[44] C.-M. Cheng, S.-L. Tsao, and J.-C. Chou. 2007. Unstructured Peer-to-Peer Session Initiation Protocol for Mobile Environment. In: Proceedings of the 18th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), pages 1–5.

[45] Y. Cheng, X. Wen, and W. Zheng. 2009. An Improved Lookup Algorithm for Chord. In: Proceedings of the 2009 Pacific-Asia Conference on Circuits, Communications and Systems (PACCS), pages 163–166.

[46] D. Chopra, H. Schulzrinne, E. Marocco, and E. Ivov. 2009. Peer-to-Peer Overlays for Real-time Communication: Security issues and Solutions. IEEE Communications Surveys Tutorials 11, no. 1, pages 4–12.

[47] Cisco. 2001. Traffic analysis for Voice over IP. White Paper. Cisco Systems. URL `http://tinyurl.com/8ng5qta`.

[48] E. Cooper, A. Johnston, and P. Matthews. 2007. A Distributed Transport Function in P2PSIP Using HIP for Multi-Hop Overlay Routing. Internet-Draft draft-matthews-p2psip-hip-hop-00, Internet Engineering Task Force. Work in Progress.

[49] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica. 2001. Wide-area Cooperative Storage with CFS. SIGOPS Operating Systems Review 35, no. 5, pages 202–215.

[50] F. Dabek, J. Li, E. Sit, J. Robertson, M. F. Kaashoek, and R. Morris. 2004. Designing a DHT for Low Latency and High Throughput. In: Proceedings of the 1st USENIX Symposium on Networked Systems Design and Implementation (NSDI), pages 85–98.

[51] L. D'Acunto, J. Pouwelse, and H. Sips. 2009. A Measurement of NAT & Firewall Characteristics in Peer-to-Peer Systems. In: Proceedings of the 15th Advanced School for Computing and Imaging (ASCI) Conference, pages 1–5.

[52] L. Daigle. 2002. IAB Considerations for UNilateral Self-Address Fixing (UNSAF) Across Network Address Translation. RFC 3424, Internet Engineering Task Force.

[53] V. Darlagiannis, N. Liebau, O. Heckmann, A. Mauthe, and R. Steinmetz. 2006. Caching Indices for Efficient Lookup in Structured Overlay Networks. In: Proceedings of the 4th International Conference on Agents and Peer-to-Peer Computing (AP2PC), pages 81–93.

[54] T. Dierks and E. Rescorla. 2006. The Transport Layer Security (TLS) Protocol Version 1.1. RFC 4346, Internet Engineering Task Force.

[55] J. R. Douceur. 2002. The Sybil Attack. In: Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS), pages 251–260. Springer-Verlag.

[56] J. Dowling, J. Sacha, and S. Haridi. 2007. Improving ICE Service Selection in a P2P System using the Gradient Topology. In: Proceedings of the 1st International Conference on Self-Adaptive and Self-Organizing Systems (SASO), pages 285–288.

[57] C. Du, H. Yin, C. Lin, and Y. Hu. 2008. VCNF: A Secure Video Conferencing System Based on P2P Technology. In: Proceedings of the 10th IEEE International Conference on High Performance Computing and Communications (HPCC), pages 463–469.

[58] D. Eastlake and P. Jones. 2001. US Secure Hash Algorithm 1 (SHA1). RFC 3174, Internet Engineering Task Force.

[59] S. El-Ansary, L. Alima, P. Brand, and S. Haridi. 2003. Efficient Broadcast in Structured P2P Networks. In: Peer-to-Peer Systems II, volume 2735 of *Lecture Notes in Computer Science*, pages 304–314. Springer. ISBN 978-3-540-40724-9.

[60] ETSI. 2001. Integrated Services Digital Network (ISDN); Signalling System No.7 (SS7); ISDN User Part (ISUP) version 4 for the International Interface; Part 1: Basic services. EN 300 356-1, European Telecommunications Standards Institute (ETSI).

[61] J. Falkner, M. Piatek, J. P. John, A. Krishnamurthy, and T. Anderson. 2007. Profiling a Million User DHT. In: Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement, pages 129–134.

[62] M. Feldman, C. Papadimitriou, J. Chuang, and I. Stoica. 2006. Free-riding and Whitewashing in Peer-to-Peer Systems. IEEE Journal on Selected Areas in Communications 24, no. 5, pages 1010–1019.

[63] M. Feldman and J. Chuang. 2005. Overcoming Free-riding Behavior in Peer-to-Peer Systems. ACM SIGecom Exchanges 5, no. 4, pages 41–50.

[64] G. Fersi, W. Louati, and M. Ben Jemaa. 2012. Distributed Hash Table-Based Routing and Data Management in Wireless Sensor Networks: A Survey. Wireless Networks pages 1–18.

[65] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. 1999. Hypertext Transfer Protocol – HTTP/1.1. RFC 2616, Internet Engineering Task Force.

[66] R. T. Fielding. 2000. Architectural Styles and the Design of Network-based Software Architectures. Ph.D. thesis. University of California.

[67] P. Ganesan, K. Gummadi, and H. Garcia-Molina. 2004. Canon in G Major: Designing DHTs with Hierarchical Structure. In: Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS), pages 263–272.

