# The Design of Efficient and Secure P2PSIP Systems

**Xianghan Zheng**

# The Design of Efficient and Secure P2PSIP Systems

Doctoral Dissertation for the degree *Philosophiae Doctor (PhD)* in Information and Communication Technology

University of Agder
Faculty of Engineering and Science
2010

Doctoral Dissertations at the University of Agder 27
ISBN: 978-82-7117-680-8
ISSN: 1504-9272

Printed by the Printing Office, University of Agder
Kristiansand

# ACKNOWLEDGEMENTS

This thesis was submitted in October 2010 to University of Agder, Norway, for the degree of PhD in Information and Communication Technology. The thesis would not have been possible unless the help of many people I want to thank.

I would like to express my sincere gratitude to my supervisor Prof. Vladimir A. Oleshchuk for continuous instrumental support in three years. How time flies! I still remember clearly three years ago when we first time sat together, discussing about research topic, method, and technical issues. I am very grateful for your admirable patience, motivation, inspiration, enthusiasm and broad knowledge in networking and security. My debt to you, your writing, your lecture, and your seminars, must be very evident from the contents of the thesis, publications, and will continuously appear in my future research. I could not expect having a better supervisor and mentor for my PhD study.

Besides, I would like to thanks to my co-supervisor, Prof. Frank Reichert, who directed me into mobile world five years ago. Without your reference, I would be not possible to have wonderful working experience in Agder Mobility Lab (AML) and later Ericsson Research. And it is this research experience that makes me work efficiently and perfectly during later PhD research.

Thanks also to another co-supervisor Prof. Andreas Prinz for his instrumental supervision in the course of modelling. Without your great patience and supervision, I was not possible to complete abstract modelling project and pass severe examination. Besides, thanks for your excellent administration work, which brings a lot of benefit to all ICT department members including me.

I would like to thank administration part of ICT department and all colleagues for all your help. I thank PhD coordinator Trine Tønnessen for many times directing me and negotiating with administration when I had problems. Without these support I cannot imagine being able to complete PhD study in only three years. Also thanks for IT help department stuff for their support.

I want to express my gratitude to all members in Agder Mobility Lab and Mobile Communication Lab (MCL). It is really pleasure and comfortable to work and stay with all of you, for many times stimulating discussions, constructive suggestions, paper cooperation, and interesting social activities.

Last but not least, I would thank my parents for giving birth to me, supporting me throughout my life. I am also grateful to my fiancé Ying Huang. Your understanding, endless patience and continuous encouragement turn my research into a pleasure journey.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# ABBREVIATIONS

| | |
|---|---|
| 3G | 3rd Generation |
| 3GPP | 3rd Generation Partnership Project |
| AH | Authentication Head |
| AJAX | Asynchronous JavaScript and XML |
| AP | Access Provider |
| AS | Application Server |
| B2BUA | Back-to-Back User Agent |
| Bi-Chord | Bi-directional Chord |
| C/S | Client/Server |
| CA | Certificate Authority |
| CAN | Content Addressable Network |
| CE | Consumer Electronics |
| CPU | Central Processing Unit |
| CSCF | Call Session Control Function |
| CSP | Chord Secure Proxy |
| CSPG | Chord Secure Proxy Gateway |
| CSS | Cascading Style Sheets |

| | |
|---|---|
| DB | Database |
| DDoS | Distributed Denial of Service |
| DHTML | Dynamic HTML |
| DLNA | Digital Living Network Alliance |
| DNS | Domain Name System |
| DOM | Document Object Model |
| DoS | Denial of Service |
| E&A | Enrollment and Authentication |
| E&D | Encryption and Decryption |
| HIP | Host Identity Protocol |
| HSS | Home Subscriber Server |
| HTML | Hypertext Markup Language |
| HTTP | Hypertext Transfer Protocol |
| ICE | Interactive Connectivity Establishment |
| ID | Identity |
| IDE | Integrated Development Environment |
| IETF | Internet Engineering Task Force |
| IGW | IMS Gateway |
| IM | Instant Messaging |

| | |
|---|---|
| IMPU | IP Multimedia Public Identity |
| IMS | IP Multimedia Subsystem |
| IOP | Initial Opinion |
| IP | Internet Protocol |
| IPSec | Internet Protocol Security |
| IPv4 | Internet Protocol version 4 |
| ISDN | Integrated Services Digital Network |
| ISP | Internet Service Provider |
| MIME | Multipurpose Internet Mail Extensions |
| MITM | Man in the Middle Attack |
| MSN | Windows Service Network |
| NAT | Network Address Translation |
| NO | Network Operator |
| OMA | Open Mobile Alliance |
| P2P | Peer to Peer |
| P2PP | Peer-to-Peer Protocol |
| P2PSIP | Peer to Peer Session Initiation Protocol |
| PC | Personal Computer |
| PDA | Personal Digital Assistant |

| | |
|---|---|
| PIGW | P2PSIP IMS Gateway |
| PKI | Public Key Infrastructure |
| PSK | Pre-Shared Key |
| PSP | PlayStation Portable |
| PSTN | Public Switched Telephone Network |
| RELOAD | REsource LOcation And Discovery |
| RFC | Request for Comments |
| RGW | Residential Gateway |
| RPC | Remote Procedure Call |
| RSA | Rivest, Shamir and Adleman |
| S-CSCF | Server Call Session Control Function |
| SDP | Session Description Protocol |
| SDS | Service Development Stutio |
| SEG | Security Gateway |
| SEP | Service Extensible P2P Peer Protocol |
| SETI@Home | Search for Extraterrestrial Intelligence At Home |
| SHA-1 | Secure Hash Algorithm |
| SIMPLE | Session Initiation Protocol for Instant Messaging and Presence Leveraging Extension |
| SIP | Session Initiation Protocol |

| | |
|---|---|
| SMTP | Simple Mail Transfer Protocol |
| SN | Son Node |
| SOAP | Simple Object Access Protocol |
| SOS | Secure Opinion Server |
| SOSIMPLE | A Serverless, Standards-based, P2P SIP Communication System |
| SP | Service Provider |
| SPIT | Spam over Internet Telephony |
| SSL | Secure Socket Layer |
| STUN | Simple Traversal of UDP through NATs |
| TCP | Transmission Control Protocol |
| TC-PPSG | Thin Client P2PSIP Gateway |
| TLS | Transport Layer Security |
| TOS | Type of Service |
| TURN | Traversal Using Relay NAT |
| UDP | User Datagram Protocol |
| UICC | Universal Integrated Circuit Card |
| UPnP | Universal Plug and Play |
| URI | Uniform Resource Identifier |
| UUCP | Unix-to-Unix Copy |

| | |
|---|---|
| VoIP | Voice over IP |
| WAP | Wireless Application Protocol |
| WG | Working Group |
| WiFi | Wireless Fidelity |
| WLAN | Wireless LAN |
| XML | Extensible Markup Language |

# Chapter 1. Introduction

P2PSIP is regarded as a promising solution for future communication systems. In this chapter, we introduce the background of research topic, the research area and target, the motivation, and the thesis overview.

## 1.1. BACKGROUND

Peer-to-Peer (P2P) computing has attracted great attention in both academia and industry. Comparing with traditional server-based system architecture in which most of functionality is executed on server side, P2P-based computing allocates computing task to all participating peers. This might eliminate/reduce functionality of server and therefore provides better robustness on system level. Today, P2P computing has been widely implemented in many kinds of networking systems and applications.

In communication field, one of the most well-known P2P applications is Skype [22], which offers free Voice-over-IP (VoIP) and Instant Messaging (IM) services for computer-to-computer and charged services for computer-to-PSTN. Additionally, Skype service has been extended to mobile world. Many mobile platforms today (e.g. Symbian S60 [15], iPhone OS [3], Android [1], Windows Mobile [13], etc), have Wi-Fi/3G connection based Skype application. According to eBay statistics [9], the number of Skype users has reached 521 million before Q3, 2009, and it is still growing fast.

However, Skype protocol has been monopolized and is not publicly available. Although part of its functionality (e.g. login, NAT traversal, media transfer, codec, etc) has been understood via analyzing Skype network traffic [33, 43, 51], researchers (outside Skype project) are still uncertain about its core technical specification, disadvantages, and required improvement. Besides, Skype does not provide interoperability with other open applications, such as SIP based WLM (Windows Live Messenger) [28] , Yahoo IM [31], etc. This is partly because of technical difficulty in seamless interconnecting among different protocols, and partly because of unwillingness of cooperation with its competitors.

The success of Skype greatly inspired the research on peer-to-peer based communication systems. Researchers were trying to find an alternative solution where on one hand, decentralized nodes are capable to auto-configure themselves in IP-based Ad-Hoc style; and on the other hand, the designed protocol supports fast location of nodes, optimized route selection, secure and reliable service delivery. Let us take a look at Figure 1-1, which illustrates an example of P2P based communication paradigm. Each node in the system has a few connections with its neighbours, and these neighbours act as intermediate nodes to deliver requests and responses. A few routing mechanisms, negotiation protocols are supposed to be implemented so that session between source peer (for instance, A in the figure) and destination peer (B) can be established in optimized way. The designed protocol is assumed to be an open standard so that everyone could develop applications on it.

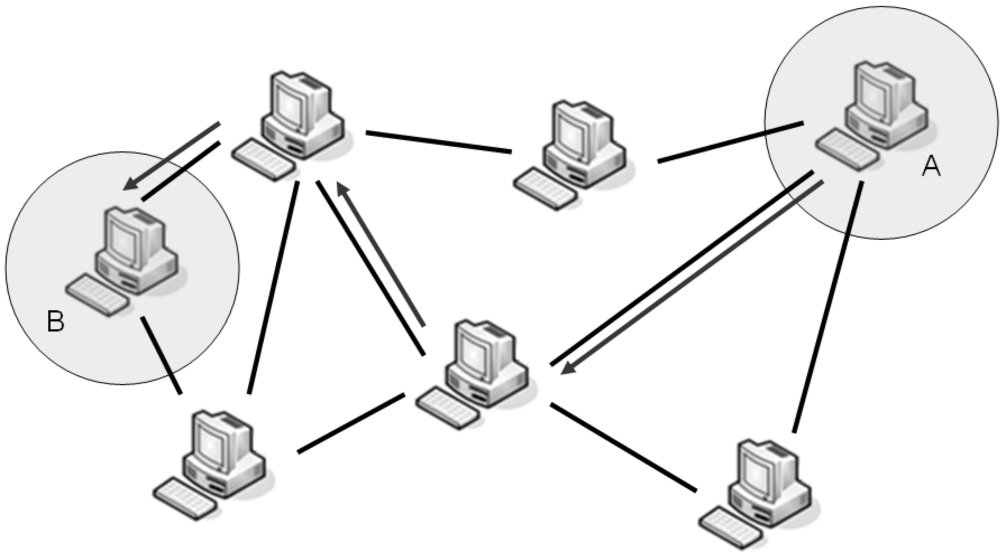


Figure 1-1: P2P based Communication

However, reality is different. After a few studies, researchers began to recognize that it was not trivial to realize this type communication paradigm. P2P protocol lacks of session level description and negotiation mechanisms, which make some functionalities (such as optimized route selection, media codec negotiation, etc) difficult to achieve. At this moment, Session Initiation Protocol (SIP) comes into the sight.

Session Initiation Protocol (SIP) is a transaction-oriented, text-based protocol that inherits the simplicity from Hypertext Transfer Protocol (HTTP) and Simple Mail Transfer Protocol (SMTP) [108]. It is designed to create, modify, and terminate sessions with one or more participants. Because of its characteristics (e.g. simplicity, extensibility, flexibility, etc), SIP is chosen by 3rd Generation Partnership Project (3GPP) as the main protocol for the IP Multimedia Subsystems (IMS)-based future All-IP network [21]. SIP to telecommunication systems is regarded as important as HTTP to Internet.

Therefore, researchers began to study approaches that combine decentralization nature of P2P with efficiency of SIP protocol. In 2003, the SIPpeer project at University of Columbia [101, 102] and the SOSIMPLE project at William & Mary College [36] were the first attempts of study of P2PSIP communication systems. In the following years, the P2PSIP research has attracted great attention both from academia and industry (e.g. Cisco, Nokia, Ericsson, HuaWei, etc). IETF P2PSIP Working Group defines the motivation of P2PSIP [18] as following: “The concept behind P2PSIP is to leverage the distributed nature of P2P to allow for distributed resource discovery in a SIP network, eliminating (at least reducing) the need for centralized servers”.

A lot of possible solutions have been proposed in recent publications [37, 41, 48, 75] and Internet-drafts [38, 59, 73, 104] . However, P2PSIP is still far from mature. Many technical issues are waiting for solutions. Two of the most critical issues are efficiency and security.

As originally designed for file sharing, P2P protocol does not focus much on system efficiency. For instance, the delay in P2P applications (e.g. BitComet [4], BitTorrent [5], etc) can be from a few seconds to even minutes. This is unacceptable in real-time multimedia services. Therefore, conventional peer/resource lookup algorithm is required to be improved or replaced.

Besides, the decentralized nature of P2P comes to the cost of less or decentralized management, which might also create security problems. In such scenario all participating nodes may distrust each other due to lack of centralized credential mechanism. It is possible that some malicious nodes create negative experience to the other nodes (for example, modify the received message and forward it out) or the overlay. Therefore, data transactions from beginning to end are regarded as not trusted.

## 1.2 Research Area and Subject Definition

As argued in the previous section, P2PSIP is a new future trend [52, 115, 123]. The research area of this work is a P2PSIP based communication paradigm. The related technical issues include: P2P, SIP, P2PSIP, client, system efficiency, security, and interworking.

There are three main entities in P2PSIP systems: peer, client, and resource. Peer is a node that participates in overlay and is responsible for distributed storage and cooperative transport services. Client is a special entity, which participates in overlay; however, does not make contribution to other nodes due to node's unwillingness or limitation of system capabilities (e.g. low CPU power, bandwidth, storage, etc). Resource is data that is distributed and stored among participating peers.

Our work focuses on P2PSIP session initiation process, especially on how to improve system efficiency and enhance security. Besides, we also consider inter-working issues. We do not consider much about advanced services, such as Voice-over-IP, group chatting, off-line messages, etc. We believe that these use scenarios are supported as soon as protocol is standardized. In addition, the thesis does not consider much about how overlay activities, such as Churn (peer join/leave frequently) and their impact on the P2P network.

## 1.3 Motivation

The goal of this work is to investigate P2PSIP based paradigm in communication systems and study possible solutions that are capable to solve critical issues in P2PSIP systems mentioned in the previous sections.

Since P2PSIP systems are not efficient, our first attempt is to use (adapt) a few conventional and novel approaches to reduce delays during session establishment phase. The proposed mechanisms should act as the extended functionality that enriches the current approaches.

Since P2PSIP system is not secure [58, 96] and quite few researchers are focused on this issue. Our aim is to investigate security challenges and propose feasible security enhancement mechanisms. For instance, it is necessary to study both central based security and distributed based security approaches. Besides, appropriate combination of these two security solutions in one system might provide the optimized result.

Moreover, a P2PSIP client, as a ”weak link” of a P2PSIP system that might not have enough capabilities (e.g. high CPU power, bandwidth, storage, etc) to access/protect services, needs special concern. Although there are already published proposals, they still contain many issues that need to be solved (insecure, immature, difficult to implement, etc., as discussed in Chapter 7). Our goal is to propose a feasible, reliable, and secure alternative solutions that are capable to be implemented by most of portable and fixed devices currently existing.

Last but not the least, the future goes for All-IP IMS based network. Therefore, P2PSIP needs efficient and secure inter-working solutions interconnecting with IMS. Our intention is to integrate concept of security mechanisms with usage of interworking gateway and propose appropriate interconnecting system architecture.

## 1.4 THESIS OVERVIEW

The following Chapter 2 introduces state of the art of this thesis' research area. It contains description of P2P computing, SIP and SDP protocol, and survey of P2PSIP approaches. In the survey, we introduce P2PSIP requirement, current proposals, and existing challenges.

Chapter 3 studies a few approaches that can be implemented to improve P2PSIP system efficiency via reducing delay during session establishment. After that, some of mechanisms are implemented and experimentally evaluated.

Chapter 4 makes a survey on P2PSIP security. It starts with introduction of security challenges including general security problems and P2P specific problems, followed by possible solution proposed.

To solve security problem, Chapter 5 introduces two solutions: central based security and distributed trust security, both of which have their own advantages and disadvantages. After that, we study possible combination of these two approaches to get optimized protection.

Chapter 6 introduces a possible inter-working model that interconnects P2PSIP with future All-IP based IMS network. The security approaches introduced in Chapter 5 can be reused here.

Chapter 7 discusses a possible system architecture for P2PSIP client to access services. The proposed solution, after comparison with early proposed solution (client protocol solution), is shown to be better than previous proposal. The security approaches described in Chapter 5 can be reused here.

At last Chapter 8 concludes thesis and suggests future work.

# Chapter 2. State-of-the Art

This chapter introduces state of the art of P2PSIP. It starts with introduction of P2P technology and specification of P2P overlay algorithms; followed by description of SIP protocol. After that, we make a survey on P2PSIP according to recent proposals. The survey includes technical requirements, current proposals, and existing challenges.

## 2.1 PEER-TO-PEER TECHNOLOGY

Peer-to-Peer system has been utilized for more than 30 years. In 1979, Usenet [25] was developed to exchange information among Unix systems, based on Unix-to-Unix-copy (UUCP) protocol. In the following years, Usenet grew from the original two sites to hundreds of thousands of sites. Generally, Usenet is the grandfather of all P2P networks.

P2P systems get fast development from 1990s, from first generation centralized P2P applications (such as Napster [14], SETI@Home [20], etc) in which a centralized server is implemented for index services, to the second generation decentralized but unstructured P2P such as Gnutella [32] and Freenet [42]. More advanced technology is based on structured network, especially Distributed Hash Table (DHT) [43], in which overlay algorithms provide auto-configuration and routing optimization.

In this section, Peer-to-Peer related technologies are described. The comparison between P2P model and traditional Client/Server (C/S) based model is presented. After that, we specify P2P overlay networks with special focus on DHT overlay suggested for P2PSIP communication.

## 2.1.1 Peer-to-Peer vs Client-Server

Before introducing what is P2P, let us take a look at C/S based system architecture presented in Figure 2-1 (a). It is the most popular system approach today. In this approach, the most important unit is the server, providing resources such as storage and computing capability to clients. A client, who wants to connect to the other clients, has to ask a server for establishing such connections. Since the number of servers is limited and system capacity is fixed, the increasing number of clients means reduction of performance for all users. One of the most famous examples of using C/S architecture is Internet infrastructure.

The peer-to-peer approach is presented in Figure 2-1 (b) [19]. In this approach, all participating nodes are equal. Nodes provide resources, which might include storage space, bandwidth, and computing power. Nodes cooperate with each other to provide storage and transport services. For example, when connection is needed, source node sends a request message to a few neighbours, asking: "Do you know the destination peer D?". The neighbours forward the request, step-by-step until the destination peer D is reached. The Total capacity of P2P system increases with the arrival of new participating nodes. Besides, in some cases this network model is more robust than C/S model when facing security breaches such as compromised or faulty nodes.

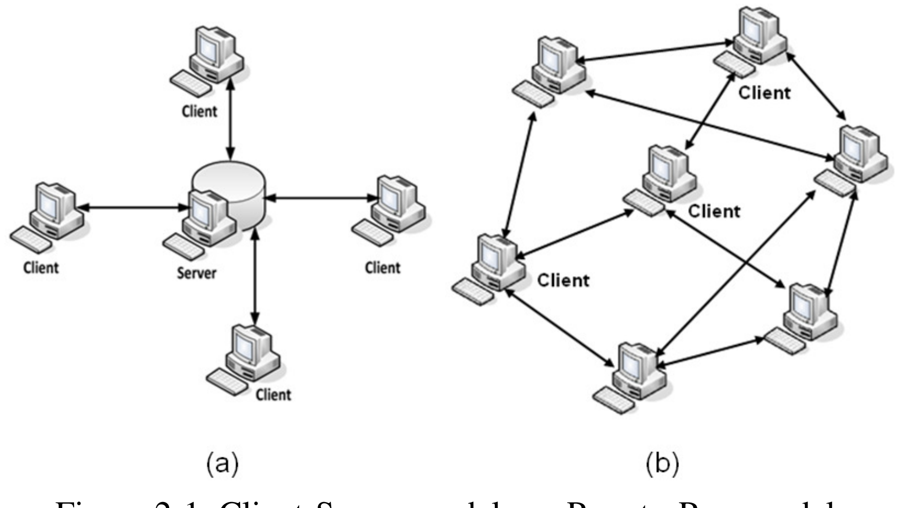


Figure 2-1: Client-Server model v.s. Peer-to-Peer model

## 2.1.2 P2P Overlay

One of the best definitions of overlay network is given in [113]: “An overlay network is virtual network of nodes and logical links that is built on top of an existing network with the purpose to implement a network service that is not available in the existing network.”

Structured overlay, especially DHT (Distributed Hash Table) technologies, has been suggested for P2PSIP communications. According to this technology, each peer maintains the state of a few neighbours. Peers in the overlay cooperate with each other to complete the lookup tasks (e.g. find a peer, resource, etc). In the following section, we specify three types of overlay algorithms and give comparison and evaluation of typical DHT overlay technologies.

### Chord

In Chord [109] overlay, peers and resources construct a ring with the space size of $[0, 2^m - 1]$, as shown in Figure 2-2 (a) where m is equal 5. In the ring, peer and resource are represented by integer Node ID or Resource ID. Each peer stores a certain amount of <*id*, *value*> pairs, in which *id* is the peer/resource ID, *value* is the peer address information or the data storage. Peer or resource ID is assigned by consistent hashing [66], e.g. based on SHA-1 algorithm, etc. For instance, the peer ID can be produced by hashing IP address of particular peer; and resource ID can be generated by hashing the data value. The Resource is stored in the first peer, whose ID is bigger or equal to Resource ID (see Figure 2-2 (c)). For example, the resources with IDs 14 and 20 are stored in peer with ID 22.

Each peer contains a routing table, called Finger table, for storing records containing routing information. The Finger table contains records about $\log N$ successors of its peer where *N* is the number of peers in the overlay (see Figure 2-2 (b)). Suppose the space size of overlay is $2^m$, for some integer *m*. Then, according to [109], the ID of the *i*-th successor a peer with ID *P* (denoted as *Succid(i)*) can be found as following:

$$Succid(i) = (P + 2^{i-1}) \bmod 2^m (0 < i \leq m)$$

Each peer contacts periodically its successors for updating corresponding Finger table. It also contacts its predecessors that are previous peers in the ring. This is useful when a peer leaves the ring and asks its predecessors to update their Finger tables.

Chord routes the message by sending messages to the next successor nearest to the destination identifier. Consider an example, where peer 3 is searching peer 28 (Figure 2-2 (d)). The peer 3 would first check its Finger table; choose a successor (peer 22) nearest to the destination, and then send a request to this successor. The peer 22 would also check its own finger table and forward the message to its successor (destination peer 28). According to the simulation result in paper [109], the average path length of Chord is $\frac{1}{2}\log N$, where $N$ is the number of peers in the overlay.

Chord also defines the advertisement function supporting joining/leaving procedure for peers. The advertisement function would tell corresponding successors and predecessors to update their Finger tables.

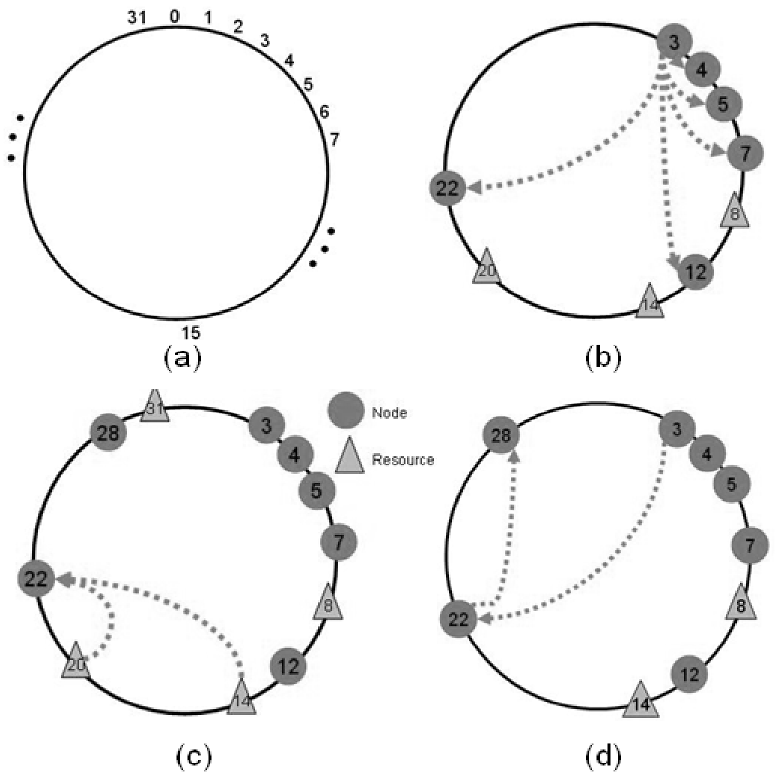


Figure 2-2: Chord Algorithm

**Kademlia**

Kademlia (Kad) is another popular DHT solution proposed in 2002 [76]. It shares some similarities with Chord. For instance, peer identifiers are produced by consistent hashing; each <*resource id, value*> pair is stored in a node with ID "close" to the *resource id*. One main difference is that Kad network is a binary tree, where peers are represented as leaves. Figure 2-3 shows a simplified structure example with 3-bit address space. Each peer and object in the Kad network has a 160-bit identifier. To construct the routing table, each peer separates this binary tree (without the peer itself) into a set of sub-trees. For instance, peer 0011 in the Figure 2-3 splits the binary tree into 4 sub-trees. Each peer maintains at least one peer profile (peer ID, IP address, port, etc) in each of the sub-trees. Kad uses XOR-based metric for calculating distance. Given two identifiers, $x$ and $y$ with peer ID 0011 and 1101, distance $d(x, y)$ between two peers is $x \oplus y = 0011 \oplus 1101 = 1110$ To route the message, source peer first analyses the distance to destination peer, and then forwards the messages to the nearest peer (with shortest distance to the destination) according to the  routing table.

Figure 2-4 shows a routing example when peer 0011 wants to contact with peer 11110 (assume that peer 101 is the only reachable peer in source peer's routing table). The request message might first go to peer 101, and be redirected to peer 1101, peer 1110, and finally to the destination peer.

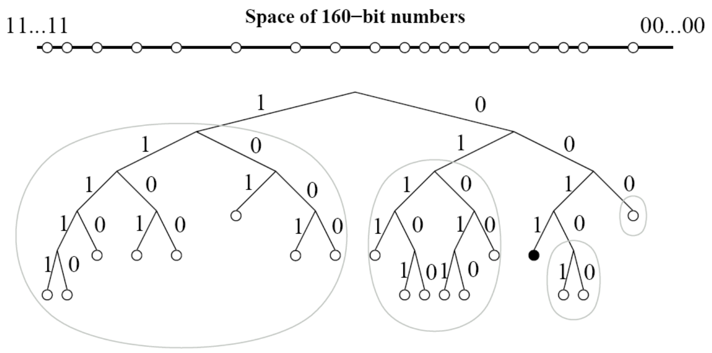


Figure 2-3: Kademlia Tree Structure [76]

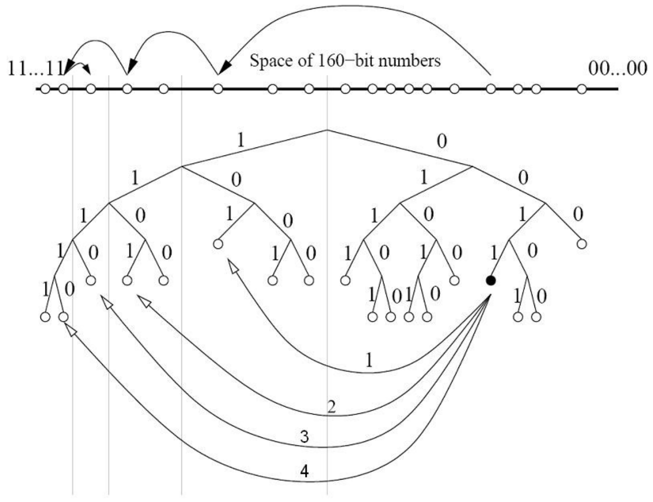


Figure 2-4: Kademlia Routing [76]

## Content Addressable Network (CAN)

CAN technology [85] shares similarities with Chord and Kad approaches. The biggest difference is that CAN uses a logical grid structure based on a virtual d-Cartesian Space. Figure 2-5 shows the simplified two-dimensional structure. Peer and resource are mapped into the space of the grid based on their coordinates. Each peer maintains a certain space around itself and stores resources within nearby coordinates. For instance, peer A is responsible to store data with the coordinate (0.7, 0.25). As to routing mechanism, each node constructs its own space routing table that records information about neighbour space (e.g. IP address, port, coordinate, etc). With this information, peers are able to forward the messages to a neighbour that is closer to the target. For instance, the request from peer H travels through a few neighbors until the destination peer J (see Figure 2-6).

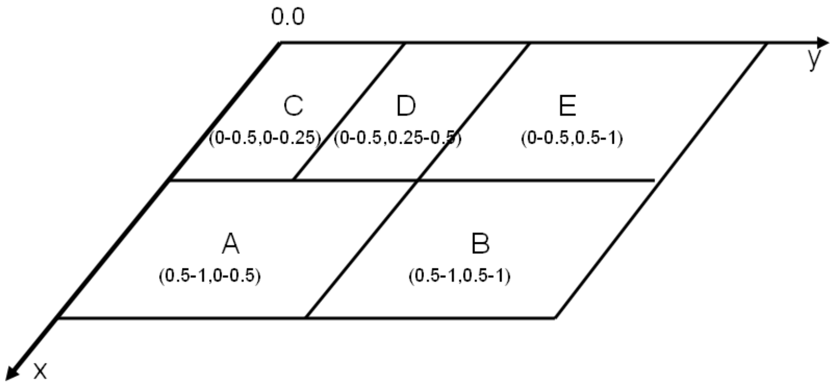


Figure 2-5: CAN Overlay

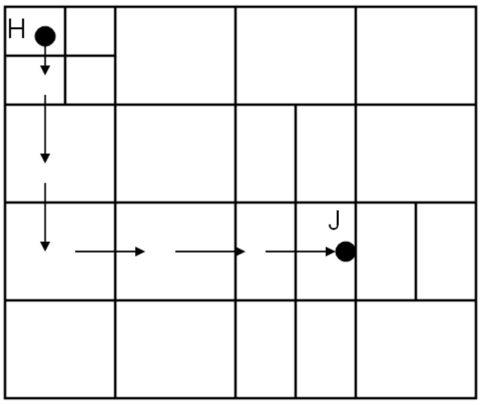


Figure 2-6: CAN Routing

## Evaluation of DHT algorithms

There are other DHT algorithms that are promising for the future P2PSIP communication, such as Pastry [93], Tapestry [117], Bamboo [51], etc. These algorithms use similar approaches as Chord and Kademlia described above. For instance, they also implement SHA-1 hashing function, have the algorithm complexity of $O(\log N)$. Table 2-1 from [55] summarizes properties of different DHT algorithms.

***Table 2-1 Overlay Technologies Comparison [55]***

| | Chord | CAN | Pastry | Bamboo | Tapestry | Kademlia |
|---|---|---|---|---|---|---|
| Lookup method | R, S-R, I | R, S-R, I | R, S-R, I | R, S-R, I | R, S-R, I | R, S-R, I |
| Complexity | simple | simple | Quite complex | Quite complex | Quite complex | Simple |
| Configuratio | A few | Many | Some | Some | A small | A few |

| | | | | | | |
|---|---|---|---|---|---|---|
| n parameters | | | | | affect | |
| Bandwidth consumption | Moderate | Moderate | High | Moderat e | Quite high | Moderate |
| Peer join/departur e | Quite simple | Very simple | Complex join | Quite simple | Complex join | Simple |
| Extendibility | Quite good | Rich already | Quite good | Quite good | Quite good | Quite good |

## 2.2. SESSION INITIATION PROTOCOL

Session Initiation Protocol (SIP) is a session layer signalling protocol for creating, modifying, and terminating sessions with other participants [54]. It inherits the simplicity from Hypertext Transport Protocol (HTTP) and Simple Mail Transfer protocol (SMTP). In November 2000, SIP was accepted as a 3GPP signalling protocol and the permanent element in future All-IP IMS architecture.

### 2.2.1 Protocol Description

SIP follows traditional Client/Server based model. A SIP network consists a few fundamental elements [90]:

- User Agent (UA). SIP User Agent is the basic element that creates, sends and receives SIP messages. It may be either hardware phones from vendors (such as Avaya [6], Cisco [7], Nortel [26], etc) or softphones (such as Windows Live Messenger [28], X/lite [8], Twinkle [24], etc). UA uses SIP identities (or SIP URI) during the communication. One typical example of using SIP URI is "sip:alice@altaland.com", where "Alice" is the user name and "altaland.com" is the domain of Alice's service provider.

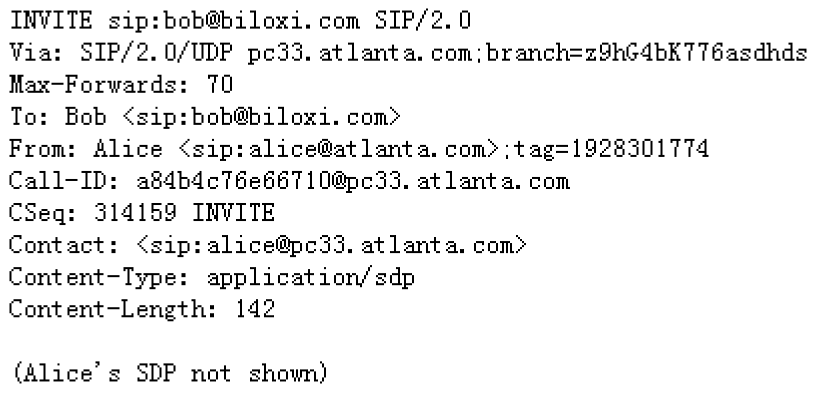
- Registrar. User Agent can not access SIP services before registering to a certain SIP network. The Registrar is a server that accepts REGISTER request and place UA's registration information into location database in the domain.

- Redirect Server. This component redirects the incoming request to another domain where destination user agents currently stay. This is useful, for example, when a user is on vocation and wishes the incoming calls to be redirected to his new places.

- Proxy Server. Proxy Server is the central element for relaying SIP traffic among different domains.

Note that the distinction among Registrar, Redirect Server, and Proxy Server is logical. Physically, they could be deployed in the same device.

The basic SIP specification RFC3261 defines some SIP methods which primarily handle "call setup" procedures, as shown in Figure 2-8. At the beginning, the user Alice sends an INVITE request (F1), which indicates destination SIP address "bob@biloxi.com", to its SIP server. An example is shown below in Figure 2-7 [90]. INVITE request can also contain service negotiation parameters (such as supported connections type, data rate, codec, etc) for session negotiation (will be specified in Section 2.2.2).



```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bK776asdhds
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.atlanta.com
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 142

(Alice's SDP not shown)
```

Figure 2-7: SIP "INVITE" Example

The atlanta server (see example in Figure 2-8) is responsible to forward the request to a corresponding SIP server (for instance, "biloxi") that is able to reach "Bob" (F2 and F4). It also returns a "100 Trying" response, indicating the request is been processed successfully (F3 and F5). After that, a "180 Ringing" message is returned (F6-F8), representing the ringing of destination client. When "Bob" decides to receive session (for example, he picks up the phone), he returns a "200 OK" response to initiator (F9-

F11). The session can be later established. F13-F14 represents the termination of the session.

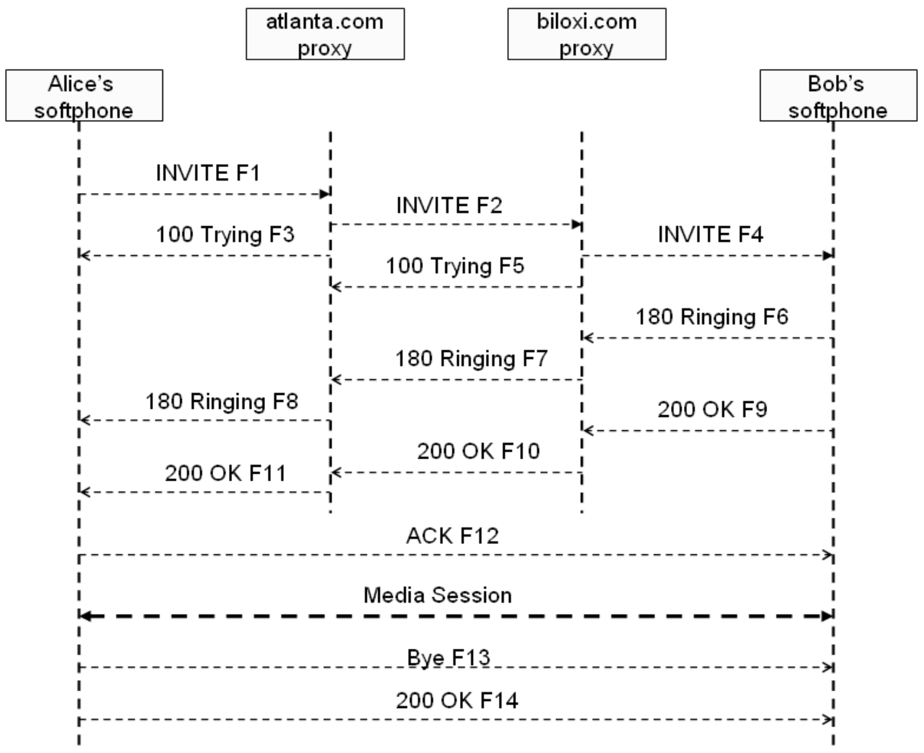


Figure 2-8: Session Initiation Protocol

A few SIP Extensions are defined to support typical use scenarios. For example, “SUBSCRIBE” and “NOTIFY” are defined in RFC3265 for supporting presence service use case [88]; “MESSAGE” is defined in RFC3428 for instant messaging use case [91].

## 2.2.2 Session Description Protocol

Session Description Protocol (SDP), defined in RFC 4566 [53], is an application layer protocol intended to describe multimedia sessions. It is used for service negotiation during the multimedia session initiation. The negotiation includes media transport protocol, media codec, encryption algorithms, etc. The following shows an example of SDP sent from Alice to Bob in the end of INVITE message. The first five lines (before the “m=” line) contains session-level information. They include the session identifier, IP address, subject, contact info, etc. The following lines illustrate the media-level information exchange for transport protocol, opening port number for audio and video:

```
v=0
o=Alice 2790844676 2867892807 IN IP4 192.0.0.1
s=Let's talk about swimming techniques
c=IN IP4 192.0.0.1
t=0 0
m=audio 20000 RTP / AVP 0
a=sendrecv
m=video 20002 RTP / AVP 31
a=sendrecv
```

## 2.3 PEER-TO-PEER SIP

P2PSIP aims to combine both the advantages of P2P and SIP protocols in a novel communication system. Known research attempts are SIPpeer and SOSIMPLE projects, both of which proposed similar solutions [36, 101, 102] where P2PSIP is built upon the SIP extension protocol (using SIP protocol for handling everything). Although the concept and solutions have been outdated, these two projects led and pushed the discussions and development in P2PSIP research.

From 2005, P2PSIP research attracted much attention both in academia and industry (Cisco, Nokia, HuaWei, Ericsson, etc). A few research projects were initiated and more and more proposals were discussed in IETF P2PSIP WG (RELOAD [59], P2PP [33], dSIP [35, 116], SEP [62], etc.). In this section, we describe the requirement of P2PSIP systems; and then discuss technical solutions according to current proposals; and finally describe the existing challenges.

### 2.3.1 Requirement

According to recent publications [37, 75, 96], the following requirements are fundamental for P2PSIP systems. The requirements are listed in priorities with respect to their importance for P2PSIP, where "A" means "already mature in research"; "B" means "already have possible solutions, however, needs extra concern"; "C" means "critical solutions missing".

**R1** Availability, Efficiency, and Stability. **Priority:** B

**Description:** The designed protocol should be able to support a certain typical P2PSIP use scenarios. Besides, suggested approaches (such as peer/resource lookup algorithm, IP layer transport and routing, etc), should be efficient and fit for user experience. Furthermore, the system should provide enough stability, even if the network system is under the environment of Churn (peer joins and leaves the overlay frequently).

**R2** DHT overlay flexibility. **Priority:** B

**Description:** Currently, Chord is suggested as the mandatory P2PSIP overlay algorithm. Considering better scalability and predictability, structured DHT overlay should be implemented. The designed peer protocol *SHOULD* be extensible to accommodate differences among overlay technologies (e.g. the existing Pastry, CAN, Kademlia, etc), including new algorithms that might appear in the future.

**R3** Inter-working. **Priority:** C

**Description:** On one hand, designed protocol *SHOULD* interwork with traditional network (e.g. PSTN/ISDN, etc); on the other hand, it should also provide the interface that interconnects with future All-IP based network (e.g. SIP, IMS, etc).

**R4** NAT Traversal. **Priority:** B

**Description:** A multitude of users are behind the protection of NAT. Therefore, to provide availability and transparency, a few suggested NAT traversal approaches, such as STUN/TURN/ICE, *SHOULD* be integrated into the design of peer protocols. , Corresponding solutions should be selected for handling NAT traversal with respect to different configurations and environments. Besides, other alternative solutions, such as UPnP based NAT traversal, need extra concern and further study.

**R5** P2PSIP Client. **Priority:** A

**Description:** IETF P2PSIP WG also proposed a new entity of P2PSIP node, called "P2PSIP Client", which are legacy devices that participate in the overlay but are excused to make contributions due to lack of the protocol support in DHT algorithm or limitation of devices capability (e.g. energy, CPU processing power, bandwidth, etc). In order to access P2PSIP services, these "Client" should affiliate and rely on a normal peer for routing and storage. Therefore, it would be necessary to define a separate "Client" protocol that associates normal peer and with access services.

**R6** Security requirement. **Priority:** A

**Description:** Security is one of the biggest challenges for peer-to-peer systems, and thus for P2PSIP. Security includes transport security, DHT overlay security, Client protocol security, and interworking interface security. P2PSIP systems can not work well before all these security problems are solved.

### 2.3.2 Current Proposals Discussion

The Internet draft [38] introduces the concept of P2PSIP, as illustrated in Figure 2-9. A typical P2PSIP overlay consists of PSTN gateway peers, SIP proxy peers, SIP redirect peers, normal peers, and clients. PSTN gateway peers are gateways that interconnect with PSTN network, SIP proxy peers and SIP redirect peers are proxies that interconnect with SIP network. Note that gateway/proxy peers have basic function of normal peer.

Peers are targeted at end devices/applications that participate in the overlay and offer distributed storage and transport services. Note that many peers are under the protection of NAT and firewall.

Clients are special entities that participate in the overlay, however, they do not provide any services for other peers. In order to access P2PSIP service, they have to find a P2PSIP peer who is willing to help and relay data traffic during connection establishment process.

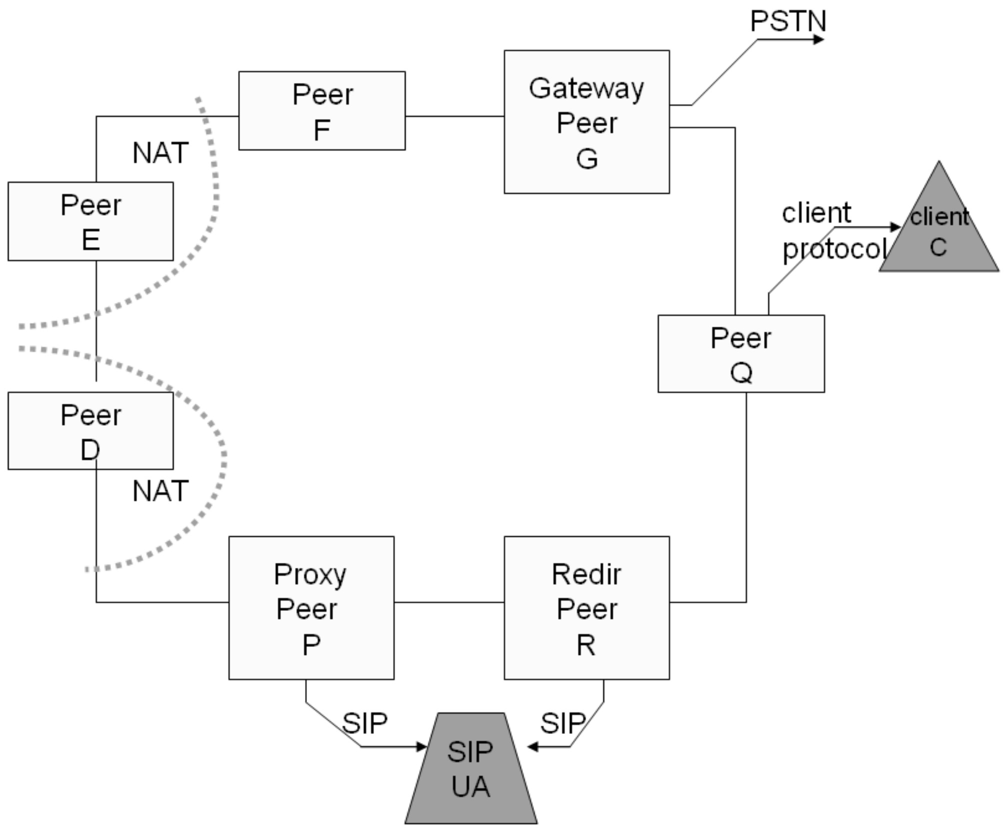


Figure 2-9: P2PSIP Overlay

There are mainly two types of functions in P2PSIP system: maintenance and lookup. Maintenance function handles with overlay activities (for example, join/leave the overlay, Finger table construction and update, etc). Lookup function sends/forwards application messages. Generally, there can be three different approaches to realize P2PSIP systems: SIP-based approach that proposes to use SIP messages for both maintenance and lookup functions; P2P-based approach which suggests a new P2P protocol to realize both functions; and compromised SIP over P2P approach that is the appropriate combination of above two solutions.