[68] M. Garcia-Martin, C. Bormann, J. Ott, R. Price, and A. B. Roach. 2003. The Session Initiation Protocol (SIP) and Session Description Protocol (SDP) Static Dictionary for Signaling Compression (SigComp). RFC 3485, Internet Engineering Task Force.

[69] G. Ghinita and Y. M. Teo. 2006. An Adaptive Stabilization Framework for Distributed Hash Tables. In: Proceedings of the 20th International Conference on Parallel and Distributed Processing (IPDPS), pages 29–38.

[70] B. Godfrey, K. Lakshminarayanan, S. Surana, R. Karp, and I. Stoica. 2004. Load Balancing in Dynamic Structured P2P Systems. In: Proceedings of the 23rd IEEE International Conference on Computer Communications (INFOCOM), pages 2253–2262.

[71] P. B. Godfrey, S. Shenker, and I. Stoica. 2006. Minimizing Churn in Distributed Systems. In: Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (ACM SIGCOMM), pages 147–158.

[72] P. Godfrey and I. Stoica. 2005. Heterogeneity and Load Balance in Distributed Hash Tables. In: Proceedings of the 24th IEEE International Conference on Computer Communications (INFOCOM), pages 596–606.

[73] Y. Gu, N. Zong, H. Zhang, Y. Zhang, F. Piccolo, and S. Duan. 2013. Survey of P2P Streaming Applications. Internet-Draft draft-ietf-ppsp-survey-04, Internet Engineering Task Force. Work in Progress.

[74] S. Guha, N. Daswani, and R. Jain. 2006. An Experimental Study of the Skype Peer-to-Peer VoIP System. In: Proceedings of the 5th International Workshop on Peer-to-Peer Systems (IPTPS), pages 677–686.

[75] D. Guinard and V. Trifa. 2009. Towards the Web of Things: Web Mashups for Embedded Devices. In: Proceedings of the 2nd Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM).

[76] K. Gummadi, R. Gummadi, S. Gribble, S. Ratnasamy, S. Shenker, and I. Stoica. 2003. The Impact of DHT Routing Geometry on Resilience and Proximity. In: Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (ACM SIGCOMM), pages 381–394.

[77] A. Gupta, B. Liskov, and R. Rodrigues. 2003. One Hop Lookups for Peer-to-Peer Overlays. In: Proceedings of the 9th Conference on Hot Topics in Operating Systems (HOTOS), pages 7–12.

[78] I. Gupta, K. Birman, P. Linga, A. Demers, and R. van Renesse. 2003. Kelips: Building an Efficient and Stable P2P DHT through Increased Memory and Background Overhead. In: Peer-to-Peer Systems II, volume 2735 of *Lecture Notes in Computer Science*, pages 160–169. Springer. ISBN 978-3-540-40724-9.

[79] E. Guttman, C. Perkins, J. Veizades, and M. Day. 1999. Service Location Protocol, Version 2. RFC 2608, Internet Engineering Task Force.

[80] M. Handley, V. Jacobson, and C. Perkins. 2006. SDP: Session Description Protocol. RFC 4566, Internet Engineering Task Force.

[81] H. Hannu, J. Christoffersson, S. Forsgren, K.-C. Leung, Z. Liu, and R. Price. 2003. Signaling Compression (SigComp) - Extended Operations. RFC 3321, Internet Engineering Task Force.

[82] E. Harjula, J. Ala-Kurikka, D. Howie, and M. Ylianttila. 2006. Analysis of Peer-to-Peer SIP in a Distributed Mobile Middleware System. In: Proceedings of the 2006 IEEE Global Telecommunications Conference (GLOBECOM), pages 1–6.

[83] E. Harjula, T. Koskela, and M. Ylianttila. 2011. Comparing the Performance and Efficiency of Two Popular DHTs in Interpersonal Communication. In: Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC), pages 2173–2178.

[84] E. Harjula, J. Hautakorpi, N. Beijar, and M. Ylianttila. 2009. Peer-to-Peer SIP for Mobile Computing: Challenges and Solutions. In: B. Seet (editor), Mobile Peer-To-Peer Computing for Next Generation Distributed Environments: Advancing Conceptual and Algorithmic Applications, pages 326–347. IGI Global. ISBN 978-1-60566-715-7.

[85] E. Harjula and M. Ylianttila. 2010. Load Balancing Models for DHT-based Peer-to-Peer Networks. Internet-Draft draft-harjula-p2psip-loadbalancing-survey-01, Internet Engineering Task Force. Work in Progress.

[86] E. Harjula, M. Ylianttila, J. Ala-Kurikka, J. Riekki, and J. Sauvola. 2004. Plug-and-Play Application Platform: towards Mobile Peer-to-Peer. In: Proceedings of the 3rd International Conference on Mobile and Ubiquitous Multimedia (MUM), pages 63–69.

[87] K. Hartke. 2013. Observing Resources in CoAP. Internet-Draft draft-ietf-core-observe-08, Internet Engineering Task Force. Work in Progress.

[88] J. Hautakorpi, A. Salinas, E. Harjula, and M. Ylianttila. 2008. Interconnecting P2PSIP and IMS. In: Proceedings of the 2nd International Conference on Next Generation Mobile Applications, Services and Technologies (NGMAST), pages 83–88.

[89] J. Hautakorpi and J. Mäenpää. 2010. Load Balancing for Structured P2P Networks Using the Advanced Finger Selection Algorithm (AFSA). In: Proceedings of the 2010 ACM Symposium on Applied Computing (SAC), pages 655–662.