**SIP based approach**

Typical example of system that realize SIP-based approach is dSIP [35], which proposes to use SIP REGISTER request to join, build, and maintain overlay network. P2P related information (for example, peer ID) and commands (e.g. join, build, and update, etc) can be recorded in SIP extension header. Besides, SDP body is able to carry parameters for session negotiation (e.g. for setting up a media session, ICE connections, etc) or function specifications (for instance, using XML file to record service profile that is needed in service discovery and announcement).

This approach sounds feasible because SIP is originally designed as an extendable protocol. However, heavy SIP messages and frequent overlay maintenance exchange messages might bring overhead problem and add a lot of burden, especially in large overlay where thousands of peers generate a huge amount of P2P maintenance messages. Therefore, this approach is not scalable.

**P2P based approach**

Typical example of P2P-based approach is SEP [57, 62],which totally abandons the usage of SIP. Through defining a separated application layer P2P protocol, SEP achieves both maintenance and lookup functionality.

The design of SEP has a few advantages. Since the designed protocol is specific for the applications overlay, it is more flexible and efficient than traditional P2P protocol to handle the maintenance activities. For example, SEP defines a prediction mechanism, in which upstream peer will be notified if its downstream peer has no enough resource to receive request at that time. When receiving notification, upstream peer chooses another alternative downstream peer that is in good condition. SEP also provides extension interface for service discovery (e.g. some peers in the overlay might be able to provide a specific services, such as STUN functions, etc).

The main problem of SEP may be that it is pure P2P. It involves difficulties to inter-work with SIP client/application. It looks more like an open "Skype" protocol proposal. Another problem is that SEP only supports STUN solution for NAT traversal, which might not work in enterprise NAT environment.

**SIP over P2P**

A third approach is the appropriate combination of the two approaches mentioned above. The idea is to use a designed P2P protocol to maintain overlay activities, while SIP (or SIP related) usage is implemented upon P2P layer. P2P protocol defines overlay network algorithms, TCP and UDP transport protocol, variety of caching, striping, congestion control algorithms and error handling [44]. Besides, SIP related usage is supposed to handle SIP application functions, such as service description and negotiation. This approach is regarded as the most promising option. Typical examples are HIP-HOP [58, 68] and RELOAD [59].

HIP-HOP achieves message forwarding by using a new Host Identity Protocol (HIP) layer [40][79] (shown in Figure 2-10). HIP layer is in the middle of IP and transport layers (also called 3.5 layer). It defines a 40 bytes HIP header that mainly includes sender's host identity, receiver's host identity, and a few control parameters. HIP-HOP inherits the concept of ID generation mechanism from HIP protocol, which proposes to generate a 128 bits host identity (also peer ID) via consistent hashing (e.g SHA-1 algorithm, etc) peer's public key [39]. After that, the mapping of peer ID and SIP URI are distributed and stored in the overlay.

When a source peer initiates a request (for instance, sending a SIP request), it has to ask overlay about the destination peer ID via DHT lookup algorithm. Then, it generates HIP message by adding a HIP header in front of IP datagram and sends to a downstream peer according to the judgment of the Distributed DB function. The intermediate peers, when reading HIP header, can understand how to forward the request. Finally, the session between source and destination can be established.

HIP-HOP proposal makes message forwarding easier via migrating application layer message forwarding function (in RELOAD proposal) into lower HIP layer. Besides, the use of HIP offers seamless roaming support in the situation that a peer changes IP frequently.

However, one of the biggest problems is in a real implementation. The revision of IP stack is not acceptable by most routing solutions currently existing, and this might result in message lost because of confusion and misunderstanding (for example, route considers received packet as malicious).

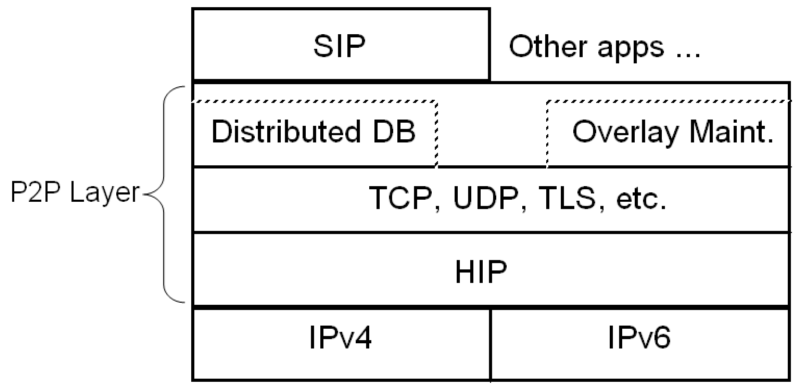


Figure 2-10: HIP Solution

Another approach called RELOAD is proposed in [59], with layer architecture illustrated in Figure 2-11. P2P layer is defined as an application layer protocol that includes four components:

- Message Transport. This component is responsible for generic key based message routing services, according to DHT algorithms.
- Storage. This is one of the basic functions for storing data in traditional P2P overlays.
- Topology Plugin. Topology Plugin handles flexible overlay algorithms (e.g. Chord algorithm in a Chord based overlay) for specific overlay. This part also deals with overlay maintenance activities. For example, when peer joins/leaves overlay, routing table is required to be reconstructed or deleted.
- Forwarding and Link Management. This part of the functionality establishes connections according to Topology Plugin and Message Transport components. Besides, it also handles ICE based NAT traversal solution.

In RELOAD proposal, each peer is represented by SIP URI and peer ID. SIP URI starts with a username (e.g. alice, etc), followed by attached overlay domain name (for example, dht.reload.com). Peer ID is a 128 or 160 bits integer randomly generated by centralized identity server. After that, the mapping of SIP URI and Peer ID is published and stored in the overlay.

When a source peer (for instance, alice@uia.no with id 20) wants to call a destination peer (for example, bob@uia.no), it has to first apply SHA-1 hash function H on bob@uia.no to compute a key H(bob@uia.no). After that, an overlay lookup algorithm is executed on H(bob@uia.no) to find ID mapping in the overlay. According to ID mapping, source peer is capable to reach destination peer ID (here dest_id) and initiate a request (via executing another lookup(dest_id) function by applying lookup algorithm to dest_id).

RELOAD provides feasible solutions for P2PSIP session initiation. However, it also has a few disadvantages. Firstly, due to complexity of ID mapping mechanism, source peer has to execute two lookup functions before locating destination peer. This may cause unacceptable delay. Secondly, RELOAD migrates most of SIP functions (e.g. routing, session description and negotiation, etc) to P2P layer. This approach, on one hand places burden on protocol design, on the other hand destroys SIP original functions. Consider an example of SIP "Via" header, which records profiles (e.g. IP, port) of intermediates nodes. In RELOAD solution, this "Via" is set to be empty because routing function is taken care by Message Transport component. This might cause problem when interconnecting with SIP network because SIP proxy will consider received message as malicious.

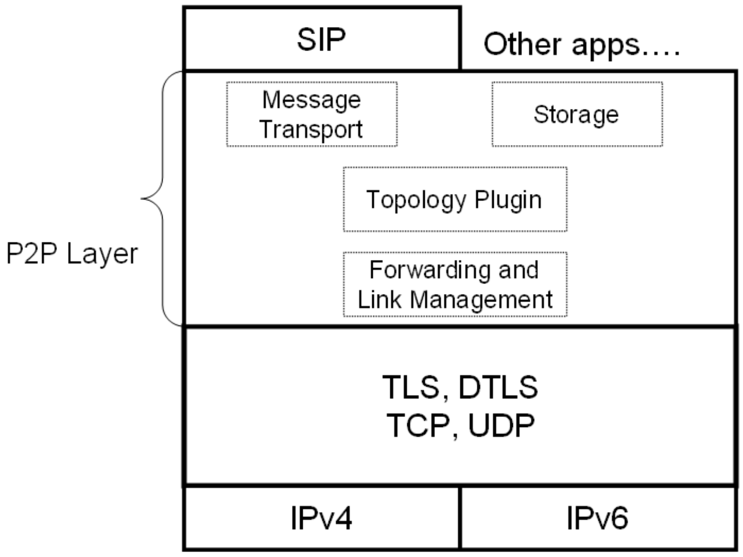


Figure 2-11: RELOAD Layer Architecture

## 2.3.3 Other Proposals

At the meantime, there are many other publications focused on P2PSIP, suggesting constructive solutions. We briefly review some of them.

Paper [70] proposes a hierarchical virtualization model, in which a P2PSIP system is logically divided into $N$ sublayers according to peer capabilities (e.g. CPU processing power, bandwidth, storage, etc). Each peer in $i-th$ layer of the overlay has one or more Son Nodes (SN) acting as contact points in $(i-1)-th$ layer. According to performance analysis, hierarchical division increases overall system capability and is capable to reduce delays when setting up connection. Besides, the system provides a "breaching" model to reduce DHT maintenance cost. Each peer has three states: *active*, *tired*, and *asleep*. *Active* is a state when peer is handling application services (e.g. sending out SIP messages, call setup, etc); *tired* state represents a state that experiences a long idle interval and becomes tired; *asleep* follows with tired state when peer "sleeps" and temporally leaves the overlay (and therefore does not need DHT maintenance).

There can be different P2PSIP domains that implement different overlay technologies (e.g Chord, Kademlia, CAN, etc). Paper [74] considers a system architecture for interconnecting among different P2PSIP domains. The idea is that each P2PSIP domain selects a super node with extra capability, and these super nodes construct a new overlay network. Super node acts as the proxy for each domain when interconnecting function is needed. Besides, P2PSIP should also handle the interconnection with future All-IP networks (e.g. SIP/IMS-based network, etc). A possible system architecture is suggested in [56], in which a Gateway Application Server (AS) is proposed as the key interworking unit between two different networks. Gateway AS acts as an ordinary P2PSIP peer in P2PSIP network and an IMS application server in IMS network.

As to security issue, paper [96] makes an overview of security problems in P2PSIP paradigm, and briefly introduces a few possible solutions. Paper [97] studies a real implementation of P2PSIP system under DoS attacks on DHT routing layer. The result shows that even in small size overlay network (only 100 peers in implementation), calling service (e.g. locating a callee, etc) is significantly degraded with simple malicious strategy.

### 2.3.4 Challenges

Critical technological challenges within P2PSIP area are still without satisfactory solutions. The first one is system efficiency. Current research suggests DHT algorithms for P2PSIP communication. However, as technology originally proposed for file sharing applications, this approach is not efficient in locating peers or resources. For example, the delay of searching a file in P2P applications (e.g. BitTorrent, BitComet, etc) could be as long as several seconds to even minutes. This is unacceptable for P2PSIP real-time applications. Therefore, it is necessary to improve conventional DHT approach, or new DHT algorithms should be developed.

Another critical issue is security. The decentralized P2P network lacks efficient credential mechanisms for authentication and authorization. Therefore, all the participating peers distrust each other, and this generates a lot of security breaches. For example, one malicious peer could pretend to be non-malicious peer, or it can act as an intermediate peer that intercepts messages and steals sensitive information. Currently, most of research is focused on technical availability, while ignoring security challenges.

Additionally, there are few research on P2PSIP extension functions focused, for example, on how  interconnect with future IMS and conventional PSTN networks efficiently and secure, and how to enable the special entity “P2PSIP Client” nodes to access services. All these questions urgently need appropriate solutions.

# Chapter 3. Improvements of P2PSIP Efficiency

Chord has been suggested by IETF P2PSIP working group as mandatory overlay technology. However, Chord has disadvantages due to a few limitations. In this chapter, we investigate several approaches that are feasible to improve Chord lookup efficiency. These approaches include peer/resource lookup algorithm revision, geographical association, cache mechanism, hierarchical layer division, and routing optimization. After that, we simulate two systems (Chord-based and improved Chord based P2PSIP systems) for comparison and evaluation.

## 3.1 BACKGROUND

P2PSIP WG has suggested Chord protocol as a mandatory underlying overlay technology. Chord allows for the available peer/resource lookup in no more than $\log N$ hops, where $N$ is the total number of peers in overlay network. In this overlay, each peer maintains a finger table that stores a few successors' connections. Chord routes a message by sending/forwarding it to the next successor, step-by-step, until the destination. However, as the protocol originally designed for background downloading applications, Chord has several disadvantages when used to support P2PSIP real time services.

Firstly, Chord lookup algorithm is unfair, especially when destination peer can be reached faster in anti-clockwise direction. For example, a peer is able to use one hop routing to access its successor, while it might take at most $\log N$ hops to access its predecessors. This causes long delay, especially in large overlay where $N$ is a big number.

Secondly, existing proposals suggest use either consistent hashing (e.g. SHA-1, etc) or random assignment (in RELOAD proposal) to generate peer identity in order to partition a keyspace so that each peer is responsible for roughly the same load of resources. However, these algorithms are unpredictable and might cause problem that geographically closed peers are assigned with different IDs that are far away from each other in overlay space. This causes long latency when setting up the connection. Therefore, the investigation of efficient topology related ID generation approaches might be required.

Thirdly, Chord does not define cache mechanism for preserving useful information for future usage (e.g. destination ID, accessible IP, port, etc). As a result, source peer has to initiate request each time even to frequently used destination.

Fourth, Chord is implemented in either iterative or recursive style (described in Chapter 3.6). However, both approaches have drawbacks. For instance, iterative routing has problem in traversing NAT. Recursive routing adds extra burden to overlay because it generates too many message flows.

In the following, we study several improvements that can be implemented to reduce connection delay, and enhance P2PSIP system efficiency. We evaluate these approaches based on the comparison in several aspects: number of hops, message flow, practical usage, and measured delay.

## 3.2 LOOKUP ALGORITHM REVISION

One simple solution is to use two-directional lookup mechanism in the search of peer/resource, called Bi-Chord [60, 61, 77]. In this solution, we suppose the overlay space size is $2^m$, each peer stores $m$ successor and $m-1$ predecessor records in its Finger table. The P2PSIP request is forwarded to one of the successors/predecessors that is clockwisely closest to the target and then forwarded step-by-step until the destination is reached.

Figure 3-1 (a) shows the connections of a peer with identifier 3. It holds five connections with its successors (peer 4, peer 5, peer 7, peer 12, and peer 21) and two connections with its predecessors (peer 1 and peer 28). For searching a peer, for instance peer 30, peer 3 firstly sends P2PSIP request to peer 28; peer 28, after checking its own finger table, forwards request to the destination peer 30, as represented in Figure 3-1 (b).

This approach reuses most of the Chord lookup algorithm and routing style. According to Chord description in Chapter 2.1, it takes $((\log N)-1)/2$ in average before the message reaches destination, where $N$ is the number of peers in the overlay.

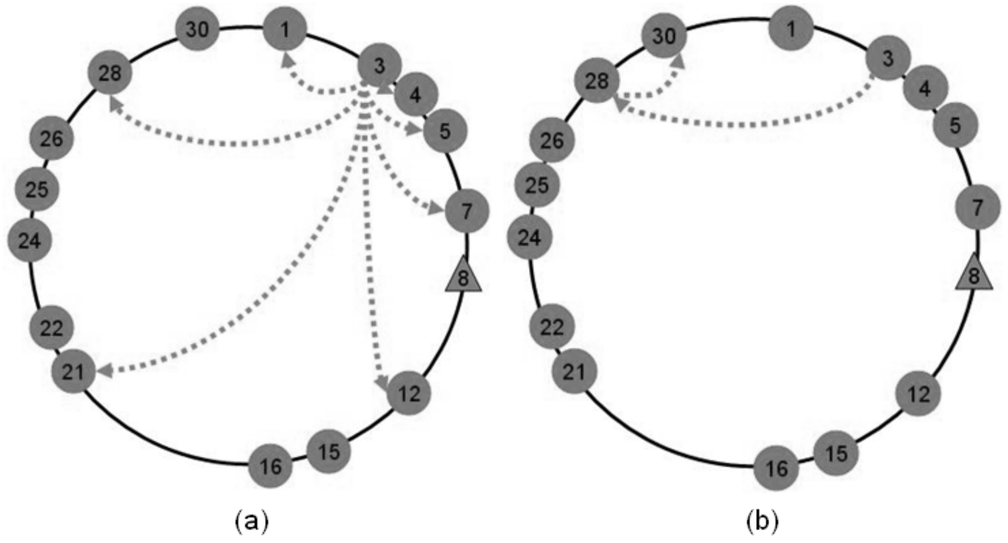


Figure 3-1: Bi-Chord Lookup

The greedy algorithm may be implemented in future to enhance Bi-Chord efficiency. In this approach, each peer maintains the same Finger table as Bi-Chord (Figure 3-2). The major difference is that each peer transmits P2PSIP request to one of its successors/predecessors that is as close as possible to the destination, independent of the clockwise or anti-clockwise direction. Each peer chooses either its successor or predecessor for routing messages, on the distance basis.

Suppose peer A wishes to initiate/forward a P2PSIP message to the destination B. It chooses the shortest path either through: (1) One of its predecessors closest to peer B, or (2) One of its successors closest to peer B. If these two choices have equal path lengths, the path selection will follow the second rule.

Figure 3-2 shows an example of the Bi-Greedy lookup initiated by peer 3 and ended in peer 24. The message is firstly routed to peer 21 (peer 3's successor), then to peer 25 (successor of peer 21), and finally to peer 24 (predecessor of 25). According to [49], the average path length of Bi-Greedy algorithm is $\log(N)/2 - \sqrt{\log(N/2\pi)} + 1$.

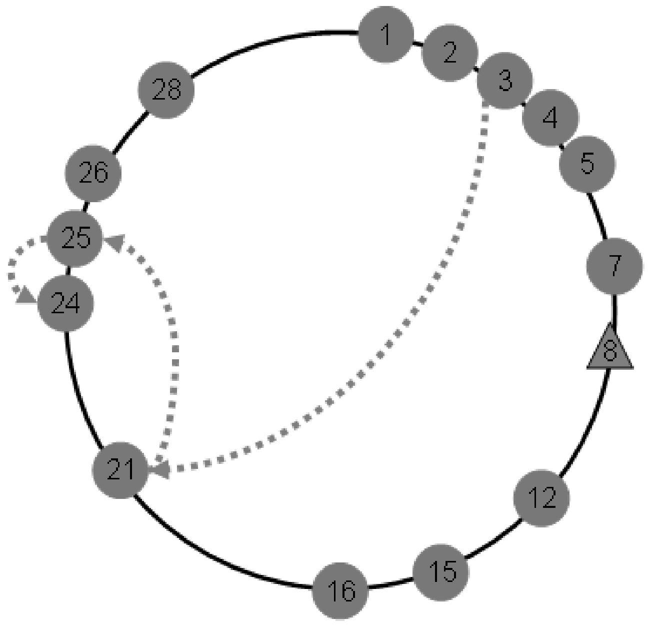


Figure 3-2: Bi-Greedy Lookup Routing

## 3.3 GEOGRAPHIC ASSOCIATION

The current peer ID generation proposals (either consistent hashing or random assignment) do not consider the relationship between network topology and peer identifier. This results in, for example, two participating peers geographically close each other in network topology (for instance, behind a same NAT), are assigned with IDs that are far in the overlay. For setting up connection, initiated P2PSIP request has to travel a long distance around overlay space, with multi-hop route traversal. This increases delay and reduces user experience.

In P2P file sharing research, there are already a few constructive proposals [57, 86, 99], introducing topology awareness solutions for generating peer identities. Generally, we advocate this concept and suggest a few revisions on that for P2PSIP overlay. We propose that geographically closed peers should be assigned with "similar" IDs that are near in overlay space. To do that, we first advocate the point in RELOAD proposal that a central ID server assigns peer ID instead of consistent hashing algorithms. Then, a landmark ID mapping mechanism is proposed. We assume there are a few landmark peers distributed evenly in the overlay, as illustrated in Figure 3-3.

The peer, who wants to join the overlay, has to use its specific peer identity from central ID server. After receiving the request, central ID server calculates the distances of this peer to each landmark peer by calculating the geographical distance within network topology. For example, the distance $D$:

$$D = | IP_{peer} - IP_{landmarkpeer} |$$

According to the calculation, central ID server computes a "nearest" landmark peer, and randomly picks up a unique identity (close to this landmark peer) for the applying peer.

Consider for example the ID application of peer *B* and peer *C*, both of which are geographically "near" from the network topology point of view (with similar public IP). After landmark peer distance calculation, central ID server picks up two unique IDs near Landmark peer 3 (L3) and assigns to peer *B* (with ID 18) and peer *C* (with ID 26). Therefore, two geographically closed peers are also close in overlay network.

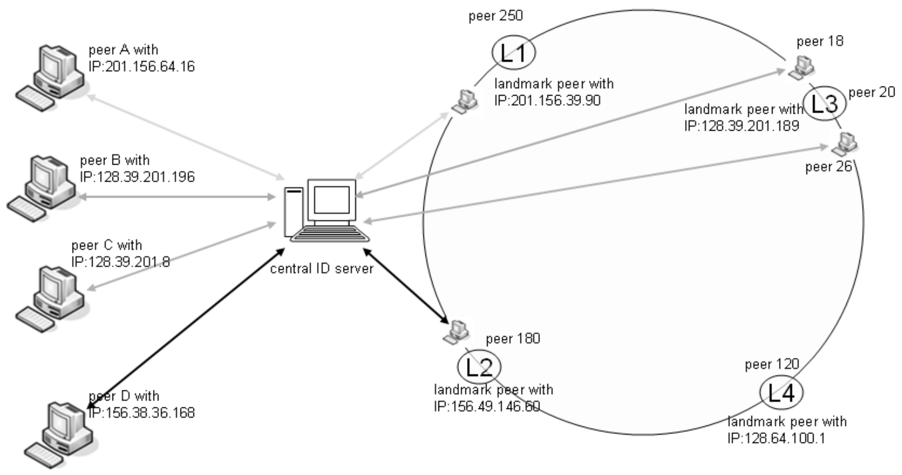


Figure 3-3: Central Identity Assignment

## 3.4 CACHE MECHANISM

In a conventional P2P network, cache mechanism is implemented to enhance the performance indirectly [34, 45, 71]. This could be reused in P2PSIP field. The concept is as following: P2PSIP peer in the overlay maintains a cache that records communication history details, including previous communicated peer identifier, corresponding public IP address, port, etc. Table 3-1 shows an example of cache entry record (of Peer C). It records *N* records of communication history. For searching destination peer, peer C first check its cache entry record. If the destination peer (peer identifier, public IP address, port, etc) is in the table, the session might be established directly. Otherwise, peer C will execute lookup algorithm described above.

In stable overlay where peers do not change IDs or public endpoints frequently, the cost is only one hop. However, in unstable overlay where peers change their IDs/IP frequently, this might even cost worse delay. It takes at most $1+\log N$ hops [109] (e.g. Bi-Chord lookup) before reaching the destination.

Table 3-1 Cache Entry Records

| Peer C | Peer Identifier | Public endpoint |
|---|---|---|
| 1 | A | 215.239.168.1:1980 |
| 2 | B | 159.250.16.2:8000 |
| .<br>. | .<br>. | .<br>. |
| N | S | 128.39.169.2:9000 |

## 3.5 HIERARCHICAL ARCHITECTURE

P2PSIP peer can be either fixed devices (e.g. desktop, web server, etc) with enough computing capabilities and high bandwidth, or it can be portable devices (e.g. mobile phone, PDA, laptop, etc) that have weak computing capabilities and slow Internet connection. Devices with weak system capabilities might cause churn problems (when peer joins and leaves the overlay frequently) [69] to the overlay. For instance, mobile phones with 3G connections might frequently join/leave overlay because of unreliable signal. Besides, one peer might not have enough CPU computing resources to support frequent P2PSIP signals, and this causes message lost. These problems might reduce overall overlay performance and decrease user experience.

Separation of an overlay into different hierarchical layers might relieve above problems [126]. The idea is shown in Figure 3-4. We divided an overlay into three sub-layers according to peer capabilities (e.g. connection type, CPU processing power, bandwidth, etc). Peers in the first sub-overlay are stable entities that have public IP addresses, powerful CPU, and stable connection. Such typical device can be a web server. The second sub-overlay contains peers with enough stability and processing power, e.g. normal PC with Internet connection. Peers in this layer do not own public IP address, and might relay on STUN/TURN/ICE for traversing NAT (described in Chapter 3.7). Peers in the lowest sub-overlay are those with unstable connections and less computing capabilities (e.g. mobile phones, PDA, laptops with wireless connection).

Each sub-overlay contains at least one gateway for handling inter-layer communication.

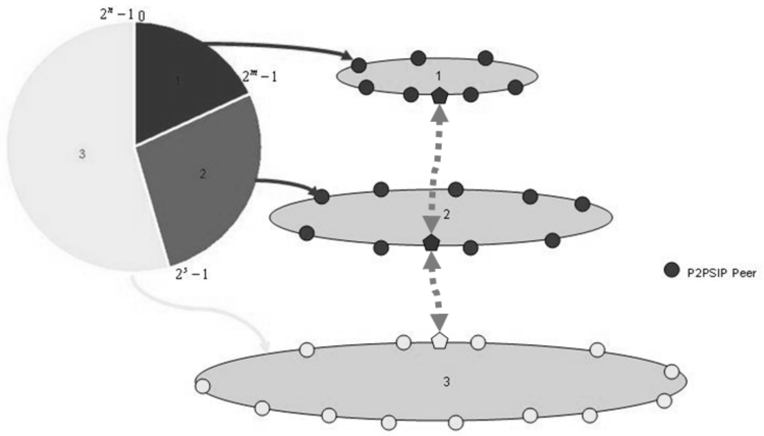


Figure 3-4: Three Layers Division Architecture

## 3.6 MESSAGE ROUTING

Generally, P2PSIP message flow in overlay network should comply with routing styles, for instance, Iterative or Recursive [36], both of which are supported by Chord technical specification. A third option “Semi-Recursive” routing is now also discussed. All of these approaches have their own advantages and disadvantages. Therefore, we advocate all three routing styles, and suggest different routing styles are implemented in different environments.

In Iterative routing, source peer S initiates a request “I need Node C” for searching the destination peer C (as shown in Figure 3-5). S then is redirected by each intermediate peer to the destination. In this approach, source peer is able to check validity and correctness of each response. It can be implemented in security sensitive environment. However, this solution does not provide guarantee for NAT traversal when destination peer is behind enterprise NAT protection.

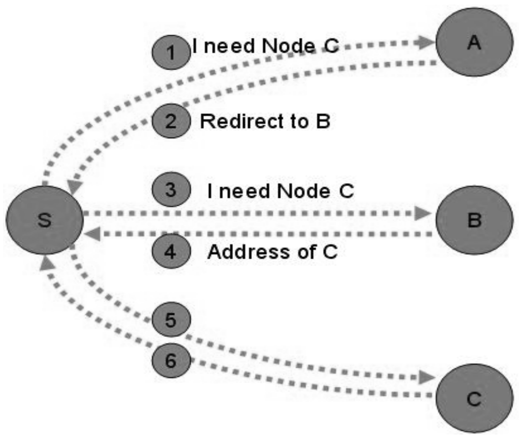


Figure 3-5: Iterative Routing

In Recursive routing, the request is forwarded hop by hop by each intermediate peer until the destination. The response follows the same route back to the source (See Figure 3-6). Recursive routing is easy for debug and has little trouble in NAT traversal. However, this approach has availability and security problems. For instance, if any one of intermediate peers fails or leaves the overlay, the routing fails. Besides, source peer is unaware about security problem in case when malicious peer sitting in the middle exposes malicious experience (e.g. misroutes the request, etc). Therefore, we only suggest this approach in the use case when the other two options are not available or debug is needed.

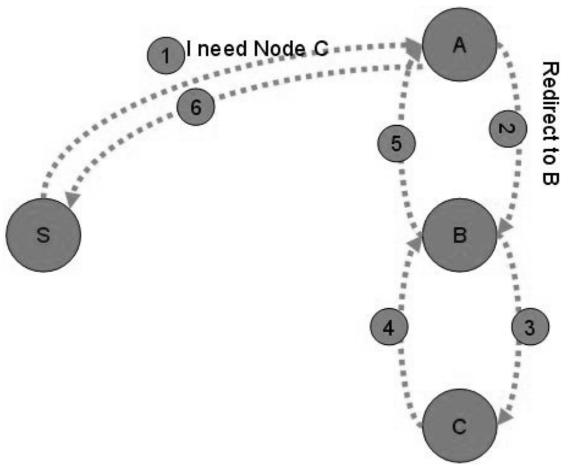


Figure 3-6: Recursive Routing

Another routing style is Semi-Recursive routing, which is the combination of above two solutions. In this approach, request message is forwarded by intermediate peers hop by hop to the destination, while the response is directly returned (See Figure 3-7). Comparing with above two solutions, this approach has the best routing efficiency because it greatly reduces number of message flows. Table 3-2 compares number of message flows between Semi-Recursive routing and Recursive (or Iterative) routing. Generally, the message number in recursive routing style is two times bigger than in Semi-Recursive (or Iterative) routing

***Table 3-2 Message Flow Comparison***

| | Recursive or Iterative Routing | Semi-Recursive Routing |
|---|---|---|
| Num-of-Message (in worst case) | $2\log N$ | $1+\log N$ |
| Num-of-Message (average) | $\log N$ | $1+\frac{1}{2}\log N$ |

Semi-Recursive routing approach provides better security than recursive routing since response is directly forwarded to source peer. It can be implemented in almost all environment. However, it still has NAT traversal problem when source peer is behind enterprise NAT environment.

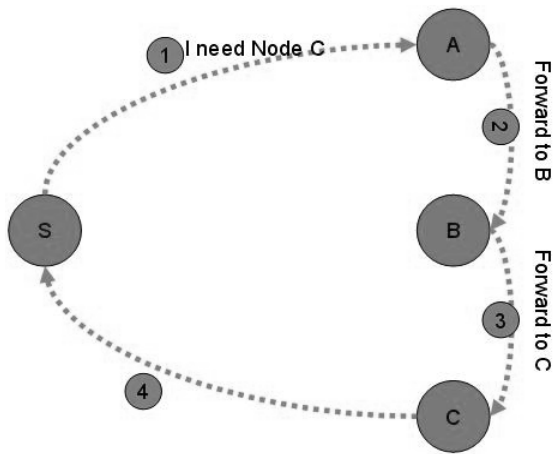


Figure 3-7: Semi-Recursive Routing

Therefore, for high security sensitive environment, we suggest Iterative routing; for efficiency sensitive environment, we suggest Semi-Recursive routing; for debugging sensitive environment, we suggest Recursive routing. Both Iterative and Semi-Recursive routing styles have NAT traversal problems. Complementary functions (will be introduced in the following section) are required to support all three message routings.

## 3.7 NAT TRAVERSAL

Network Address Translators (NAT) provides benefits (e.g. reusing IP address, constructing home network environment, etc) as well as drawbacks. One main drawback is that NAT is not friendly for connection establishment between two endpoints. In order to solve this problem, STUN/TURN/ICE-based approaches [89, 92, 110] have been proposed.

STUN approach (Figure 3-8) uses a STUN server in the middle between two endpoints to learn the NAT status (e.g. existence of NAT, NAT type, public endpoint address, port, etc). With the information, two endpoints might be able to establish the session directly. However, STUN approach does not work in symmetric NAT (enterprise NAT environment with strict security rules) where an external host (outside symmetric NAT) can not send data packet back to an internal host before receiving a packet.

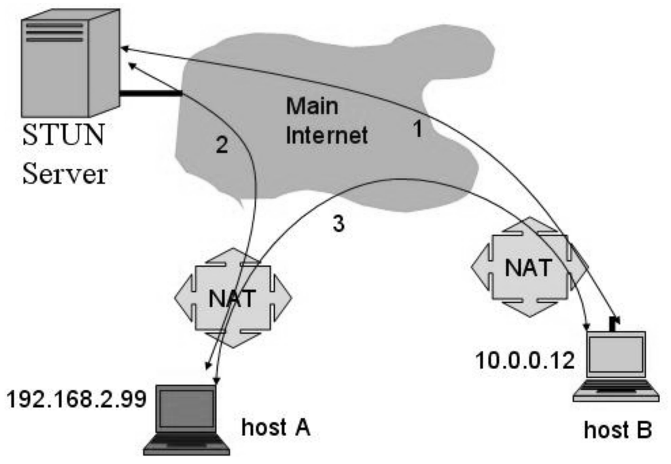


Figure 3-8: STUN NAT Solution

To solve symmetric NAT traversal problem, TURN-based approach is proposed (Figure 3-9). TURN server is a proxy that relays data traffic between two participating peers during the connection and transmission. Although this might be not efficient, it is feasible.

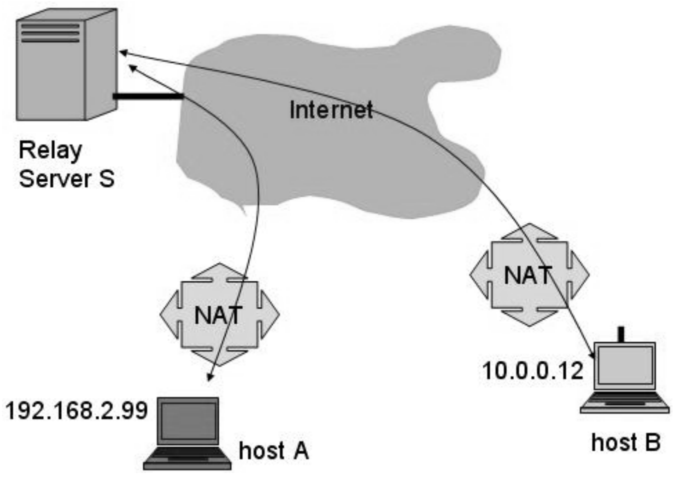


Figure 3-9: TURN NAT Solution

ICE combines the usage of STUN and TURN approaches. It firstly selects STUN for handling, while turns to TURN if STUN is not available. Besides, ICE supports session negotiation (e.g. latency, jitter measurement, error handling, best route, etc) so that connection can be established in an optimized way.

Another novel approach is based on Universal Plug and Play (UPnP) [27, 112]. In the solution, the client queries the NAT via UPnP, asking what mapping it should use if it wants to receive on a certain port. The NAT responds with the IP and port pair that can be reached from the public Internet. Today, more and more Internet gateway vendors (e.g. D-Link, Intel, Arescom, etc) offer the support of UPnP protocol, which makes this technology quite promising for P2PSIP communication systems.

## 3.8 IMPLEMENTATION

### 3.8.1 Implementation Model

Our implementation layer architecture is illustrated in Figure 3-10. P2P layer is defined to be an application layer protocol, handling overlay activities. It includes Topology Plugin component for overlay maintenance and diagnose, Storage component for data storage, and Link Management component for connections management and initiation. Besides, we propose Security Enhancement component for providing security mechanisms (will be discussed in Chapter 5).

SIP related usage, called “P2PSIP”, is built upon P2P layer. P2PSIP layer combines SIP functionality as well as part of P2P functionality. Let us consider an example presented in Figure 3-11. The major difference with SIP “INVITE” (Figure 3-11(a)) is that P2PSIP message ((Figure 3-11(b))) replaces SIP URI by peer ID (for instance, alice@atlanta.com is changed to id 20, and bob@biloxi.com is changed to id 3). Based on overlay understandable identity, Message Transport component is capable to send/forward request to a downstream peer. The message route (e.g. ID, public IP, port, etc) useful for routing and debugging is recorded in “Via” header.

Session negotiation services, such as ICE based NAT traversal, etc., are supposed to be handled by SIP related protocol, more precisely, by P2PSIP request message. The negotiation parameter can be included in SIP extension header, or encapsulated in SDP body (can be XML based). Besides, security mechanisms should be also implemented in P2PSIP usage layer (discussed in Chapter 5).

In this approach, P2P layer handles overlay maintenance functions, while P2PSIP layer is responsible for session negotiation, message routing. We believe this division preserves SIP original functions. Also, due to close relationship between P2PSIP and SIP messages, it would not be difficult to handle translation tasks when interconnecting with SIP based network.

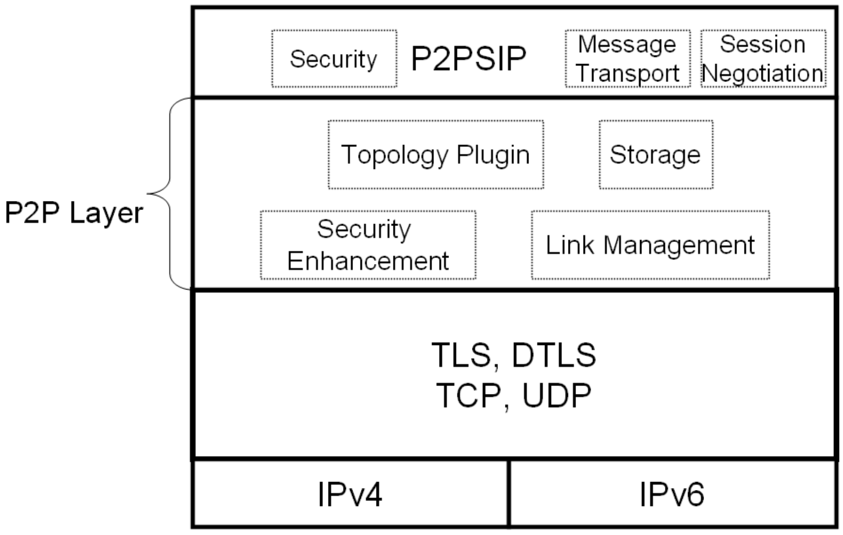
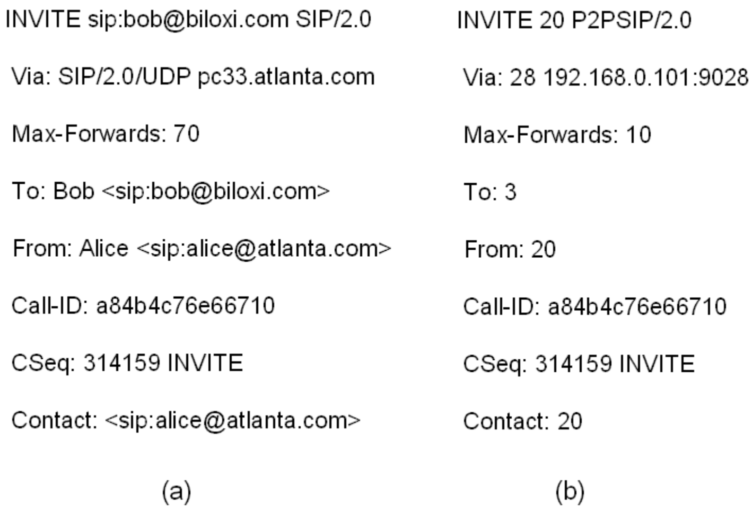


Figure 3-10: Implementation Layer Architecture



```
INVITE sip:bob@biloxi.com SIP/2.0          INVITE 20 P2PSIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com          Via: 28 192.168.0.101:9028
Max-Forwards: 70                           Max-Forwards: 10
To: Bob <sip:bob@biloxi.com>               To: 3
From: Alice <sip:alice@atlanta.com>        From: 20
Call-ID: a84b4c76e66710                    Call-ID: a84b4c76e66710
CSeq: 314159 INVITE                        CSeq: 314159 INVITE
Contact: <sip:alice@atlanta.com>           Contact: 20

                (a)                                  (b)
```

Figure 3-11: P2PSIP Request

Besides, a corresponding P2PSIP "180 Ringing" is defined as the success response. An example of such success response is presented below:

```
P2PSIP/2.0 180 Ringing
To: 20
From: 3
Contact: 20
CSeq: 1 Response
Content-Length: 0
```

Via: 3 192.168.0.101:9003; 20 192.168.0.101:9020

### 3.8.2 Implementation Description

The implementation is built with respect to the following assumptions:

- Overlay space is defined in the range of [0, 2047]. We randomly pick up 512 numbers as peer IDs of 512 peers. This simplifies ID generation algorithm and makes graphical presentation more understandable.
- Overlay maintenance activities (join/leave the overlay) are not considered and implemented. Peers in the overlay are supposed to be stable.
- Peer's neighbours are pre-configured manually in the background database.
- NAT traversal solutions are not implemented.

Firstly, we simulate a Chord-based P2PSIP system. This system contains 512 P2PSIP peers in the overlay with overlay space size 2048. We use Java to create an application that contains 512 threads, each of which represents one P2PSIP peer. We configure successors for each peer in a background database so that in initiation phase, peers could fetch their specific finger tables.

A P2PSIP Peer management center is used to create P2PSIP peer threads and configure the peer attributes (e.g. peerID and the opening port). Each peer uses the loopback address (127.0.0.1) as its IP address (can be also 192.168.0.100 in NAT) and opens a specific port (we set it as 9000+peerID) for receiving the requests.

For each P2PSIP Peer thread, it is able to send out "INVITE" message when clicking "Search" button and receive the response in the background.

Then, based on Chord-based P2PSIP system, we build an improved system with the feature of Bi-Chord, Cache entry record, and Semi-Recursive routing. The Chord lookup protocol is revised to realize Bi-Chord; the open source Apache Derby [2] is chosen as the embedded database for cache entry record implementation; finally the routing style is changed to Semi-Recursive routing. Note we do not implement the concept of hierarchical layer division because peers are assumed to be stable.

Two systems are deployed separately on a platform with Windows XP professional system, 2*2.4G Intel Core CPU and 3G memory. We use the Wireshark [29] to monitor the message transmission.

## 3.9 EVALUATION

Our evaluation is based on geographic association model (Figure 3-12), in which geographically related peers are assigned with IDs that are near in the overlay. We assume that the overlay is divided logically into $S$ equal parts and peer $M$ only communicate with peers in parts A and D.

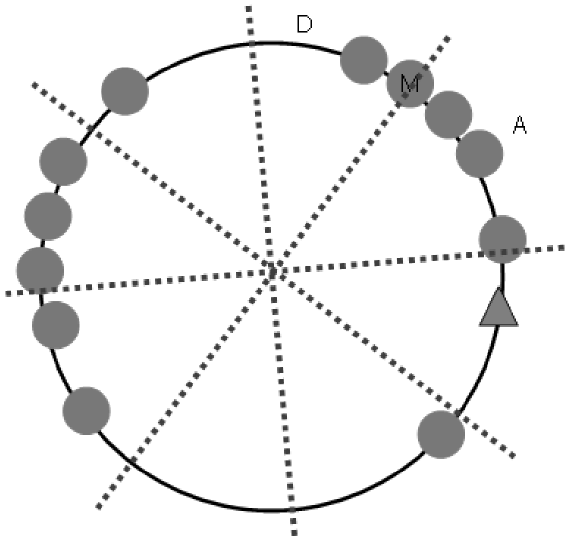


Figure 3-12: Evaluation Model

### 3.9.1 Model Analysis

According to [109], Chord lookup protocol need in average $\frac{1}{2}\log N$ number of hops to reach a destination peer, where $N$ is the number of peers in the overlay. Assume the destination peer is within the area A (that is only $1/S$ part of overlay), it takes $\frac{1}{2}\log(\frac{N}{S})$ number of hops in average for the source peer M to route the message.

Also according to Chord standard [109], it takes $\log S$ hops before the message arrives to the left boundary of area D. Therefore, the average number of hops when source peer $M$ will reach a peer in the area D is:

$$\frac{1}{2}\log(\frac{N}{S})+\log S\cdot$$

Since Bi-Chord algorithm provides fairness in bi-directional peer/resource lookup, the average number of hops is $\frac{1}{2}\log(\frac{N}{S})$ in both area A and D.