[90] M. V. J. Heikkinen and S. Luukkainen. 2008. Technology Evolution of Mobile Peer-to-Peer Communications. In: Proceedings of the 4th Annual International Conference on Wireless Internet (WICON), pages 1–9.

[91] K. Horowitz and D. Malkhi. 2003. Estimating Network Size from Local Information. Information Processing Letters 88, no. 5, pages 237–243.

[92] D. Howie, E. Harjula, J. Ala-Kurikka, and M. Ylianttila. 2005. Harnessing SIP for Autonomous Mobile Peer-to-Peer Networking. In: Proceedings of the 2005 IEEE Global Telecommunications Conference (GLOBECOM), pages 1–5.

[93] Y. hua Chu, S. Rao, S. Seshan, and H. Zhang. 2002. A Case for End System Multicast. IEEE Journal on Selected Areas in Communications 20, no. 8, pages 1456–1471.

[94] C. Huitema. 2006. Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs). RFC 4380, Internet Engineering Task Force.

[95] IEEE. 2006. Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specification for Low Rate Wireless Personal Area Networks (LR-WPANs). IEEE Standard for Information Technology 802.15.4-2006, IEEE.

[96] ITU-T. 1999. Network Grade of Service Parameters and Target Values for Circuit-Switched Services in the Evolving ISDN. Recommendation E.721, International Telecommunication Union.

[97] J. Javornik, M. Volk, I. Humar, and A. Kos. 2009. Empirical Performance Evaluation of Peer-to-Peer VoIP Telephony Using SIP. In: Proceedings of the IEEE EUROCON 2009, pages 1838 –1843.

[98] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset, and H. Schulzrinne. 2013. REsource LOcation And Discovery (RELOAD) Base Protocol. Internet-Draft draft-ietf-p2psip-base-26, Internet Engineering Task Force. Work in Progress.

[99] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset, H. Schulzrinne, and T. Schmidt. 2013. A SIP Usage for RELOAD. Internet-Draft draft-ietf-p2psip-sip-09, Internet Engineering Task Force. Work in Progress.

[100] C. Jennings, J. Rosenberg, and E. Rescorla. 2007. Address Settlement by Peer to Peer. Internet-Draft draft-jennings-p2psip-asp-00, Internet Engineering Task Force. Work in Progress.

[101] J. Jiang, R. Pan, C. Liang, and W. Wang. 2005. BiChord: An Improved Approach for Lookup Routing in Chord. In: Advances in Databases and Information Systems, volume 3631 of *Lecture Notes in Computer Science*, pages 338–348. Springer. ISBN 978-3-540-28585-4.

[102] X. Jiang and H. Zheng. 2008. Service Extensible P2P Peer Protocol. Internet-Draft draft-jiang-p2psip-sep-01, Internet Engineering Task Force. Work in Progress.

[103] J. Jimenez, J. Lopez-Vega, J. Mäenpää, and G. Camarillo. 2013. A Constrained Application Protocol (CoAP) Usage for REsource LOcation And Discovery (RELOAD). Internet-draft draft-jimenez-p2psip-coap-reload-03, Internet Engineering Task Force. Work in Progress.

[104] D. R. Karger and M. Ruhl. 2004. Simple Efficient Load Balancing Algorithms for Peer-to-Peer Systems. In: Proceedings of the 16th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), pages 36–43.

[105] O. Kassinen, E. Harjula, J. Korhonen, and M. Ylianttila. 2009. Battery life of Mobile Peers with UMTS and WLAN in a Kademlia-based P2P Overlay. In: Proceedings of the 20th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, pages 662–665.

[106] O. Kassinen, E. Harjula, and M. Ylianttila. 2010. Analysis of Messaging Load in a P2PP Overlay Network under Churn. In: In Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM), pages 1–5.

[107] O. Kassinen, E. Harjula, and M. Ylianttila. 2009. Suitability of DHT-Based Peer-to-Peer Session Initiation Protocol for Wireless Distributed Services. In: Proceedings of the 12th International Symposium on Wireless Personal Multimedia Communications (WPMC), pages 1–5.

[108] O. Kassinen, T. Koskela, E. Harjula, J. Riekki, and M. Ylianttila. 2009. Analysis of Connectivity and Session Management for Mobile Peer-to-Peer Applications. Journal of Mobile Multimedia 5, no. 2, pages 81–112.

[109] O. Kassinen, Z. Ou, M. Ylianttila, and E. Harjula. 2008. Effects of Peer-to-Peer Overlay Parameters on Mobile Battery Duration and Resource Lookup Efficiency. In: Proceedings of the 7th International Conference on Mobile and Ubiquitous Multimedia (MUM), pages 177–180.

[110] I. Kelenyi and J. Nurminen. 2008. Energy Aspects of Peer Cooperation Measurements with a Mobile DHT System. In: Proceedings of the IEEE International Conference on Communications (ICC), pages 164–168.

[111] I. Kelenyi, J. Nurminen, and M. Matuszewski. 2010. DHT Performance for Peer-to-Peer SIP - A Mobile Phone Perspective. In: Proceedings of the 7th IEEE Consumer Communications and Networking Conference (CCNC), pages 1–5.

[112] K. Kenthapadi and G. S. Manku. 2005. Decentralized Algorithms Using both Local and Random Probes for P2P Load Balancing. In: Proceedings of the 17th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), pages 135–144.