Bi-Greedy further improves lookup efficiency. According to previous research [49], the average number of hops is:

$$\frac{1}{2}\log(\frac{N}{S})-\sqrt{\log(\frac{N}{2\pi S})}+1$$

We compare three lookup algorithms by setting different S value (e.g. we set $S = 8$ and $S = 16$ for example), as represented in Figure 3-13. X axis represents the peer number in the overlay and Y axis represents the number of hops in average. We get the information that firstly, Bi-Chord and Bi-Greedy lookup algorithms are much more efficient than the original Chord lookup; secondly, the higher value of $S$, the smaller number of hops; thirdly, Bi-Greedy approach provides better result than Bi-Chord.

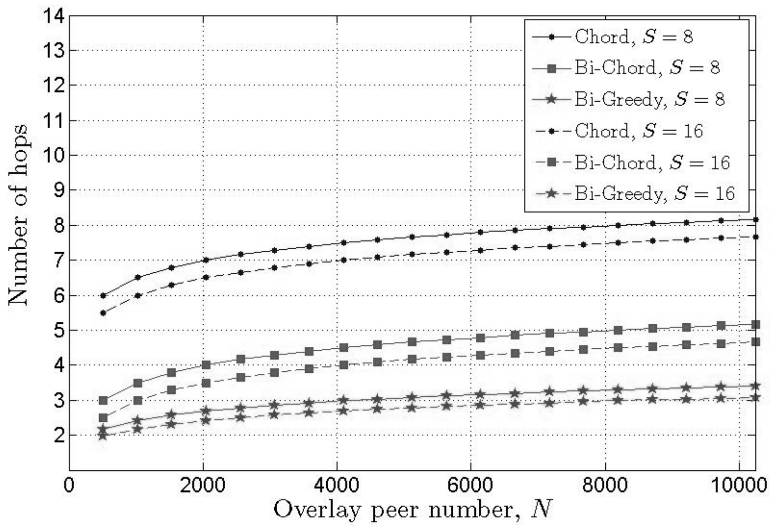


Figure 3-13: Three Algorithms Comparison

### 3.9.2 Feasibility

Bi-Chord approach preserves most of the original Chord lookup and routing mechanisms. It should be easy to implement in reality. However, Bi-Greedy does not always better. The enhancement of Bi-Greedy algorithm is limited comparing with Bi-Chord (about one hop difference). Additionally, the malicious peer in Bi-Greedy approach will be able to send/forward the message in both directions and makes the debug and trace even more difficult. Furthermore, the additional functionality to accurate distance calculation and comparison might on the other way, add the burden for P2PSIP peer and offset the enhancement of Bi-Greedy. In summary, we advocate Bi-Chord approach.

Cache record entry approach is a common solution in today's applications and systems for recording useful data, e.g. communication history, etc.

### 3.9.3 Delay Testing

Finally, the delays in two systems are measured. We choose one peer (here we use peer 586 for example) as the source peer and select 8 destination peer groups, each of which contains 32 random selected peers in either district A or D. (We assume that *S*=8 and therefore both of district A and D have 64 P2PSIP peers). For each system, we initiate 32*8 P2PSIP requests (from group 1 to group 8) manually from peer 586 and measure the delays of the responses. After that, we calculate the average delay for each group, as represented in Figure 3-14.

In Chord-based P2PSIP system, the delay of each group is almost the same (about 15ms-16ms); however, in the improved Chord based P2PSIP system, the delay is greatly reduced, especially when the number of testing increases. We believe it is the contribution of the Cache entry records.

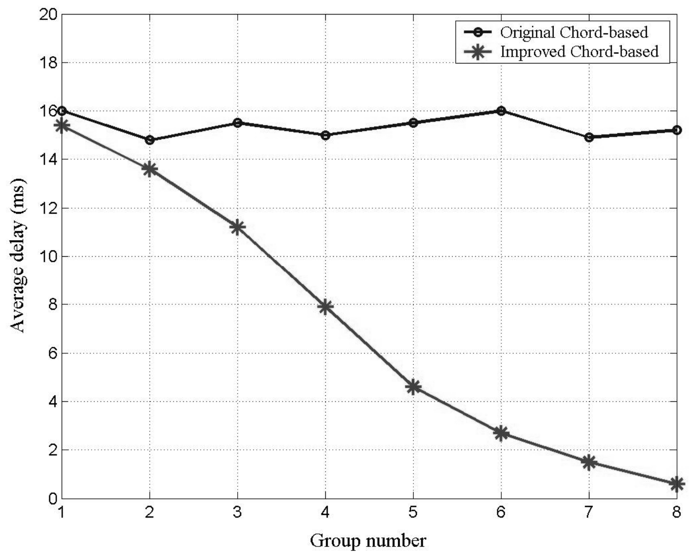


Figure 3-14: Delay Comparison

## 3.10 CONCLUSIONS

In this chapter, we study several approaches to improve peer/resource lookup efficiency of Chord protocol. After considering all aspects, we came to conclusion that the combination of Bi-Chord, Cache entry record, Geographical association, and Semi-Recursive routing might be one of the best options for P2PSIP systems to reduce number of hops and delay during session initiation.

# Chapter 4. Security Challenge of P2PSIP

Security is one of biggest challenges in P2PSIP systems. The decentralized nature of P2P comes to the cost of reduced manageability and therefore causes security problems, e.g. distrust, privacy leaks, unpredictable availability, etc. This chapter introduces security problems, including general security problems and P2P specific problems; and then suggests possible solutions suggested.

## 4.1 GENERIC SECURITY PROBLEMS

Generic security problems are vulnerabilities appearing in the most of networking systems. In the following, we specify three kinds of generic security problems.

### Denial-Of-Service attack

Denial-Of-Service (DoS) attack is an attempt to make a computer resource unavailable to its intended users [80]. Generally, DoS attack is implemented by either forcing the target to reset or consuming its resource so that the victim is not able to offer intended services. In P2PSIP network, a malicious peer could flood a multitude of P2PSIP request to one or more peers. This might consume the computing resource or prevent legitimate network traffic. The flooding can be also overlay maintenance packets, which might endanger overlay performance.

Distributed Denial-Of-Service (DDoS) [78] is the evolution of DoS when a multitude of compromised systems are involved into the attack. DDoS attack depends on a wide range of victim machines remote controlled by malicious program, called "Trojan horse" [23]. The "Trojan horse" can be remotely activated and direct an attack to a certain peers or a part of overlay network.

DoS and DDoS attacks are big issues to most of network systems. Efficient credential mechanism is able to reduce these attacks; however, it is difficult to eliminate them.

### Man-in-the-middle Attack

Man-in-the middle (MiM) attack [83] is a form of active eavesdropping. It might happen when an attacker impersonates enough sensitive information (e.g. IP, port, secret, etc) of endpoints that are talking each other. The attacker splits a normal connection into two separate tunnels, however, makes victims believe they are talking over a private connection, as illustrated in Figure 4-1. After that, the attacker M is capable to intercept all the messages going between two victims and send whatever response he wants.

In P2PSIP network, each participating peer must help each other in routing and storage. This gives great opportunity for malicious intermediate peer to invade privacy of other peers. The information can be used to initiate MiM attack during interaction of victims.

Appropriate authentication mechanisms should be implemented to reduce the impact of this attack.

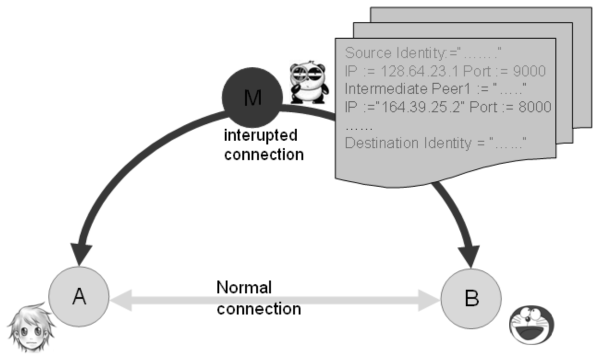

M
interupted connection
Source Identity:="......."
IP := 128.64.23.1 Port := 9000
Intermediate Peer1 := "....."
IP :="164.39.25.2" Port := 8000
......
Destination Identity = "......"
A
Normal connection
B

Figure 4-1: Example of Man-in-the-Middle Attack

### Worm Propagation

Worm [107] is a self-replicating malware computer program. It spreads by exploiting vulnerabilities in software or operating systems. Worms propagating through P2P systems and applications might be disastrous. Since all the computers in P2P network are running the same (or similar) software, an attacker might compromise the entire overlay by finding only one exploitable security breach. Besides, it is much easier to propagate worm application because each P2P peer maintains a list of neighbouring peers, and these peers are considered trusted each other. Moreover, most peers are personal computers in real life, therefore, worm program is more likely to catch person private data (e.g. credit card numbers, user name, passwords, etc), which is attractive to attackers.

## 4.2 P2P Specific Security Challenges

Besides generic secure threats, P2PSIP systems also face specific security problems caused by P2P decentralized nature or specific functions proposed. Security problems include identity attack, overlay attack, data attack, Spam over Internet (SPIT), and other malicious behaviour. In this section, we specify these attacks.

### 4.2.1 Identity Attacks

To participate in the overlay, each peer applies its unique identity (for instance, received from an E&A centralized server). However, such identity might be misused. Following are description of such identity-based attacks.

1) Sybil attack [46]

A malicious attacker can create many identities and use them to join the overlay network. If these identities become valid, they can gain control on a part of the network. These malicious entities are capable to compromise the network through malicious behaviours (e.g. compromise message routing, delete storage, etc).

2) Eclipse attack [100]

Eclipse attack is closely related to Sybil attack. Intermediate peers can conspire to hijack and dominate the neighbour set of correct peers by controlling the data traffic through routing.

3) Identity hijack [96]

Source peer sends out a request by forwarding the request to its neighbour peer who is nearest to the destination. However, a malicious neighbour is capable to intercept request message and responds to source that he is the destination peer. By pretending to be destination peer, the attacker can hijack a connection at setup time.

### 4.2.2 Overlay attacks

Overlay functions include maintenance activities, such as peer join/leave management, routing table construction and updates, link management, etc. A malicious peer might exploit vulnerability of these operations and initiate malicious attacks.

1) Free ridding [96, 114]

Malicious peer might use P2PSIP services while refuse to provide reasonable contribution to overlay. For example, it might refuse to relay message for the other peers, which causes message lost. Also, it might refuse storage request from other peers by reducing/deleting its storage cache.

2) Join-leave attack [96]

Joining and leaving overlay would generate a series of maintenance messages to neighbouring peers (for example, notification to successors and predecessors to update their routing table). A malicious peer might compromise the overlay by generating and distributing a multitude of join-leave messages to confuse the overlay, and consume the resource of neighbouring peers.

### 4.2.3 Data Attacks

Let us consider a typical malicious behaviour model in Figure 4-2, which represents a typical P2PSIP session initiation interaction between source peer A and destination peer D. There are non-malicious peer (the cat B) and malicious peer (the panda C) as the intermediate peers. This example illustrates the following possible security related behaviour.

1) Data Temper

For example, peer C is capable to drop, misroute, and modify the message it received.

2) Replay Attacks

The malicious peer can retransmit the previous message to confuse the overlay or replace newer data with old information.

3) Privacy leaks

Let us take a look at an example of possible "P2PSIP INVITE" message presented on Figure 4-3. The "From" header indicates the source peer identity; the "To" header shows the destination peer identity; the "Via" header stores the identity/address of previous intermediate peers. This information may cause privacy violation. For instance, malicious peer C is capable to record a profile of source and destination peers (e.g. Identifier, IP, Port, etc.) through parsing the incoming P2PSIP messages. This information can be used to initiate a DoS attack on a peer or overlay network. It might also be sold to illegal advertisement parties, which results in spam messages and calls.

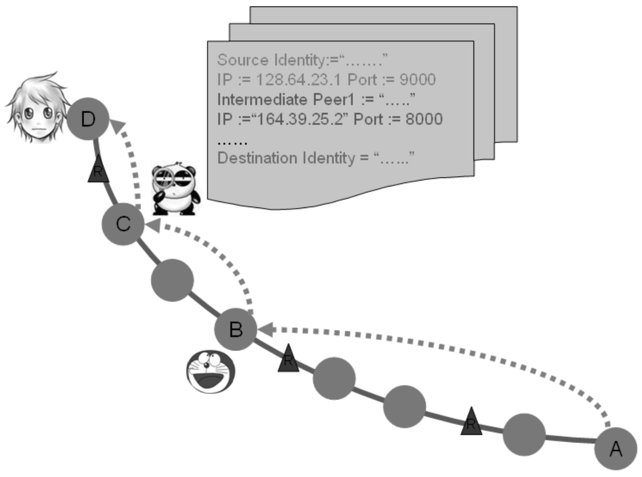
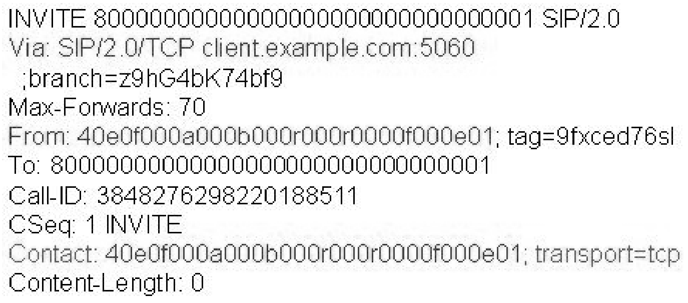

Source Identity:="……."
IP := 128.64.23.1 Port := 9000
Intermediate Peer1 := "….."
IP :="164.39.25.2" Port := 8000
……
Destination Identity = "……"
D
R
C
B
R
R
A

Figure 4-2: A Malicious Behaviour Model



```
INVITE 80000000000000000000000000000001 SIP/2.0
Via: SIP/2.0/TCP client.example.com:5060
  ;branch=z9hG4bK74bf9
Max-Forwards: 70
From: 40e0f000a000b000r000r0000f000e01; tag=9fxced76sl
To: 80000000000000000000000000000001
Call-ID: 3848276298220188511
CSeq: 1 INVITE
Contact: 40e0f000a000b000r000r0000f000e01; transport=tcp
Content-Length: 0
```

Figure 4-3: An example of P2PSIP Request

### 4.2.4 SPIT Attacks

Similarly to junk mail (or SPAM), SPAM over Internet Telephony (SPIT) [94] greatly degrades user experience. SPIT can be generated by an advertising agent who randomly selects a party to call for advertisements. The impact of this attack primarily depends on resources available to the advertising agency (e.g. the number of employees, etc). Another type of SPIT can be generated by computer software. It systematically or randomly selects one or more parties and transmits pre-recorded advertising message once connection is established. This annoys users because P2PSIP based devices might ring anytime.

As a type of synchronous communication, SPIT is more difficult to prevent than traditional asynchronous E-mail systems because there is not as much time to apply filter mechanism during communication establishment.

### 4.2.5 Other threats

**Anonymity** [96]

P2P systems allow anonymity during communication. The easiest way to do this is just hide source identifier in P2PSIP request. However, the destination peer, who does not know the source, might regard such request as malicious and refuse to receive it.

**Lawful interception** [95]

A Lawful Interception activity gets triggered by a Law Enforcement Agency (LEA) which authorises a Network Operator, Access Provider, or Service Provider to intercept traffic for a target identity. Lawful interception in P2PSIP systems, to some degree, damages the privacy because all information, including source identity, destination identity, calls duration, and other signalling are intercepted.

## 4.3 SECURITY SOLUTIONS

To overcome security threats, a few approaches have been suggested. However, these approaches are not capable to completely eliminate the problems but they relieve them.

### 4.3.1 Certificate based security

Public Key infrastructure (PKI) based certificate is supposed to be implemented [41]. Certificate is issued by a credential server, which is the Certification Authority (CA) that might collocate with enrolment server (as illustrated in Figure 4-4). It proves the existence and legitimacy of the specific peer so that the communication session is trustful. In addition to a few basic elements (e.g. version number, signature algorithm, digital signature of the issuer, etc), P2PSIP peer certificate might include P2PSIP related information: peer specific ID and one or more user names (e.g. alice@operator.com, etc). Public and private keys are used to handle the task of encryption and decryption.

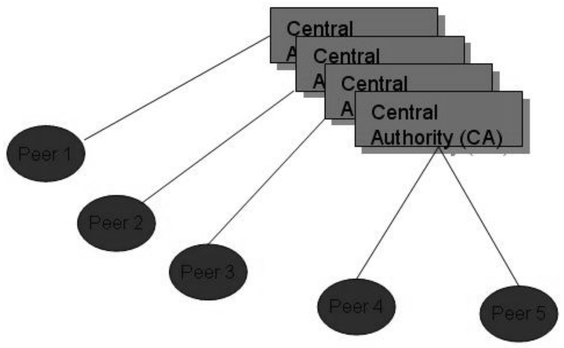


Figure 4-4: PKI Certificate Architecture

Another usage of certificate is for digital signature. Since the ownership of the private key is bound to a specific peer, a valid signature proves that the message is sent by this peer.

### 4.3.2 Pre-Shared Key Based Security

In closed or ephemerals network, pre-shared key (PSK) approach [47] can be more convenient. Pre-shared keys are symmetric keys shared among the peers in advance to establish secure connection. It can be a password like “hElLo#QWoRld”, a passphrase like “Wo ai ni”, or a hexadecimal string like “AUS30209-DOP745”. The secret is used by all the peers in the overlay to secure data traffic among each other. The PSK exchange algorithm is described below:

1. Client (request peer) send a “ClientHello” Message to the server (organized peer), indicating its willingness to use pre-shared key authentication. “ClientHello” message also includes one or more PSK ciphersuites it supports.

2. The server generates a “ServerHelloDone” message by placing one of the PSK ciphersuites and an appropriate ServerKeyExchange message (multi pre-shared key choices), and sends to the client side.

3. The client includes a ClientKeyExchange message (showing which key to choose) in the “Finished” message and sends to the server.

4. Server returns a “Finished” acknowledgement.

Using pre-shared keys can help to avoid the need for public key operations, which is especially efficient if security solution (e.g. TLS, etc) is implemented in performance-constrained environment with limited CPU computing capability. Besides, it is more convenient to configure a PSK than to use certificate since in closed environments connections are configured mostly manually in advance. However, pre-shared key can only provide limited security. For instance, an attacker could initiate a DoS attack by sending a larger mount of exchange key request to a peer. Also, it lacks efficient mechanisms to prevent MiM and replay attacks.

## 4.4 EXISTING SECURITY CHALLENGES

P2PSIP paradigm still faces serious security breaches. PKI approach is capable to reduce generic security threats (such as DoS attack, MITM attack, etc). Its authentication mechanism also reduces identity attacks (e.g. sybil, eclipse, etc).

However, certificate does not protect the system from overlay attacks, data attacks, SPIT attacks and other malicious kinds of behaviours. For example, a malicious peer might pretend to be non-malicious and get legal certificate. After that, it is able to join the overlay and expose many kinds of malicious behaviours. Generally, all participating peers in P2PSIP system should distrust each other (at least in the beginning).

Therefore, to provide security guarantee, it is necessary to explore some approaches that could either improve the trustworthiness level among participating peers or prevent malicious peers for exposing malicious experience. In the next chapter, we study a few possible approaches to improve the P2PSIP system security.

# Chapter 5. Security Enhancement of P2PSIP

The key security problem is the distrust among participating P2PSIP peers. In order to solve the problem, this chapter describes three solutions for security enhancement during P2PSIP session establishment. These solutions include: proxy based security, subjective logic based trust enhancement, and the combination of these two. The corresponding use scenarios are also described.

## 5.1. PROXY BASED SECURITY

Proxy based architecture is used in a huge variety of networking systems and applications. A proxy is an intermediary entity that intercepts communication and performs necessary services on behalf of network system. In most cases, proxy acts as a protocol translator, content adapter, security and privacy provider.

The original intention of P2PSIP is to eliminate the need of centralized entities. However, researchers begin to realize that this is not trivial, especially when considering security issues. Due to the distrust among participating peers, the session initiation process from the beginning and the subsequent data traffic are distrusted and insecure.

In this section, a possible proxy based security framework is proposed. The proposed proxy solution should, on one hand protect security and privacy, and on the other hand not add much burden in system efficiency. Proxy entities must be assumed pre-configured and pre-exist at the backbone of overlay.

### 5.1.1 Proxy-based Architecture

Our proposed architecture involves three main parts: P2PSIP Peer, Resource, Chord Secure Proxy (CSP), as shown in Figure 5-1.

P2PSIP peer, which can be a mobile phone, laptop, PC, etc., is connected to the Internet. Resource is the data value stored in a particular peer. Each peer and resource is identified by an integer ID. Chord Secure Proxy (CSP) is the secure and trusted proxy server as well as a preconfigured P2PSIP peer in the overlay.

For locating a peer/resource in the overlay, the source peer first sends P2PSIP request to a specific CSP that is the nearest to the destination. We logically regard this part of the network as source network (Step 1). The CSP acts as a proxy server to probe the existence of the destination peer and securely forward P2PSIP request message to the destination peer (Step 2). We logically consider this part of network as the destination network. After locating the destination peer, the connection can be established (Step 3).

Note that all the connections are SSL/TLS secured.

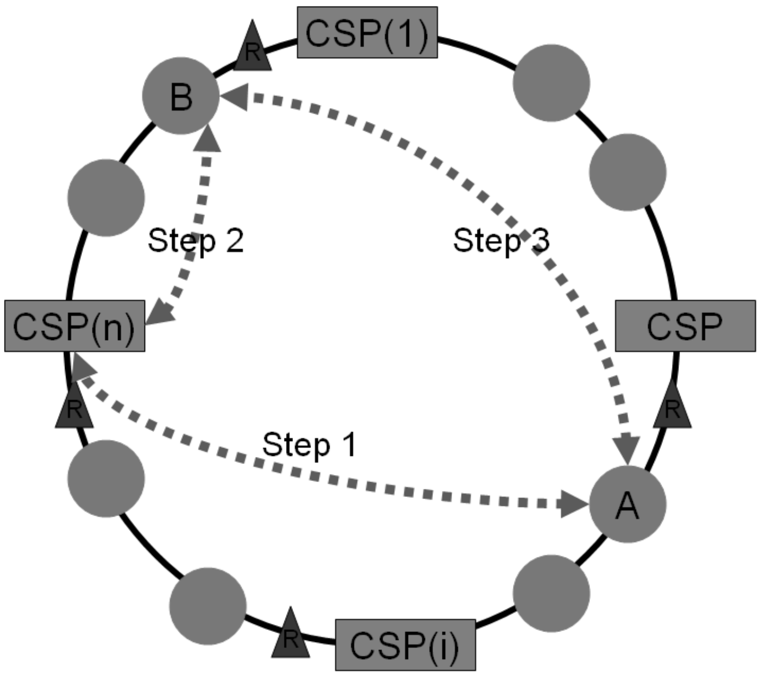


Figure 5-1: Proxy-based Architecture Overview

Chord Secure Proxy (CSP) is the key inter-working unit, acting as a bridge between source peer and destination peer. It is deployed as P2PSIP application server with the functionality of a normal P2PSIP peer. There are four main components inside a CSP unit (see Figure 5-2):

- Source inter-working. This part receives P2PSIP request from source peer. Based on security requirement in the request, source inter-working component chooses corresponding handling strategy.

- Policy management. This part is the decision center to decide which type of secure service should be handled. Inside this component, there is a policy database recording all the policy items.

- Encryption & Decryption. This component helps CSP to encrypt outgoing messages and decrypt incoming messages.

- Destination Inter-working. Destination inter-working component is the portal function that probes destination peer, and forwards the P2PSIP request. The UA subcomponent acts as normal P2PSIP peer; B2BUA subcomponent handles the secure services in the destination network.

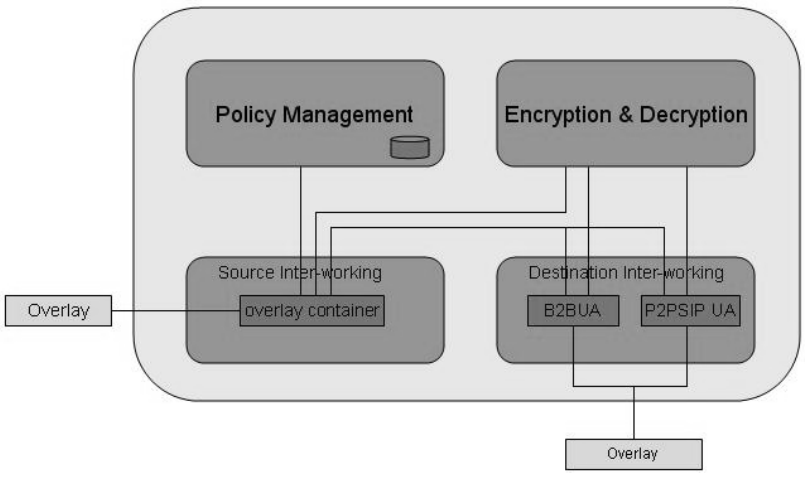


Figure 5-2: Chord Secure Proxy Internals

### 5.1.2 Security

The communications in the source network and the destination network should be securely encrypted. P2PSIP WG [38] has suggested PKI-based certificate approach to provide security guarantees and functionalities such as encryption, decryption, digital signature, etc. This can be reused in our proposed system.

### 5.1.3 Source Inter-working

Source Inter-working component contains an overlay container for receiving P2PSIP requests. With the assistance of Encryption & Decryption component, it is able to encrypt the outgoing response and decrypt the arriving requests. Besides, based on secure requirement from the message, Inter-working component turns to Policy Management component for the corresponding handling mechanism and delivers the message to the corresponding component in destination Inter-working component. Moreover, source inter-working component is responsible to response with the error code if exception happens in source network (e.g. the request message is in bad format; secure service request is not understood, etc).

### 5.1.4 Policy Management

Policy management contains a policy database that stores requirement items of how to handle the secure services. A new P2PSIP extension header is defined to include the secure service requirement from the P2PSIP peers. The secure header starts with a header field “Secure”, and follows the corresponding value. We suggest that the system should at least support three types of different secure services:

**"none" / "critical" / “anonymous”**

where

**none** means that the user requests no security  for this message request regardless of any pre-provisioned profile or default requirement of the device. The overlay peer can specify this option when the system does not require secure service.

**critical** indicates that the secure services are critical in the session. The CSP should ensure the data confidentiality, integrity, and hide source privacy before the destination peer is authenticated. The request should be rejected if service can not be supported.

**anonymous** value requests that the CSP should hide all the source sensitive privacy information from the other peers, including the destination peer. The request should be rejected if service can not be supported.

### 5.1.5 Destination Network

We logically define destination network that represents the connections from the specific CSP to the destination peer. The idea is that CSP multicasts “HelloRequest” message (similar with ICMP message [84]) to CSP’s successors in the anti-clockwise direction of the destination peer. These successors forward the received “HelloRequest” based on the original Chord lookup mechanism until they reach the destination peer. This causes that the destination peer might receive several identical “HelloRequest” messages from different routes in a certain time period. Then, the destination peer randomly chooses one of the routes and return “HelloResponse” to the specific CSP. The “HelloRequest” and “HelloResponse” message can be sent by either TCP or UDP.

Multicast mechanism causes more data traffic than the original Chord lookup. However, it on the other hand makes the system more resilient to the failure and malicious experience (e.g. discard, misroute, temper, modify the received message, etc) of the intermediate peer.

We define the structure of “HelloRequest” and “HelloResponse” messages (See Figure 5-3 and Figure 5-4) that include three fields (TOS, Code, Checksum) and five fields of P2P information (Call-ID, CSP Identifier, CSP public address and port, Destination Identifier). In the “HelloResponse” message, two more fields (Destination peer public address and port) are added. The descriptions of these fields are as following:

**TOS**: describes the service type of this message. For instance, we can define 8 as the “HelloRequest” and 0 as the “HelloResponse”.

**Code**: this is the further specification of the "Hello" message. For example, an unreachable destination might have this field set from 1 to 15. Each different number represents different error types.

**Checksum**: this field contains error checking of data from the whole "Hello" message.

**Call-ID:** a random number for identify "Hello" message.

**CSP Identifier:** an Integer ID of CSP.

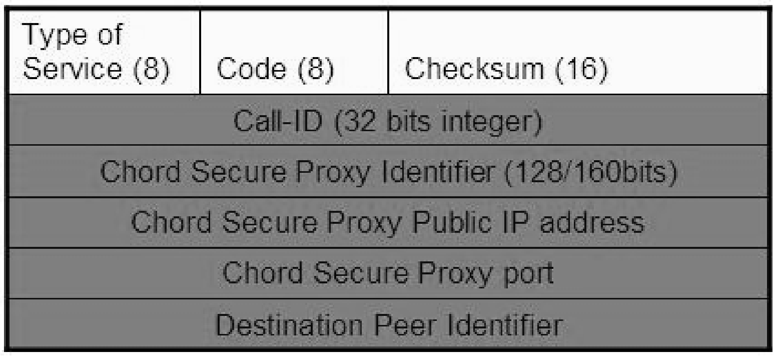
**CSP Public IP Address:** public accessible address of CSP.

**CSP port:** public accessible port of CSP.

**Destination peer Identifier**: P2PSIP ID of destination peer.

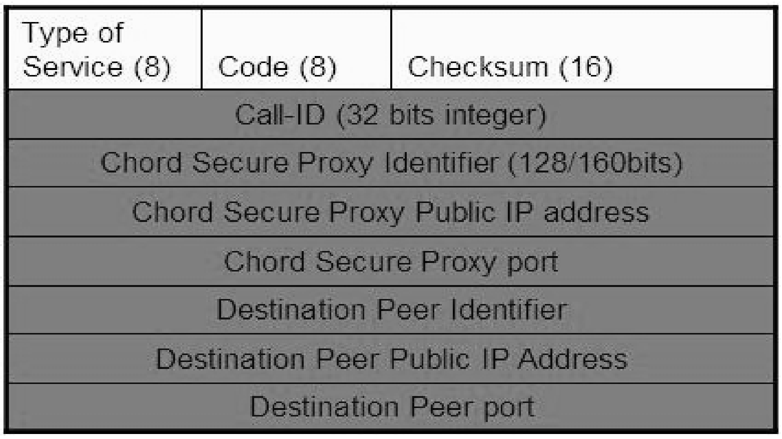
**Destination Public IP Address:** public accessible address of destination peer.

**Destination port:** public accessible port of the destination Peer.



| Type of Service (8) | Code (8) | Checksum (16) |
|---|---|---|
| Call-ID (32 bits integer) | | |
| Chord Secure Proxy Identifier (128/160bits) | | |
| Chord Secure Proxy Public IP address | | |
| Chord Secure Proxy port | | |
| Destination Peer Identifier | | |

Figure 5-3: "HelloRequest" Message Format



| Type of Service (8) | Code (8) | Checksum (16) |
|---|---|---|
| Call-ID (32 bits integer) | | |
| Chord Secure Proxy Identifier (128/160bits) | | |
| Chord Secure Proxy Public IP address | | |
| Chord Secure Proxy port | | |
| Destination Peer Identifier | | |
| Destination Peer Public IP Address | | |
| Destination Peer port | | |

Figure 5-4: "HelloResponse" Message Format

### 5.1.6 Use Scenarios

Use case 1 (see Figure 5-5) describes the P2PSIP communication establishment process between source peer A and destination peer B. Possible messages flows are:

1) Source peer sends the P2PSIP “INVITE” message to a specific CSP that is the clockwise nearest to the destination peer.

2) CSP multicasts a “HelloRequest” message to a few successors before the destination. Intermediate peers forward the “HelloRequest” to the next hop, step by step, until the destination.

3) Destination peer receives several identical “HelloRequest”, and randomly chooses one of them. Then a “HelloResponse” is returned to CSP.

4) CSP forwards P2PSIP “INVITE” message to the destination peer.

5) Destination peer returns a P2PSIP “180 Ringing” to the source peer.

6) Session negotiation and establishment.

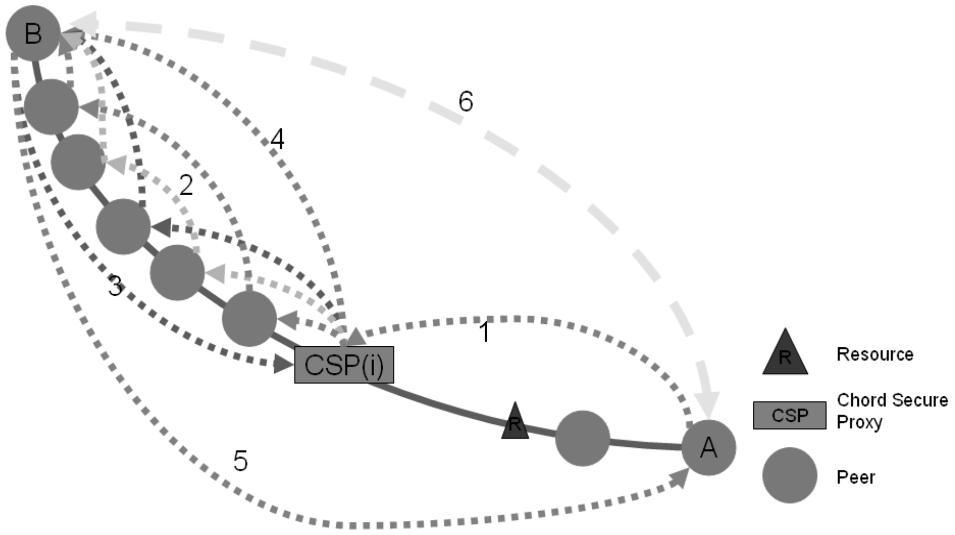


Figure 5-5: A Communication establishment use scenario

Figure 5-6 shows a use scenario with a malicious or compromised peer allocated in the destination network to interfere the message flow. Malicious/compromised peer (represent as a panda) is capable to discard, misroute, temper, and eavesdrop the data received. However, CSP-based system is tolerant to the malicious behaviour and guarantees the system availability because CSP multicast mechanism increases the surviving rate of “HelloRequest” messages. Possible interaction can be as following:

1) Source peer sends the P2PSIP “INVITE” message to a specific CSP that is responsible for the destination peer.

2) CSP multicasts “HelloRequest” message to probe the destination. A few messages might be received by the malicious intermediate peer and thus is possible to be discarded, misrouted, tempered, etc. However, the others are routed to the destination peer B.

3) Destination peer receives several identical “HelloRequest”, and randomly chooses one of them for handling. Then a “HelloResponse” is returned to the specific CSP.

4) CSP forwards P2PSIP “INVITE” message to the destination peer.

5) Destination peer returns a P2PSIP “180 Ringing” back to the source peer.

6) Session can be negotiated and established.

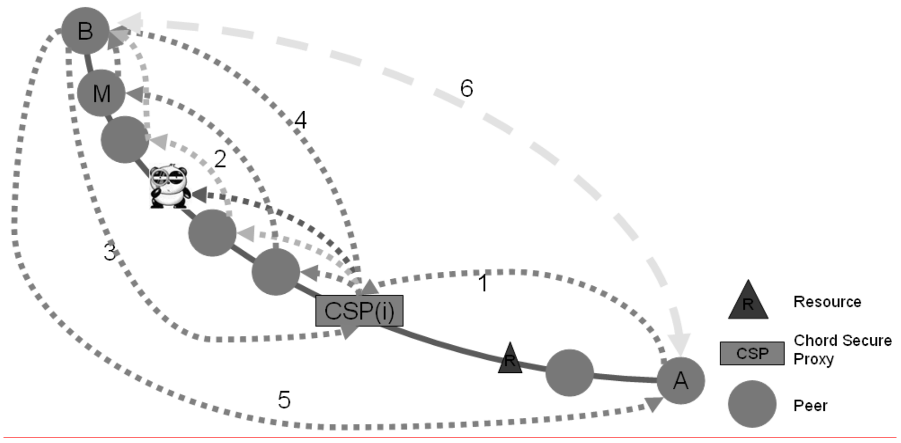


Figure 5-6: Malicious Interference Scenario

### 5.1.7 Evaluation

We evaluate the proposed system from several aspects: theoretical analysis, delay testing, and the implementation of a typical malicious use scenario.

#### Theoretical Analysis

In Chord-based system, the average number of hops is $\frac{1}{2}\log N$ [109], where $N$ is the number of peers in the overlay (as described in Chapter 2). Assuming that there are $S$ CSPs that splitting the overlay into $S$ parts evenly, the average number of hops between CSP and destination peer (for instance, CSP(i) to peer B in Figure 5-5) is:

$$\frac{1}{2}\log(N/S)$$

Adding one hop connection to source network, the average number of hops in CSP-based overlay is:

$$\frac{1}{2}\log(N/S)+1$$

The comparison in Figure 5-7 (we select $S$=16 and $S$=32 for illustration) shows that CSP-based system reduces the number of hops comparing with original Chord-based system. Besides, the more CSPs are in the overlay, the less hops are needed (in average).

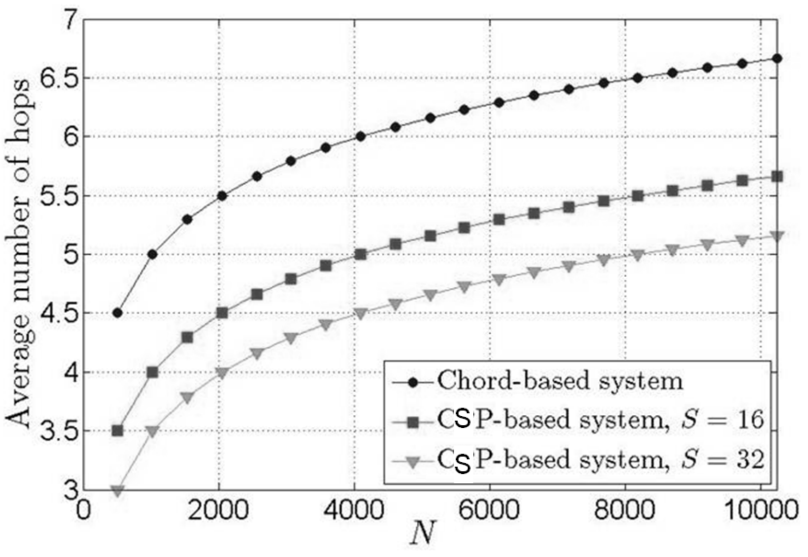


Figure 5-7: Comparison of Number of Hops

**Delay Measurement**

We also measure the delay in CSP based overlay. Peer 586 is set to be the source peer that sends out P2PSIP request to 100 random destination peers. We measure the time period between sending out request “P2PSIP INVITE” and receiving the response “180 Ringing”, and get the average delay 62ms. This is much higher than Chord-based overlay (16ms as described in Figure 3-14 in Chapter 3).

We believe that this is because destination peer in CSP-based system should wait a certain period of time for receiving multiple “HelloRequest” requests. (For instance, we set it to 30ms.) Besides, the data traffic caused by multicast might increase the burden of the system and therefore increase the delay.

**Malicious Interference Use Scenario**

We implement a typical malicious use scenario to show that CSP-based system architecture is able to protect the networks from the security breaches coming from the compromised or malicious peers.

We initiate a P2PSIP request from peer 586, searching for the destination peer 1618 (as shown in Figure 5-8). In Chord system, the message flow should go through Peer 586 -> peer 1100 -> peer 1613 -> peer 1617-> peer 1618. Then, we set the intermediate peer 1617 as a malicious/compromised intermediate peer that might discard, misroute, revise or temper the data message. Therefore, it is not possible to locate the destination peer in original Chord-based systems.

However, this is different in CSP-based system. The request would be directed to the CSP 1536. Then “HelloRequest” is distributed by multicasting and therefore causes several routes. Although one of the routes is interfered by malicious peer 1617 (the red route in Figure 5-9), two others (green and black routes) can still reach the destination peer. Finally, the destination peer could randomly pick up the black or the green route for handling.

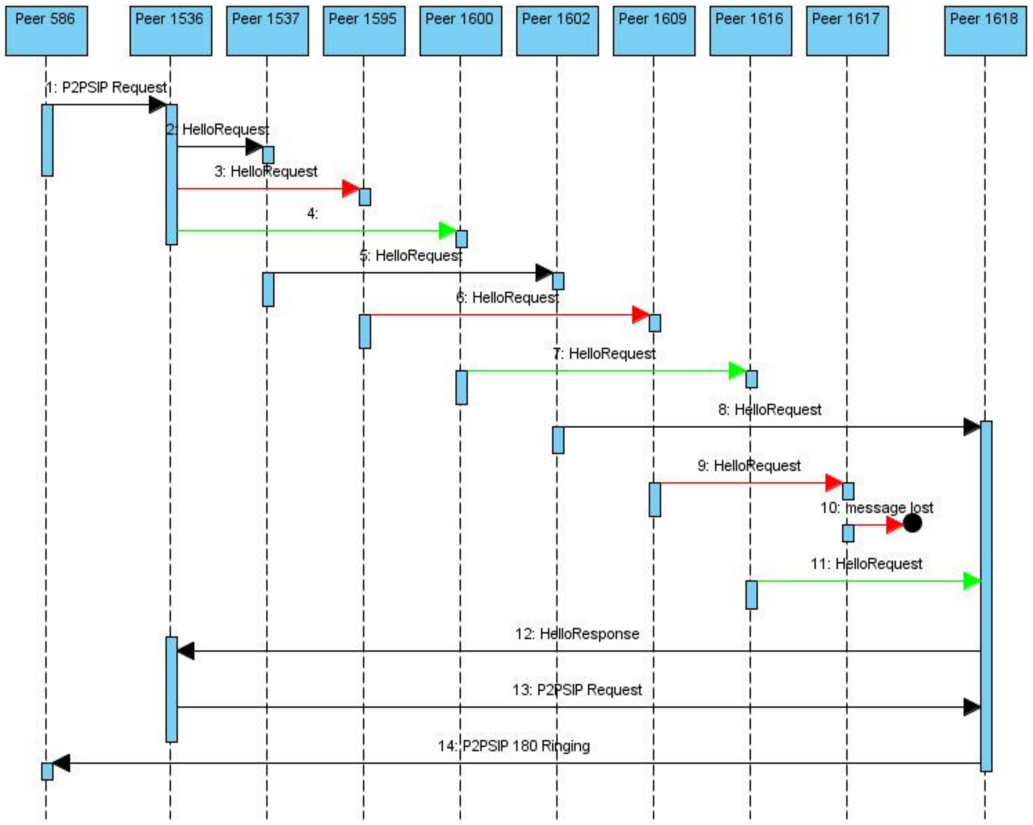


Figure 5-8: A Malicious Interference Use Scenario

### 5.1.8 Summary

In this section we have proposed a proxy-based secure architecture for P2PSIP session initiation. The system architecture resolves several issues including security, source inter-working, policy management, message transaction, destination inter-working. We use the implementation to show feasibility of this solution. Also, the evaluation shows that this system offers protection of the network from the compromised/malicious peers.

P2PSIP aims to build decentralized communication systems without (or with limited) help of centralized server. However, the proposed system model breaks (slightly) the original concept of P2PSIP. During the research, we realize that it is difficult for P2PSIP system to provide secure services without any centralized trusted entities. Therefore, our proposed system model is the compromise between theory and reality.

However, no system is completely secure. It is possible that some of P2PSIP multicast messages are received by malicious intermediate peers. This might make destination peer confusing in selecting a route. Even worse, if these multicast messages are all intercepted by a few malicious peers, the system availability would be greatly jeopardized. Therefore, a few extra mechanisms (e.g. subjective based trust [65, 82]) should be further integrated to select the most trustful route. We consider one such possible approach in the next section.

## 5.2. TRUST-BASED SECURITY APPROACH

Research efforts to improve P2PSIP trustworthiness are mostly based on PKI-based certificate approaches that have been proposed in the literature [37, 59]. In this approach, certificates are issued by a Certification Authority (CA). Certificates prove the existence and legitimacy of the specific peers.

Reputation system can be another approach to provide distributed trust. A reputation system collects, distributes, and aggregates feedback about participants' past behaviour. Several typical approaches for P2P distributed system are described in [67, 98, 106]. The idea of these approaches is similar: the reputation is represented as the discrete reputation value (e.g. 1 represents good reputation and 0 represents bad reputation). In P2PSIP services, the reputation can be earned by contributing P2P services, for instance, delivering the data traffic for the other peers, acting as the STUN server, etc. A peer with good reputation behaves as expected and thus is trustworthy, while a malicious peer that does not behave as expected will get low reputation score, and thus is less trustworthy. Usually, the impact of malicious experience is greater than positive experience. Figure 5-10 [111] shows a typical example how the reputation value is influenced by positive and negative experiences. From the time 1 to 5, the reputation value increases due to the good behaviour. However, the value decreases a lot (more than the sum of the previous 5 steps) in time 6 when the peer demonstrates bad behaviour. Steps 7-10 illustrate the positive experiences, however with lower increasing rate of trustworthiness value than during steps 1-5.