[113] A. Keränen, G. Camarillo, and J. Mäenpää. 2012. Host Identity Protocol-Based Overlay Networking Environment (HIP BONE) Instance Specification for REsource LOcation And Discovery (RELOAD). Internet-Draft draft-ietf-hip-reload-instance-05, Internet Engineering Task Force. Work in Progress.

[114] S. Kim, X. Wang, H. Kim, T. Kwon, and Y. Choi. 2010. Measurement and Analysis of BitTorrent Traffic in Mobile WiMAX Networks. In: Proceedings of the 10th IEEE International Conference on Peer-to-Peer Computing (P2P), pages 1–4.

[115] R. Klauck and M. Kirsche. 2009. Integrating P2PSIP into Collaborative P2P Applications: A Case Study with the P2P Videoconferencing System BRAVIS. In: Proceedings of the 5th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), pages 1–10.

[116] A. Knauf, G. Hege, T. C. Schmidt, and M. Waehlisch. 2012. A RELOAD Usage for Distributed Conference Control (DisCo). Internet-Draft draft-ietf-p2psip-disco-00, Internet Engineering Task Force. Work in Progress.

[117] A. Knauf, G. Hege, T. C. Schmidt, and M. Waehlisch. 2012. A Usage for Shared Resources in RELOAD (ShaRe). Internet-Draft draft-ietf-p2psip-share-03, Internet Engineering Task Force.

[118] A. Knauf, G. Hege, T. C. Schmidt, and M. Wählisch. 2010. A Virtual and Distributed Control Layer with Proximity Awareness for Group Conferencing in P2PSIP. In: Proceedings of the 2010 International Conference on Principles, Systems and Applications of IP Telecommunications (IPTComm), pages 122–133.

[119] T. Kohonen. 1990. The Self-Organizing Map. Proceedings of the IEEE 78, no. 9, pages 1464–1480.

[120] E. Kokkonen, S. Baset, and M. Matuszewski. 2008. Demonstration of Peer-to-Peer Session Initiation Protocol (P2PSIP) in the Mobile Environment. In: Proceedings of the 5th IEEE Consumer Communications and Networking Conference (CCNC), pages 1221–1222.

[121] J. Koskela. 2008. A HIP-based Peer-to-Peer Communication System. In: Proceedings of the International Conference on Telecommunications (ICT), pages 1–7.

[122] J. Koskela, J. Heikkilä, and A. Gurtov. 2010. A Secure P2P SIP System with SPAM Prevention. SIGMOBILE Mobile Computer Communication Review 13, no. 3, pages 26–29.

[123] T. Koskela, E. Harjula, O. Kassinen, and M. Ylianttila. 2011. Robustness of a P2P Community Management System based on Two-level Hierarchical DHT Overlays. In: Proceedings of the IEEE Symposium on Computers and Communications (ISCC), pages 881–886.

[124] D. Kostoulas, D. Psaltoulis, I. Gupta, K. Birman, and A. Demers. 2005. Decentralized Schemes for Size Estimation in Large and Dynamic Groups. In: Proceedings of the 4th IEEE International Symposium on Network Computing and Applications (NCA), pages 41–48.

[125] G. Lawton. 2004. Machine-to-Machine Technology Gears up for Growth. IEEE Computer 37, no. 9, pages 12–15.

[126] H. Le, T. Duong, and Y. Kim. 2012. Presence Information Distribution in the P2P Overlay Using ALTO Service. In: Convergence and Hybrid Information Technology, volume 7425 of *Lecture Notes in Computer Science*, pages 698–707. Springer. ISBN 978-3-642-32644-8.

[127] E. Le Merrer, A. M. Kermarrec, and L. Massoulie. 2006. Peer to Peer Size Estimation in Large and Dynamic Networks: A Comparative Study. In: Proceedings of the 15th IEEE International Symposium on High Performance Distributed Computing, pages 7–17.

[128] J. Leskovec and E. Horvitz. 2008. Planetary-Scale Views on a Large Instant-messaging Network. In: Proceedings of the 17th International Conference on World Wide Web (WWW), pages 915–924.

[129] J. Li, J. Stribling, T. Gil, R. Morris, and M. Kaashoek. 2005. Comparing the Performance of Distributed Hash Tables Under Churn. In: Peer-to-Peer Systems III, volume 3279 of *Lecture Notes in Computer Science*, pages 87–99. Springer. ISBN 978-3-540-24252-9.

[130] J. Li, J. Stribling, R. Morris, M. F. Kaashoek, and T. M. Gil. 2005. A Performance vs. Cost Framework for Evaluating DHT Design Tradeoffs Under Churn. In: Proceedings of the IEEE INFOCOM 2005, pages 225–236.

[131] L. Li, Y. Ji, T. Ma, L. Gu, and C. Zhang. 2008. Locality-Aware Peer-to-Peer SIP. In: Proceedings of the 14th IEEE International Conference on Parallel and Distributed Systems (ICPADS), pages 295–302.

[132] D. Liben-Nowell, H. Balakrishnan, and D. Karger. 2002. Observations on the Dynamic Evolution of Peer-to-Peer Networks. In: Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS), pages 22–33.

[133] E. K. Lua, T. Griffin, M. Pias, H. Zheng, and J. Crowcroft. 2005. On the Accuracy of Embeddings for Internet Coordinate Systems. In: Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement (IMC), pages 125–138.