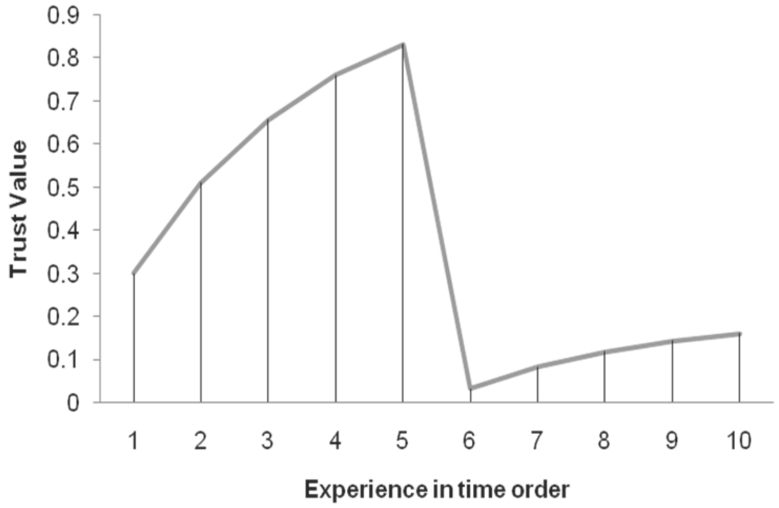


Figure 5-9: Trust based Security

In this section, we propose a novel trust-awareness based security enhancement approach. The proposed solution is based on subjective logic trust calculation, which on one hand absorbs advantages of reputation system (for instance, encourage and punishment mechanism), on the other hand offers more realistic and precise effect than traditional reputation system. The system model can be integrated with PKI-based certificate approach for enhancement of security during P2PSIP communication. In next section we describe subjective logic first proposed in [63, 64, 81] and then use it in the following sections as metric for determining trustworthiness of message transaction flow.

### 5.2.1 Subjective Logic

Papers [63, 64] define the term opinion, denoted $\omega$, which expresses an opinion about trustworthiness level. Let *t*, *d* and *u* be such that, $\{t, d, u\} \in [0,1]$ and $t + d + u = 1$. Then a triple $\omega = \{t, d, u\}$ is called an opinion where components *t*, *d* and *u* represent levels of *trust*, *distrust* and *uncertainty* respectively. For example, trustworthiness level associated with distrust could be expressed as opinion $\omega_1 = \{0, 0.95, 0.05\}$, but trustworthiness level associated with high level of trust could be expressed as opinion $\omega_2 = \{0.89, 0.00, 0.11\}$. By varying these parameters one can express several levels of trust. The level of trustworthiness can be defined based on context and properties of peers, Expressing trust by using three parameters instead of one simple trust level gives more adequate trust model of real world since when different opinions are combined these parameters are treated differently.

The subjective logic defines a set of logical operators for combining opinions such that conjunction, recommendation, consensus, etc [63, 64]. Let $\omega_p^B = \{t_p^B, d_p^B, u_p^B\}$ denote an opinion of peer *B* about logical statement *p*. In context of this section *B* is a P2PSIP peer in the overlay and statement *p* may be a statement that "data received by *B* are unchanged".

Assume that a peer *A* has an opinion $\omega_B^A = \{t_B^A, d_B^A, u_B^A\}$ about trustworthiness of recommendations given by peer *B*. Since entity *A* does not have any direct opinion $\omega_p^A$ when the message arrives via peer *B*, it will try to deduce the indirect opinion about trustworthiness of *p*, denoted $\omega_p^{AB}$, based on recommendation (opinion $\omega_p^B$) given by *B*. For this purpose the recommendation operator $\otimes$ is introduced as follows.

$$\omega_p^{AB} = \omega_B^A \otimes \omega_p^B = \{t_p^{AB}, d_p^{AB}, u_p^{AB}\}$$

where

$$t_p^{AB} = t_B^A t_p^B, \ d_p^{AB} = t_B^A d_p^B \quad \text{and}$$

$$u_p^{AB} = d_B^A + u_B^A + t_B^A u_p^B.$$

### 5.2.2 Subjective Logic based Architecture

In this section we propose a subject logic based trust architecture. The proposed architecture involves three main parts: P2PSIP Peer, Resource, and Secure Opinion Server (SOS), as shown in Figure 5-11. P2PSIP peer is connected to the Internet.

Resource is the data value stored in a specific peer. Secure Opinion Server (SOS) is the trust management server that stores and computes the dynamic opinion for each P2PSIP peer.

To locate a peer/resource, the source peer first multicasts P2PSIP request to a certain number of successors which are in the anti-clockwise direction of the destination peer. Intermediate peers forward the received request step by step until the destination peer is reached. Finally, the destination peer might receive multiple request messages, and it turns to the SOS server for selecting the most trustful one. After that, the session between source peer and destination peer could be securely established.

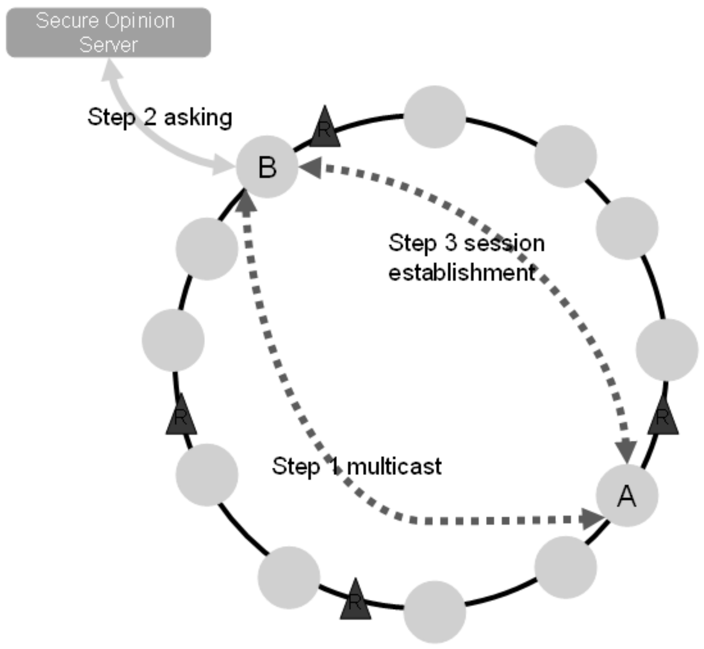


Figure 5-10: Subjective Logic Trust Model

Secure Opinion Server (SOS) is the key inter-working unit, acting as a decision maker for the destination peer. Note that SOS can be either collocated inside the Enrolment and Authentication (E&A) server or as a separate unit. It contains three components (see Figure 5-12):

- Connection handling receives the RPC (Remote Procedure Call) request message (that contains a list of routing options), and generates the corresponding response.

- Opinion computation is responsible to calculate the opinion of each message flow based on the subjective logic rules. Besides, it updates the opinion for each peer periodically (according to the rules defined in Section 5.2.4).

- Opinion DB is a component that stores current opinion about trustworthiness of each P2PSIP peer.

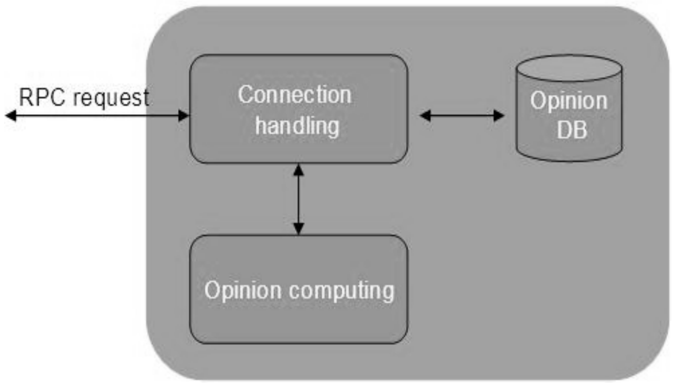


Figure 5-11: Secure Opinion Server (SOS) Internals

## 5.2.3 Opinion Calculation

The following will demonstrate opinion calculation based on subjective logic rules. Suppose that a request goes from the source peer *A*, through intermediate peers $B_1$, $B_2$, $B_{n-1}$, and to the destination peer $B_n$. Let *p* denote as "data received by $B_n$ is unchanged". By applying the rules of subjective logic described in Section 5.2.1, the trustworthiness of this data delivered through this route can be calculated as following:

$$\omega_p^{AB_1B_2\ldots B_{n-1}B_n} = \omega_{B_1}^{A} \otimes \omega_{B_2}^{B_1} \otimes \omega_{B_3}^{B_2} \otimes \ldots\ldots \otimes \omega_{B_n}^{B_{n-1}} \otimes \omega_p^{B_n}$$

where

$$t_p^{AB_1B_2\ldots B_{n-1}B_n} = t_{B_1}^{A} t_{B_2}^{B_1} t_{B_3}^{B_2} \ldots . t_{B_n}^{B_{n-1}} t_p^{B_n} = t_{B_1}^{S} t_{B_2}^{S} t_{B_3}^{S} \ldots . t_{B_n}^{S} t_{B_n}^{S}$$

$$d_p^{AB_1B_2\ldots B_{n-1}B_n} = t_{B_1}^{A} t_{B_2}^{B_1} t_{B_3}^{B_2} \ldots . t_{B_n}^{B_{n-1}} d_p^{B_n} = t_{B_1}^{S} t_{B_2}^{S} t_{B_3}^{S} \ldots . t_{B_n}^{S} d_{B_n}^{S}$$

$$u_p^{AB_1B_2\ldots B_{n-1}B_n} = 1 - t_{B_1}^{A} t_{B_2}^{B_1} t_{B_3}^{B_2} \ldots . t_{B_n}^{B_{n-1}} t_p^{B_n} (1 - u_p^{B_n})$$
$$= 1 - t_{B_1}^{S} t_{B_2}^{S} t_{B_3}^{S} \ldots . t_{B_n}^{S} t_{B_n}^{S} (1 - u_{B_n}^{S})$$

Based on the opinion result, we introduce another parameter *v* that represents the final score of a specific message transmission. The higher value of *v*, the higher trustworthiness of the message flow. We define *v* in the following:

$$v = t + (1/2)u - 2d$$

## 5.2.4 Opinion Maintenance

SOS is the secure server that stores the opinion for each peer in the overlay. We define the Initial Opinion (IOP) as the first opinion when peer joins the overlay for the first time. The Secure Opinion Server (SOS) assigns the IOP based on the system capabilities of the peer, such as available processing power (*p*), memory (*m*), bandwidth (*b*), etc. For instance, the rule could be as following:

- Initiative distrust value is: *d=0.*
- If the available processing is larger than *200MHZ*, *p=(1/6)*; otherwise, *p=(1/6)*(processing/200M).*
- If bandwidth is larger than 300k, *b=(1/6)*; otherwise, *b=(1/6)*(bandwidth/300k).*
- If memory is larger than *100M*, *m* = (1/6); otherwise, *m=(1/6)*(free memory/100M).*
- Initial trust value is: *t=p+b+m, if p+b+m<0.5*, otherwise, *t=0.5.*
- Initial uncertainty value is: *u=1-t.*

The opinion is dynamically updated according to the behaviour of each peer. It may increase in some rate according to the contribution of the overlay (e.g. act as the intermediate peer to relay the traffic, etc) or degrade when there is no contribution (for example, nothing happened during some time period). The encouragement rules could be as following:

- When the peer acts as an intermediate peer that relays the data traffic, the trust *t* increases and the uncertainty *u* decreases if the peer behaves as expected:

$$t = t_{prev} + (1/200)d_{prev} + (1/200)u_{prev}$$

$$d = d_{prev} * (199/200) \qquad \text{and}$$

$$u = u_{prev} * (199/200)$$

Periodically, the SOS server inspects the opinion DB. If a peer does not contributes in some time period (e.g. 10 minutes, etc), it may be suspected to be malicious or faulty peer. Therefore, the distrust *d* increases while the trust *t* and the uncertainty *u* decrease. We define the rule as following:

$$d = d_{prev} + (1/50)t_{prev} + (1/50)u_{prev}$$

$$t = t_{prev} * (49/50) \qquad \text{and}$$

$$u = u_{prev} * (49/50)$$

Usually, the impact of negative experience is greater than the impact of positive experience. In our system model, we define that degrading rate based on negative experience is four times faster than increasing rate based on positive behaviour.

### 5.2.5 A Typical Use Scenario

We implement a typical use scenario to show that proposed approach provides better availability and security than traditional Chord-based system. Our implementation is based on previous implementation of Chord-based P2PSIP system with 512 peers in the space size 2048 (introduced in Chapter 3.8). We modify several functions (e.g. one-hop multicast, semi-recursive routing, etc) to realize the proposed system. We also assume the system contains most of the normal P2PSIP peers (for example, 99%) and a few malicious / faulty peers.

We also implement a Secure Opinion Server by using Java as the programming language, Apache Derby as the opinion database, and Apache tomcat as the background HTTP container.

In this case, we initiate a P2PSIP request from peer 668, searching for the destination peer 1616. In the original Chord system, the message flow goes through peer 668 -> peer 939 -> peer 1030 -> peer 1110-> peer 1116. Then, we set the intermediate peer 1110 as a malicious/fault intermediate peer that might discard, misroute or temper the data message. The testing of the original Chord-based system shows inability to locate the destination peer. However, this is different in the current approach. Because of the one-hop multicast function in the source peer, the destination peer might be able to receive multiple P2PSIP requests, as represented in Figure 5-14. Although the red one is misrouted/blocked by the malicious intermediate peer 1110, the other two routes (Black one and Green one) can still reach the destination.

Then, we assume in a certain period, the opinions of related peers are:

Table 5-1 Peer Opinion Table

| Peer ID | Trust | Distrust | Uncertainty |
|---|---|---|---|
| 668 | 0.9 | 0.05 | 0.05 |
| 784 | 0.8 | 0.1 | 0.1 |
| 796 | 0.82 | 0.08 | 0.08 |
| 1040 | 0.75 | 0.15 | 0.1 |
| 1052 | 0.92 | 0.04 | 0.04 |
| 1104 | 0.85 | 0.1 | 0.05 |
| 1112 | 0.9 | 0.05 | 0.05 |
| 1116 | 0.95 | 0.04 | 0.01 |

We simulate this by manually modifying the opinion database. According to subjective logic rules specified before, the opinions about trustworthiness of two routes are:

$$\omega_p^{\mathrm{Re}d} = \{0.392, 0.016, 0.592\} \text{ with } v=0.656$$

$$\omega_p^{Black} = \{0.645, 0.027, 0.328\} \text{ with } v=0.755$$

After the opinion calculation, SOS returns the most trustful route (the black one) to the destination peer 1116 that allows choosing the most trustful route for session establishment.

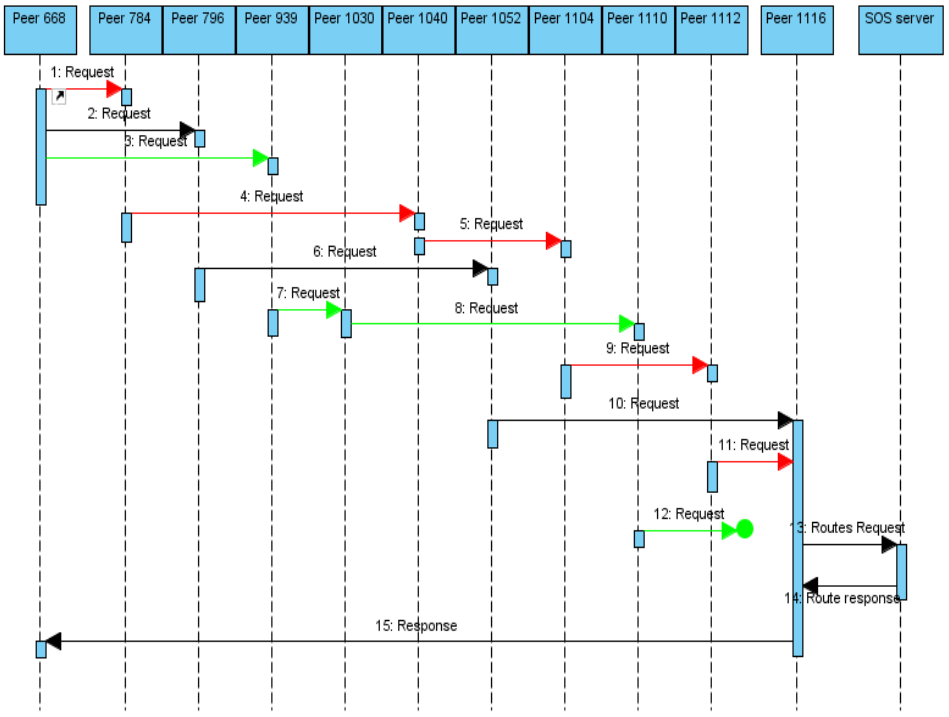


Figure 5-12: A Typical Use Scenario

### 5.2.6 Summary

Subjective logic based trust model provides secure services via selecting the most trustful message routes. The system resolves several issues including opinion calculation, opinion maintenance, message routing, and NAT traversal. Our approach improves the trustworthiness in the P2PSIP session establishment and protects the system from security breaches caused by misbehaviour of the malicious or faulty peers.

However, some issues are still remaining. For example, source peer in this solution multicasts session layer “P2PSIP INVITE” messages into the overlay. This might increase load within overlay network. Besides, the malicious peer who receives multicasted messages is capable to collect sensitive privacy information. That creates new security concerns. Additionally, it might happen, in the worst case, multicast messages from source peer are all intercepted by a few malicious peers. In this case, the request fails.

## 5.3. COMBINING EFFICIENCY AND SECURITY

Two security solutions proposed above have their limitations. Appropriate combination of these two solutions might provide solution that is both efficient and sufficiently secure. In this section, we study use case that combines both centralized proxy model and subjective logic trust model. Besides, we also combine two efficiency improvement approaches: cache mechanism and hierarchical layer division. According to previous study (in Chapter 3), cache mechanism is efficient in reducing chord lookup delay. Hierarchical layer division, according to theoretically analysis in [70], is capable to increase the overall capability of the overlay and reduce the system delay. On the other hand, computationally strong devices (with strong CPU power, big memory, and stable connection) have generally better protection (e.g. anti-virus software, firewall, etc) against security breaches than weak devices (for example, mobile phones in WiFi/3G connections). Therefore, the division of hierarchical suboverlay also provides, to some degree, security improvement for the top suboverlays.

### 5.3.1 System Architecture

We suggest divide the overlay into three sub-overlays according to peer capabilities, as shown in Figure 5-15. The first sub-overlay consists of stable peers that have public IP addresses, more powerful CPU, and stable connection. Such typical device can be a web server. Peers in the second sub-overlay are those who have enough stability and processing power, e.g. normal PC with Internet connection. Peers in this layer do not own public IP address, and might relay on STUN/TURN/ICE for NAT traversal. The lowest sub-overlay is those with unstable connection (e.g. mobile phones, PDA, laptops with wireless connection). Note that each sub-overlay contains a few CSPs for handling security services in intra-layer, and at least one CSPG (Chord Secure Proxy Gateway) for handling secure inter-layer communication. Both of CSP and CSPG are stable P2PSIP peers.

It is expected that many legacy P2PSIP peers are unstable peers (e.g. a large amount of mobile phones, PDA, laptops, etc) with wireless connections. Therefore, the division of three sub-overlay guarantees peer/resource lookup efficiency, and security protection in the top two layers.

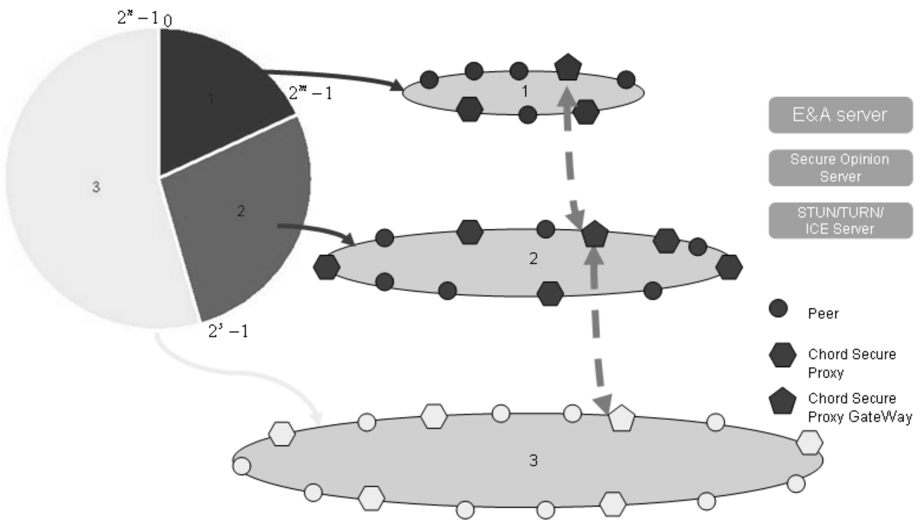


Figure 5-13: Three Layer Architecture

## 5.3.2 "Ping" Multicast

Subject logic trust calculation requires the record of message route. However, this is not available in centralized proxy based solution (using "Hello" message, specified in Chapter 5.1). Therefore, we suggest a session level "Ping" multicast mechanism, which contains "PingRequest" and "PingResponse". "PingRequest" message consists of a "Via" header for recording the profile (e.g. peer ID, IP, port, etc) of each intermediate peer. An example is represented below:

```
P2PSIP PingRequest
Via: 586 158.36.228.48:9000; 612 128.39.189.61:8080
Call-ID : 9849303
CSP-ID: 512
CSP-IP: 158.36.228.48
CSP-Port: 9512
Dest-ID:586
```

When destination peer receives multiple "PingRequest" messages, it selects the most trusted route for handling (based on subjective logic trust calculation) and replies with a "PingResponse". An example of Ping Response is:

```
P2PSIP PingResponse
Call-ID: 9849303
CSP-ID: 512
CSP-IP: 158.36.228.48
CSP-Port: 9512
Dest-ID: 586
Dest-IP: 69.0.128.30
Dest-Port: 9001
```

### 5.3.3 Use Cases

Figure 5-16 illustrates inter-layer P2PSIP session initiation process between source peer A and destination peer B. Possible messages flows are:

1) Source peer sends P2PSIP “INVITE” message to the CSPG in its sub-overlay.

2) CSPG forwards “INVITE” to another CSPG in destination sub-overlay.

3) The “INVITE” is forwarded to the CSP that is clockwise nearest to the destination peer.

4) CSP multicasts a “PingRequest” to a few successors. Intermediate peers forward “PingRequest” step by step until the destination.

5) Destination peer receives several identical “PingRequest”. It asks SOS server via sending all possible routes. After trust calculation, SOS replies with a best route.

6) Destination peer returns a “PingResponse” to CSP.

7) CSP forwards original P2PSIP “INVITE” message to destination peer.

8) Destination peer returns a P2PSIP “180 Ringing” to source peer.