[134] J. Mäenpää. 2005. Performance of Signaling Compression in the Third Generation Mobile Network. Master's thesis, Helsinki University of Technology, Finland.

[135] J. Mäenpää. 2007. Use of Ad Hoc Networks for Wireless Internet Access. In: Proceedings of the 6th Conference on Telecommunication Techno-Economics (CTTE), pages 1–8.

[136] J. Mäenpää. 2013. Using SOMs and CoAP-RELOAD to Enable Autonomous Wide Area Sensor Networks. In: Proceedings of the 10th IEEE Consumer Communications and Networking Conference (CCNC), pages 1–4.

[137] J. Mäenpää and G. Camarillo. 2013. Service Discovery Usage for REsource LOcation And Discovery (RELOAD). Internet-Draft draft-ietf-p2psip-service-discovery-08, Internet Engineering Task Force. Work in Progress.

[138] J. Mäenpää, G. Camarillo, and J. Hautakorpi. 2013. A Self-tuning Distributed Hash Table (DHT) for REsource LOcation And Discovery (RELOAD). Internet-Draft draft-ietf-p2psip-self-tuning-08, Internet Engineering Task Force. Work in Progress.

[139] R. Mahajan, M. Castro, and A. I. T. Rowstron. 2003. Controlling the Cost of Reliability in Peer-to-Peer Overlays. In: Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS), pages 21–32.

[140] R. Mahy, P. Matthews, and J. Rosenberg. 2010. Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN). RFC 5766, Internet Engineering Task Force.

[141] G. S. Manku. 2004. Balanced Binary Trees for ID Management and Load Balance in Distributed Hash Tables. In: Proceedings of the 23rd Annual ACM Symposium on Principles of Distributed Computing (PODC), pages 197–205.

[142] E. Marocco and D. Bryan. 2007. Interworking between P2PSIP Overlays and Conventional SIP Networks. Internet-Draft draft-marocco-p2psip-interwork-01, Internet Engineering Task Force. Work in Progress.

[143] E. Marocco and E. Ivov. 2007. Extensible Peer Protocol (XPP). Internet-Draft draft-marocco-p2psip-xpp-01, Internet Engineering Task Force. Work in Progress.

[144] E. Marocco, A. Manzalini, M. Sampò, and G. Canal. 2008. Interworking between P2PSIP Overlays and IMS Networks – Scenarios and Technical Solutions. In: Proceedings of the 9th International Conference on SIP (International SIP), pages 1–5.

[145] I. Martinez-Yelmo, C. Guerrero, R. Cuevas, and A. Mauthe. 2009. A Hierarchical P2PSIP Architecture to Support Skype-like Services. In: Proceedings of the 17th Euromicro International Conference on Parallel, Distributed and Network-based Processing, pages 316–322.

[146] I. Martínez-Yelmo, A. Bikfalvi, and C. Guerrero. 2009. Benefits on Using H-P2PSIP in Mobile Environments. In: Proceedings of the VIII Jornadas de Ingeniería Telemática (JITEL), pages 1–8.

[147] L. Massoulié, E. Le Merrer, A.-M. Kermarrec, and A. Ganesh. 2006. Peer Counting and Sampling in Overlay Networks: Random Walk Methods. In: Proceedings of the 25th Annual ACM Symposium on Principles of Distributed Computing (PODC), pages 123–132.

[148] F. Mattern and C. Floerkemeier. 2010. From the Internet of Computers to the Internet of Things. In: From Active Data Management to Event-Based Systems and More, volume 6462 of *Lecture Notes in Computer Science*, pages 242–259. Springer. ISBN 978-3-642-17225-0.

[149] G. Matthew and J. Rodriquez. 2009. A Proposed P2PSIP Framework for Extreme Emergency Situations in a Wireless Environment. In: Proceedings of the 5th International ICST Mobile Multimedia Communications Conference (Mobimedia), pages 1–5.

[150] M. Matuszewski and E. Kokkonen. 2008. Mobile P2PSIP - Peer-to-Peer SIP Communication in Mobile Communities. In: Proceedings of the 5th IEEE Consumer Communications and Networking Conference (CCNC), pages 1159–1165.

[151] P. Maymounkov and D. Mazières. 2002. Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. In: Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS), pages 53–65.

[152] B. Meyer and M. Portmann. 2008. Practical Performance Evaluation of Peer-to-Peer Internet Telephony Using SIP. In: Proceedings of the 8th IEEE International Conference on Computer and Information Technology Workshops (CIT Workshops), pages 204–209.

[153] D. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, and Z. Xu. 2002. Peer-to-Peer Computing. Technical Report HPL-2002-57 (R.1), HP Laboratories.

[154] M. Mitzenmacher, A. W. Richa, and R. Sitaraman. 2000. The Power of Two Random Choices: A Survey of Techniques and Results. In: Handbook of Randomized Computing, volume 1. Kluwer. ISBN 978-0-7923-6959-2, 255-312 pages.

[155] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler. 2007. Transmission of IPv6 Packets over IEEE 802.15.4 Networks. RFC 4944, Internet Engineering Task Force.

[156] C. D. Morse and H. Wang. 2005. The Structure of an Instant Messenger Network and Its Vulnerability to Malicious Codes. In: Proceedings of ACM SIGCOMM 2005 Poster Session, pages 1–2.

[157] R. Moskowitz, P. Nikander, P. Jokela, and T. Henderson. 2008. Host Identity Protocol. RFC 5201, Internet Engineering Task Force.

[158] S. Narayanan, D. Braun, J. Buford, R. Fish, A. Gelman, A. Kaplan, R. Khandelwal, E. Shim, and H. Yu. 2007. Peer-to-peer Streaming for Networked Consumer Electronics. IEEE Communications Magazine 45, no. 6, pages 124–131.

[159] B. A. Nardi, S. Whittaker, and E. Bradner. 2000. Interaction and Outeraction: Instant Messaging in Action. In: Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work (CSCW), pages 79–88.

[160] A. Nicolosi and S. Annapureddy. 2003. P2PCAST: A Peer-to-Peer Multicast Scheme for Streaming Data. In: Proceedings of the 1st IRIS Student Workshop (ISW), pages 1–6.

[161] Z. Ou, E. Harjula, and M. Ylianttila. 2009. Effects of Different Churn Models on the Performance of Structured Peer-to-Peer Networks. In: Proceedings of the 20th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, pages 2856–2860.

[162] Z. Ou, E. Harjula, O. Kassinen, and M. Ylianttila. 2009. Feasibility Evaluation of a Communication-oriented P2P System in Mobile Environments. In: Proceedings of the 6th International Conference on Mobile Technology, Application and Systems (Mobility), pages 1–8.

[163] Z. Ou, J. Zhou, E. Harjula, and M. Ylianttila. 2009. Truncated Pyramid Peer-to-Peer Architecture with Vertical Tunneling Model. In: Proceedings of the 6th IEEE Consumer Communications and Networking Conference (CCNC), pages 1227–1231.

[164] G. Panagiotou. 2008. A P2PSIP Event Notification Architecture. Master's thesis, Royal Institute of Technology (KTH), Stockholm.

[165] C. Papageorgiou, K. Birkos, T. Dagiuklas, and S. Kotsopoulos. 2010. Extending P2PSIP VoIP Communication for Ad Hoc Networks. In: Proceedings of the 6th ACM Workshop on QoS and Security for Wireless and Mobile Networks (Q2SWinet), pages 106–109.

[166] J. Peltotalo, J. Harju, A. Jantunen, M. Saukko, L. Vaatamoinen, I. Curcio, I. Bouazizi, and M. Hannuksela. 2008. Peer-to-Peer Streaming Technology Survey. In: Proceedings of the 7th International Conference on Networking (ICN), pages 342–350.

[167] Y. Peng, W. Wang, Z. Hao, and Y. Meng. 2012. An SNMP Usage for RELOAD. Internet-Draft draft-peng-p2psip-snmp-05, Internet Engineering Task Force. Work in Progress.

[168] C. Perkins, E. Belding-Royer, and S. Das. 2003. Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561, Internet Engineering Task Force.

[169] C. E. Perkins and P. Bhagwat. 1994. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In: Proceedings of the 1994 Conference on Communications Architectures, Protocols and Applications (ACM SIGCOMM), pages 234–244.

[170] L. Peterson, A. Bavier, M. E. Fiuczynski, and S. Muir. 2006. Experiences Building PlanetLab. In: Proceedings of the 7th Symposium on Operating Systems Design and Implementation (OSDI), pages 351–366.

[171] M. Petit-Huguenin, J. Rosenberg, and C. Jennings. 2012. A Usage of Resource Location and Discovery (RELOAD) for Public Switched Telephone Network (PSTN) Verification. Internet-Draft draft-petithuguenin-vipr-reload-usage-04, Internet Engineering Task Force. Work in Progress.

[172] R. Price, C. Bormann, J. Christoffersson, H. Hannu, Z. Liu, and J. Rosenberg. 2003. Signaling Compression (SigComp). RFC 3320, Internet Engineering Task Force.

[173] D. Psaltoulis, D. Kostoulas, I. Gupta, K. Birman, and A. Demers. 2004. Practical Algorithms for Size Estimation in Large and Dynamic Groups. In: Proceedings of the 23rd Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC), pages 1–10.

[174] K. Pussep, M. Weinert, A. Kovacevic, and R. Steinmetz. 2008. On NAT Traversal in Peer-to-Peer Applications. In: Proceedings of the 17th IEEE Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), pages 139–140.

[175] A. Rao, K. Lakshminarayanan, S. Surana, R. M. Karp, and I. Stoica. 2003. Load Balancing in Structured P2P Systems. In: Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS), pages 68–79.

[176] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. 2001. A Scalable Content-addressable Network. In: Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (ACM SIGCOMM), pages 161–172.

[177] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu. 2003. Data-centric Storage in Sensornets with GHT, a Geographic Hash Table. Mobile Networks and Applications 8, no. 4, pages 427–442.

[178] S. Ren, L. Guo, and X. Zhang. 2006. ASAP: An AS-Aware Peer-Relay Protocol for High Quality VoIP. In: Proceedings of the 26th IEEE International Conference on Distributed Computing Systems (ICDCS), pages 70–80.

[179] E. Rescorla and N. Modadugu. 2006. Datagram Transport Layer Security. RFC 4347, Internet Engineering Task Force.

[180] S. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz. 2004. Handling Churn in a DHT. In: Proceedings of the 2004 USENIX Annual Technical Conference (ATEC), pages 127–140.

[181] S. Rhea, B. Godfrey, B. Karp, J. Kubiatowicz, S. Ratnasamy, S. Shenker, I. Stoica, and H. Yu. 2005. OpenDHT: A Public DHT Service and Its Uses. In: Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (ACM SIGCOMM), pages 73–84.

[182] S. Rieche, L. Petrak, and K. Wehrle. 2004. A Thermal-dissipation-based Approach for Balancing Data Load in Distributed Hash Tables. In: Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks, pages 15–23.

[183] J. Risson and T. Moors. 2007. Survey of Research towards Robust Peer-to-Peer Networks: Search Methods. RFC 4981, Internet Engineering Task Force.

[184] A. B. Roach. 2002. Session Initiation Protocol (SIP)-Specific Event Notification. RFC 3265, Internet Engineering Task Force.

[185] J. Rosenberg. 2004. A Presence Event Package for the Session Initiation Protocol (SIP). RFC 3856, Internet Engineering Task Force.

[186] J. Rosenberg, R. Mahy, P. Matthews, and D. Wing. 2009. Session Traversal Utilities for NAT (STUN). RFC 5389, Internet Engineering Task Force.

[187] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. 2002. SIP: Session Initiation Protocol. RFC 3261, Internet Engineering Task Force.

[188] J. Rosenberg. 2010. Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols. RFC 5245, Internet Engineering Task Force.

[189] A. I. T. Rowstron and P. Druschel. 2001. Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems. In: Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), pages 329–350.

[190] J. Sacha, J. Dowling, R. Cunningham, and R. Meier. 2006. Discovery of Stable Peers in a Self-organising Peer-to-Peer Gradient Topology. In: Distributed Applications and Interoperable Systems, volume 4025 of *Lecture Notes in Computer Science*, pages 70–83. Springer. ISBN 978-3-540-35126-9.

[191] R. Schollmeier. 2001. A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications. In: Proceedings of the 1st International Conference on Peer-to-Peer Computing (P2P), pages 101–102.

[192] H. Schulzrinne, E. Marocco, and E. Ivov. 2010. Security Issues and Solutions in Peer-to-Peer Systems for Realtime Communications. RFC 5765, Internet Engineering Task Force.

[193] J. Seedorf. 2006. Security Challenges for Peer-to-Peer SIP. IEEE Network 20, no. 5, pages 38–45.

[194] J. Seedorf. 2008. Lawful Interception in P2P-Based VoIP Systems. In: Principles, Systems and Applications of IP Telecommunications. Services and Security for Next Generation Networks, volume 5310 of *Lecture Notes in Computer Science*, pages 217–235. Springer. ISBN 978-3-540-89053-9.

[195] J. Seedorf and E. Burger. 2009. Application-Layer Traffic Optimization (ALTO) Problem Statement. RFC 5693, Internet Engineering Task Force.

[196] A. Sentinelli, G. Marfia, M. Gerla, L. Kleinrock, and S. Tewari. 2007. Will IPTV ride the peer-to-peer stream? IEEE Communications Magazine 45, no. 6, pages 86–92.

[197] T. M. Shafaat, A. Ghodsi, and S. Haridi. 2008. A Practical Approach to Network Size Estimation for Structured Overlays. In: Self-Organizing Systems, volume 5343 of *Lecture Notes in Computer Science*, pages 71–83. Springer. ISBN 978-3-540-92156-1.

[198] Z. Shelby, K. Hartke, C. Bormann, and B. Frank. 2012. Constrained Application Protocol (CoAP). Internet-Draft draft-ietf-core-coap-13, Internet Engineering Task Force. Work in Progress.

[199] Z. Shelby, S. Krco, and C. Bormann. 2013. CoRE Resource Directory. Internet-Draft draft-shelby-core-resource-directory-05, Internet Engineering Task Force. Work in Progress.

[200] Z. Shelby. 2012. Constrained RESTful Environments (CoRE) Link Format. RFC 6690, Internet Engineering Task Force.

[201] K. Singh and H. Schulzrinne. 2005. Peer-to-Peer Internet Telephony Using SIP. In: Proceedings of the International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV), pages 63–68.

[202] P. Srisuresh, B. Ford, and D. Kegel. 2008. State of Peer-to-Peer (P2P) Communication across Network Address Translators (NATs). RFC 5128, Internet Engineering Task Force.

[203] P. Srisuresh and M. Holdrege. 1999. IP Network Address Translator (NAT) Terminology and Considerations. RFC 2663, Internet Engineering Task Force.

[204] P. Srisuresh, G. Tsirtsis, P. Akkiraju, and A. Heffernan. 1999. DNS extensions to Network Address Translators (DNS_ALG). RFC 2694, Internet Engineering Task Force.

[205] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan. 2003. Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications. IEEE/ACM Transactions on Networking 11, no. 1, pages 17–32.

[206] D. Stutzbach and R. Rejaie. 2006. Understanding Churn in Peer-to-Peer Networks. In: Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement (IMC), pages 189–202.

[207] S. Surana, B. Godfrey, K. Lakshminarayanan, R. Karp, and I. Stoica. 2006. Load Balancing in Dynamic Structured Peer-to-Peer Systems. Performance Evaluation 63, no. 3, pages 217–240.

[208] Z. Tian, X. Wen, W. Zheng, Y. Sun, and Y. Cheng. 2009. Evaluation and Simulation on the Performance of DHTs Required by P2PSIP. In: Proceedings of the 5th International Conference on Information Assurance and Security (IAS), pages 701–704.

[209] UPnP Forum. 2008. UPnP Device Architecture 1.1. Device Architecture Document, Universal Plug and Play Forum.

[210] P. van der Stok and K. Lynn. 2011. CoAP Utilization for Building Control. Internet-Draft draft-vanderstok-core-bc-05, Internet Engineering Task Force. Work in Progress.

[211] J.-P. Vasseur and A. Dunkels. 2010. Interconnecting Smart Objects with IP: The Next Internet. Morgan Kaufmann. ISBN 978-0-123-75165-2.

[212] D. W. and C. Poellabauer. 2010. Fundamentals of Wireless Sensor Networks: Theory and Practice. John Wiley and Sons. ISBN 978-0-470-99765-9.

[213] A. Wacker, G. Schiele, S. Holzapfel, and T. Weis. 2008. A NAT Traversal Mechanism for Peer-To-Peer Networks. In: Proceedings of the 8th International Conference on Peer-to-Peer Computing (P2P), pages 81–83.

[214] J. Wang, Z. Chen, Y. Meng, and J. Shen. 2009. P2PSIP Event Notification Extension. Internet-Draft draft-wang-p2psip-event-notification-extension-01, Internet Engineering Task Force. Work in Progress.

[215] J.-F. Wauthy and L. Schumacher. 2007. Implementation and Performance Evaluation of a P2PSIP Distributed Proxy/Registrar. In: Proceedings of the 2007 International Conference on Next Generation Mobile Applications, Services and Technologies (NGMAST), pages 119–124.

[216] M. Ylianttila, E. Harjula, T. Koskela, O. Kassinen, and J. Riekki. 2008. Mobile Plug-and-Play Architecture for Collaborative Hybrid Peer-to-Peer Applications. In: Proceedings of the Congress on Services Part II (SERVICES-2), pages 81–87.

[217] M. Ylianttila, E. Harjula, T. Koskela, and J. Sauvola. 2008. Analytical Model for Mobile P2P Data Management Systems. In: Proceedings of the 5th IEEE Consumer Communications and Networking Conference (CCNC), pages 1186–1190.

[218] C. Zhang, Y. Ji, and L. Li. 2008. Bandwidth and Efficiency of P2PSIP Based Server Farm. In: Proceedings of the 2008 International Conference on Computer Science and Software Engineering (CSSE), pages 495–498.

[219] C. Zhang, J. Shi, L. Li, W. Lin, Y. Wang, L. Gu, Y. Ji, and Z. Feng. 2008. Signaling Latency Analysis of Peer-to-Peer SIP Systems. In: Proceedings of the 5th IEEE Consumer Communications and Networking Conference (CCNC), pages 505–509.

[220] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz. 2004. Tapestry: A Resilient Global-scale Overlay for Service Deployment. IEEE Journal on Selected Areas in Communications 22, no. 1, pages 41–53.

[221] X. Zheng and V. Oleshchuk. 2010. Secure Interworking with P2PSIP and IMS. In: Proceedings of the 2010 International Symposium on Collaborative Technologies and Systems (CTS), pages 481–488.

[222] X. Zheng and V. Oleshchuk. 2009. Improving Chord Lookup Protocol for P2PSIP-Based Communication Systems. In: Proceedings of the 2009 International Conference on New Trends in Information and Service Science (NISS), pages 1309–1314.

[223] X. Zheng and V. Oleshchuk. 2009. A Secure Architecture for P2PSIP-based Communication Systems. In: Proceedings of the 2nd International Conference on Security of Information and Networks (SIN), pages 75–82.

[224] J. Zhou, J. Buford, K. Dhara, M. Kolberg, V. Krishnaswamy, and X. Wu. 2007. Discovery and Composition of Communication Services in Peer-to-Peer Overlays. In: Proceedings of the 2007 IEEE Global Telecommunications Conference (GLOBECOM), pages 1–8.

Errata

Publication I

The lines in the legend of Figure 2 are in a different order than the curves in the figure. The line for the highest churn rate (1/5s) should be the top one in the legend and the line for the lowest churn rate (1/30s) the bottom one.

Publication VI

The first sentence of the last paragraph of Section 7.5 should be: "It should be noted that the results look slightly different when the overlay is static (i.e., when there is no peer arrivals and N stays constant at 10,000) since in that case the root of the tree is always stored by the same peer."

Over the past decade, Peer-to-Peer (P2P) technologies have proven themselves as a viable option for providing services in the Internet. This dissertation focuses on the use of P2P technologies in the context of communication services, including Voice over IP (VoIP) and Machine-to-Machine (M2M) communication. The main result presented in the dissertation is a framework architecture for decentralized communications. The framework architecture represents a toolkit that can be used to provide communication services in a decentralized mode. The framework architecture is adaptive, scalable, modular, generic, based on current and emerging standards, and has high performance.

Aalto-DD 62/2013

9 789526 051208

ISBN 978-952-60-5120-8
ISBN 978-952-60-5121-5 (pdf)
ISSN-L 1799-4934
ISSN 1799-4934
ISSN 1799-4942 (pdf)

Aalto University
Aalto School of Electrical Engineering
Communications and Networking
www.aalto.fi