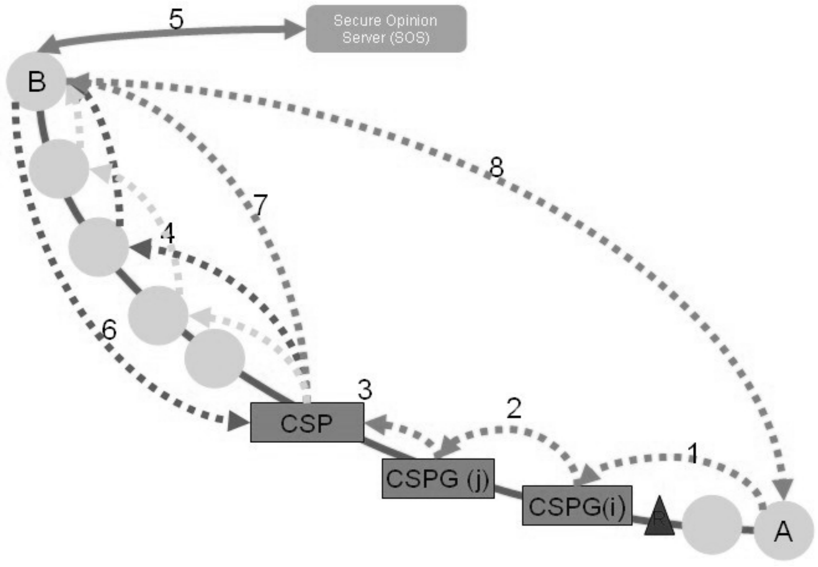


Figure 5-14: Inter-Layer Session Initiation

### 5.3.4 Efficiency Study

We first analyze the lookup algorithm without the consideration of cache mechanism. We assume that the number of peers and CSPs in the overlay is $N$ and $S$ respectively, where $N_1, N_2, N_3$ are the number of peers in each sub-overlay from top to bottom and $S_1$, $S_2$, $S_3$ are number of CSPs in each sub-overlay from top to bottom. Besides, we assume that source peer communicates with the other peers in each sub-overlay (layer 1, layer 2, and layer 3) with a probability of $p_1$, $p_2$, $p_3$. Also, we assume that peers and CSPs are evenly distributed in the overlay space.

Based on Chord routing protocol [109], the average num-of-hop of "PingRequest" multicast is $\frac{1}{2}\log(N_i / S_i)$, where $i$ denotes a corresponding sub-overlay. Therefore, the number of hops of intra-suboverlay is $\frac{1}{2}\log(N_i / S_i)+1$ due to the addition of one CSP; the complexity of inter-suboverlay is $\frac{1}{2}\log(N_i / S_i)+3$ due to addition of two CSPGs and one CSP (see Steps 1-3 in Figure 5-16).

According to the mean rule [103], the average number of hops is:

$$\sum_{i=1}^{3}(p_i*(1+\frac{1}{2}\log(N_i / S_i))+(1-p_i)*(3+\frac{1}{2}\log(N_i / S_i)))$$
$$=3-\frac{2(p_1N_1+p_2N_2+p_3N_3)}{N}+\frac{1}{2}\log_2(\frac{N}{S})$$

After that, we assume that $p_1 = p_2 = p_3 = 0.8$ , based on the concept that most communication sessions are geographically related to each other (according to Section 3.3). Therefore, the average num-of-hops is:

$$1.4 + \frac{1}{2}\log_2(\frac{N}{S})$$

Figure 5-17 shows the improved result (we set *S*=16 and *S*=32 separately) comparing with conventional Chord-based system. We come to the conclusion that our proposed lookup mechanism is more efficient than a conventional Chord lookup approach. Besides, with increasing *S*, the better lookup efficiency will be also provided.

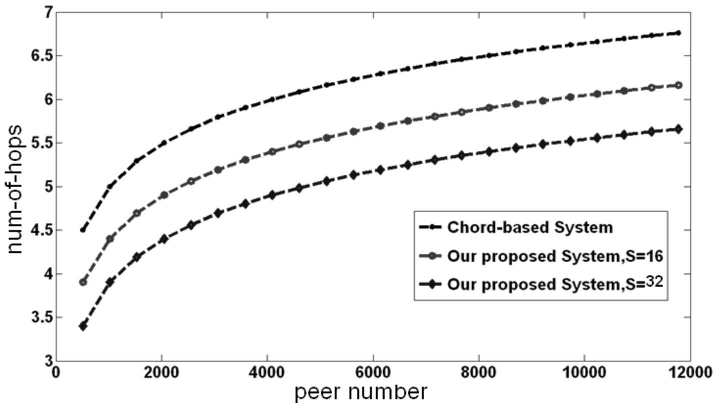


Figure 5-15: Num-of-Hops Comparison

In order to evaluate the impact of cache mechanism, we measure the system delay. We choose one peer (we use peer 586 in this example) as the source peer, and randomly select 100 peers (which are divided into 10 groups, each of which contains 10 peers) in each suboverlay as the destination peer. We initiate P2PSIP request from source peer and measure the latency between request and response. After that, we calculate the average delay in each group (shown in Figure 5-18).

In the beginning, the delays in three sublayers are more or less similar (between 250-300ms). However, the latencies in layer 1 (green one) and layer 2 (red one) are greatly reduced with the increasing of the number of groups. We believe this is the contribution of cache mechanism.

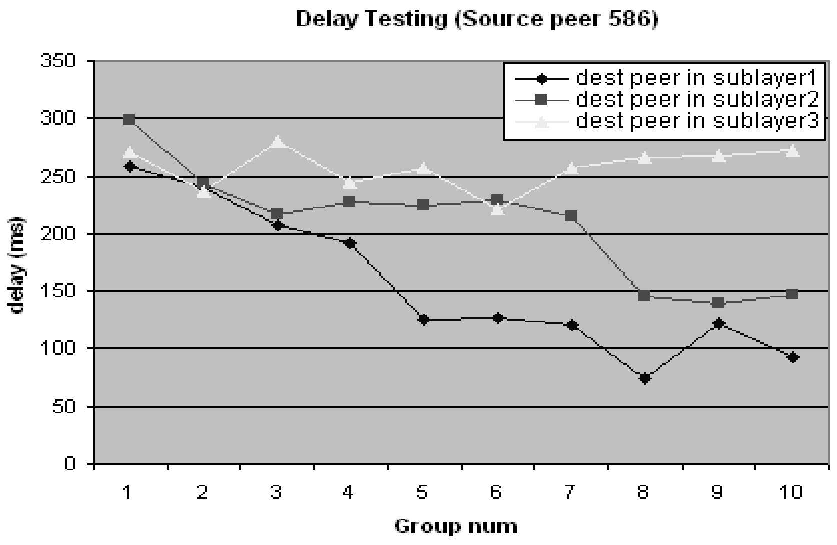


Figure 5-16: Delays Testing

## 5.3.5 Security Assessment

Table 5-1 gives the security comparison among three solutions introduced in this chapter.

**Table 5-2 Three Solutions Security Comparison**

| | CSP-based solution | Subjective logic based trust enhancement | Combination solution + hierarchical division |
|---|---|---|---|
| Availability | Low, ”PingRequest” is sent one way out by Chord routing algorithm. Malicious peers in this route could drop, modify, or misdirect messages. | High, “P2PSIP INVITE” multicast creates several route options. | High, “PingRequest” multicast creates several route options. |
| Confidentiality | Certificate based encryption and decryption. | Certificate based encryption and decryption. | Certificate based encryption and decryption. |
| Data Integrity | Certificate based digital signature. | Certificate based digital signature. | Certificate based digital signature. |

| | | | |
|---|---|---|---|
| Authentication | CSP is able to provide authentication. | No trusted entities in overlay, no authentication. | CSP and CSPG authenticate request and response. |
| Privacy | Strong guarantee.<br>CSP protects privacy of source peer. | Weak because request is multicasted. Any intermediate malicious peer who receives the request can understand sensitive privacy. | Strong guarantee.<br>CSP and CSPG protect privacy of source peer. |
| Route trustworthiness | Not trusted. Because "PingRequest"can be modified by malicious intermediate peer in the middle. | Good. SOS server selects best route according to opinion calculation. | Good. SOS server selects best route according to opinion calculation. |
| Security vulnerabilities | Low availability. | Privacy problem. | Peers in lowest sublayer still face security problems. Mostly because of their own security vulnerability (e.g. no protection, dangerous under the virus, etc). |

Security is mainly based on previous proposals, including CSP based security, subjective logic based trust enhancement, PKI certificate based security, etc. Besides, three-layer hierarchical division also guarantees, to some degree, security in the first two sublayers through classifying unstable peers (usually also security vulnerable peers) belonging to the lowest sublayer. We believe the proposed solution combines the advantages of several solutions and provides improved overall security.

However, from one side hierarchical division improves overall system security; from other side it moves security issues to the lowest sublayer. Therefore, the question how to enhance security in that vulnerable sublayer of the overlay should be considered in the future.

# Chapter 6. Secure Interconnecting with P2PSIP and IMS

P2PSIP should provide means for interconnecting different networks. This chapter introduces an approach for interworking between P2PSIP system and future IP Multimedia Subsystem (IMS) based system.

## 6.1. INTRODUCTION

Currently, researchers are beginning to study the possibility of interconnecting between P2PSIP and IMS networks. One typical proposal is described in [56], which implements a Gateway Application Server (AS) that is a peer on P2PSIP side and an Application Server on IMS side (Figure 6-1). Through the bridge function of Gateway AS, the users in different networks are capable to communicate with each other. The system model looks feasible from networking point of view.

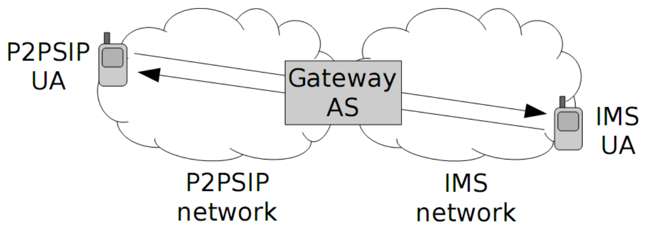


Figure 6-1: Interconnection Model

However, the proposed interconnection model faces serious security problems. Firstly, security of the messages traversing inside P2PSIP overlay is not guaranteed due to the nature of P2P (distrust among participating peers, etc). Let us consider a typical malicious model (represented in Figure 4-2), which also specifies interaction between

Gateway AS and P2PSIP UA. Peer B (the panda) that acts as a malicious intermediate peer in P2PSIP overlay is capable to misroute, discard, temper, and replay the received P2PSIP messages.

Secondly, Gateway AS is public to all malicious participating peers in the overlay, which makes a few malicious behaviours possible. For example, a malicious peer is capable to initiate SPAM attack to IMS network through sending a multitude of P2PSIP requests to Gateway AS. This may cause unnecessary trouble (e.g. ring call, instance message, etc) in IMS users. Besides, the malicious peer (who receives P2PSIP message) could spy and record the profiles of previous intermediate peers (e.g. peer ID, public IP, Port, etc.) through parsing incoming messages. This sensitive information could be used to initiate DoS and SPAM attacks on a peer or the PIGW.

Therefore, in order to provide full interconnecting solutions, security issues should be taken into consideration, especially security issues within P2PSIP network. Before we propose possible solutions, we assume two requirements for interworking between P2PSIP and IMS that must be satisfied:

- Networking availability: At least one trusted gateway for relaying messages between P2PSIP and IMS domains must be available.
- Security Guarantee: The secure message routing in P2PSIP domain should be guaranteed. Besides, the gateway should be resilient on a series of attacks, e.g. DoS attack, SPAM, etc.

In this chapter, we investigate on P2PSIP and IMS systems and propose P2PSIP-IMS GateWay (PIGW) as a secure interworking gateway between P2PSIP and IMS domains. Security is achieved by combination of Chord Secure Proxy (CSP), PKI-based certificate and subjective logic based trust approaches (as described in Chapter 5).

## 6.2. INTERWORKING ARCHITECTURE

Figure 6-2 shows the proposed system architecture, which contains following five elements:

- P2PSIP-IMS Interworking Gateway (PIGW) is the key interworking unit for translation of messages between P2PSIP and IMS networks.

- P2PSIP peer, which can be a PC, laptop, PDA, mobile phones etc., is connected into the Internet. Each P2PSIP peer has a corresponding CSP as its master node.

- Chord Secure Proxy (CSP) is the secure proxy that relays the messages among PIGW and P2PSIP peers. The main task of CSP is to protect sensitive information (e.g. peer ID, public IP, port, etc) from understanding to most of P2PSIP peers but itself.

- Enrolment & Authentication (E&A) Server handles enrolment and authentication task when P2PSIP peers join P2PSIP overlay.

- Secure Opinion Server (SOS) is the security enhancement server that handles dynamic opinion computing and storage task for each P2PSIP peer.

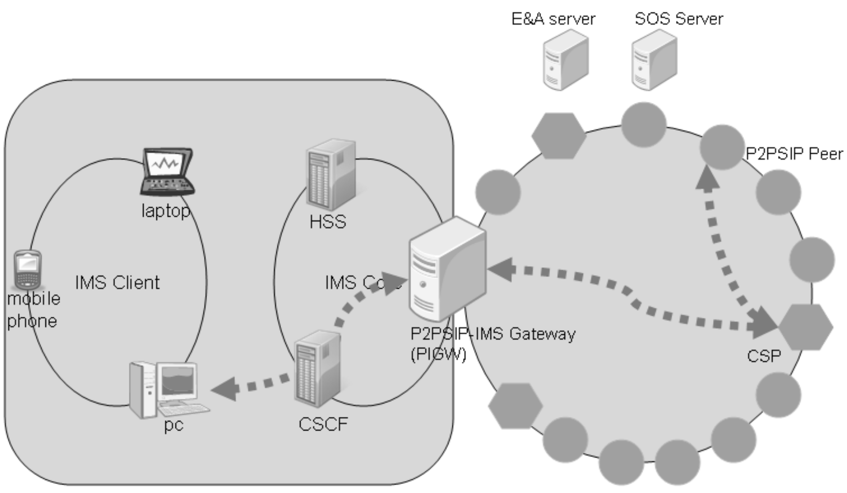


Figure 6-2: Secure System Architecture

HSS (Home Subscriber Server) is the IMS core element which provides identity authentication and management for IMS clients, including PIGW; CSCF is the IMS core element that relays messages between PIGW and IMS clients. CSPs and PIGW are pre-deployed backbone nodes in P2PSIP network. They are assumed to be trusted in P2PSIP network. In the following sections, we will specify technical approaches including networking and security.

## 6.3. P2PSIP-IMS GATEWAY (PIGW)

P2PSIP-IMS Gateway is the key inter-working unit, acting as bridge between P2PSIP and IMS networks. PIGW acts as a normal P2PSIP peer on P2PSIP side and as an IMS application server on IMS side. There are five components inside PIGW (see Figure 6-3):

- P2PSIP Peer. This subcomponent acts as a normal peer that receives/sends P2PSIP message from/to P2PSIP network.
- Translation Logic component. This component handles translation between P2PSIP and IMS messages.
- Forwarding Logic component. This component decides where and how to forward P2PSIP messages. It defines message routing strategy. For example, it defines the rule: P2PSIP message is forwarded to a specific CSP that is anti-clockwise nearest to the destination peer. Inside this subcomponent, there is a database recording all the connections to CSPs (e.g. CSP ID, public IP, port, etc) in P2PSIP overlay.
- IMS UA. IMS UA handles IMS client functionality that sends/receives IMS messages to/from IMS core. It contains an UICC smart card for IMS authentication.
- IMS Application Server. This part receives IMS request from IMS client and sends the corresponding response to IMS core.

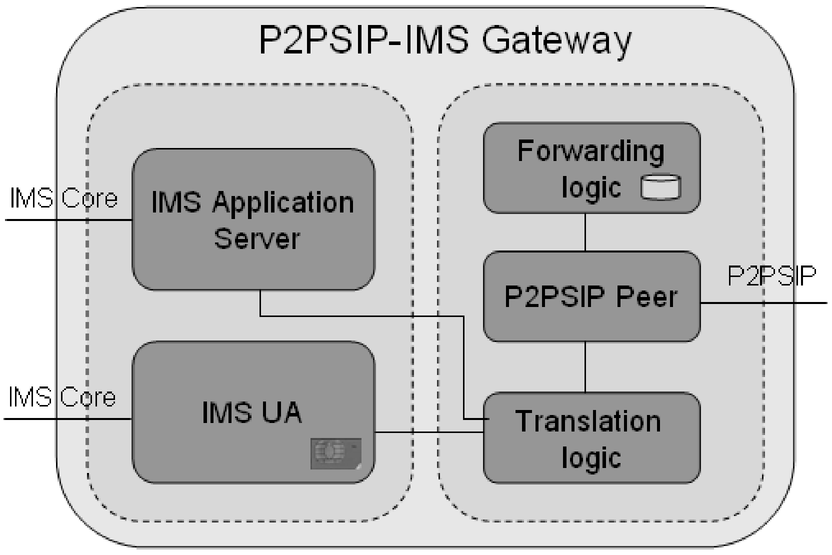


Figure 6-3: P2PSIP-IMS Gateway Internal

When IMS Application Server component receives IMS request, the translation logic component parses the messages, retrieve destination peer identity and generate corresponding P2PSIP message. After that, P2PSIP UA sends out P2PSIP request to a specific CSP according to the direction of the Forwarding logic component.

When receiving P2PSIP request, translation logic component parses P2PSIP messages, retrieves IMS related information (e.g. destination IMS ID, etc), generates IMS messages, and forwards to IMS UA component, which sends out IMS request.

With respect to security, we suggest that PIGW is only capable to communicate with CSPs. Since CSP is assumed to be trusted, sensitive privacy (e.g. PIGW peer ID, public IP, port, etc) is revealed to most of P2PSIP peers. Therefore, it is difficult for malicious peer to initiate a few attacks, such as DoS attack and SPAM to PIGW.

## 6.4. SECURITY AND PRIVACY

IMS core is assumed to be secure due to its own mature security framework (e.g. IPsec based security, HSS based authentication and authorization, etc).

PIGW is assumed to be a trusted entity. Therefore, it is capable to protect privacy of IMS client and P2PSIP peer from leaking confidential data. For protecting data confidentiality and integrity, all connections must be encrypted.

P2PSIP overlay security is enhanced as described in Chapters 5, including CSP based security, subjective logic based trust enhancement, PKI certificate based security, E&A server, etc.

## 6.5. Error Handling

The initiated request might not be able to reach the target due to a few reasons. For example, IMS client or P2PSIP peer might loss the connection with network due to the limitation of device capability (e.g. no power, system deadlock, etc) or network problem (e.g. no signal, etc). Therefore, it is necessary to notify the source when the target is unreachable. We propose that PIGW handles notification task by sending "P2PSIP MESSAGE" and "SIP MESSAGE". One typical example will be shown in the following section.

## 6.6. Use Case Scenarios

In the following subsections we demonstrate the using of the proposed architecture for text based instant messing services, with three use cases. We define "P2PSIP MESSAGE" and "SIP MESSAGE" as the requests, "P2PSIP 200 OK" and "SIP 200 OK" as corresponding responses. Note that the proposed system architecture can be extendable with other advanced services (e.g. presence services, VoIP, etc).

Use Case 1 (see Figure 6-4) describes how IMS client sends message to a P2PSIP peer. Possible message exchange among IMS client, PIGW, CSP, intermediate peers, and destination peer is shown in the following description:

1. IMS client sends "SIP MESSAGE" message to PIGW (for example, 260 as P2PSIP ID and pigw@ericsson.com as IMS ID).

2. PIGW returns "SIP 200 OK" to IMS client.

3. PIGW sends "P2PSIP MESSAGE" message to the specific CSP that is responsible for destination peer.

4. CSP multicasts "PingRequest", which is then forwarded by intermediate peers to the destination.

5. Destination peer receives several “PingRequest” from different routes. It asks SOS server to select one of them.

6. SOS server returns the most trustful route.

7. Destination peer returns a “PingResponse” to corresponding CSP.

8. CSP forwards original “P2PSIP MESSAGE” to destination peer.

9. Destination peer returns a “P2PSIP 200 OK” the corresponding CSP.

10. CSP forwards original “P2PSIP 200 OK” back to PIGW.

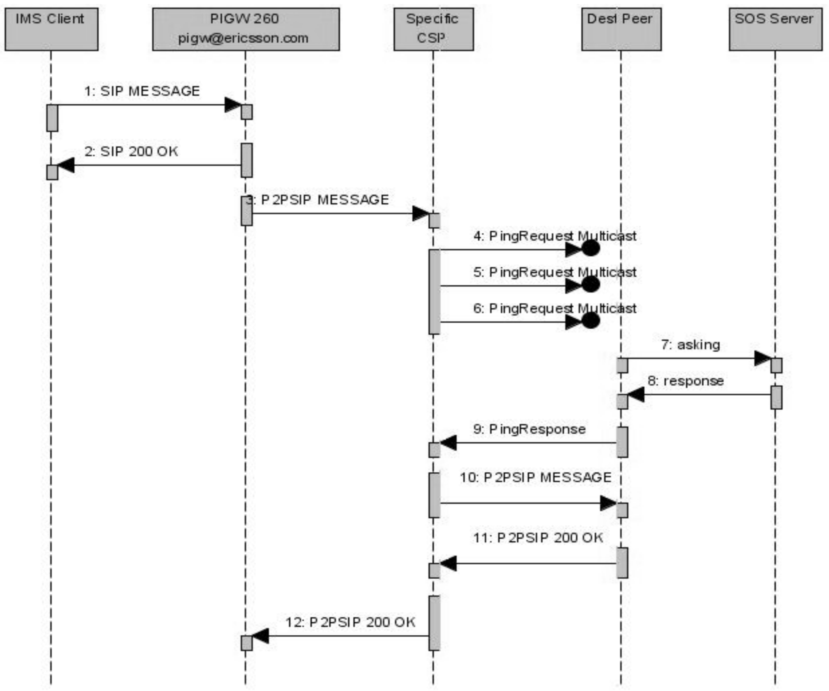


Figure 6-4: IMS Client sends message to P2PSIP Peer

Figure 6-5 describes how P2PSIP peer sends message to IMS client. Possible message exchange among P2PSIP peer, CSP, PIGW, and IMS client is shown in the following description:

1. P2PSIP peer sends “P2PSIP MESSAGE” to a responsible CSP.

2. CSP forwards “P2PSIP MESSAGE” to PIGW (for example, 260 as P2PSIP ID and pigw@ericsson.com as IMS ID).

3. PIGW returns “SIP 200 OK” to the corresponding CSP.

4. Corresponding CSP returns “P2PSIP 200 OK” back to P2PSIP peer.

5. PIGW sends “SIP MESSAGE” to IMS client.

6. IMS client returns “SIP 200 OK” to PIGW.

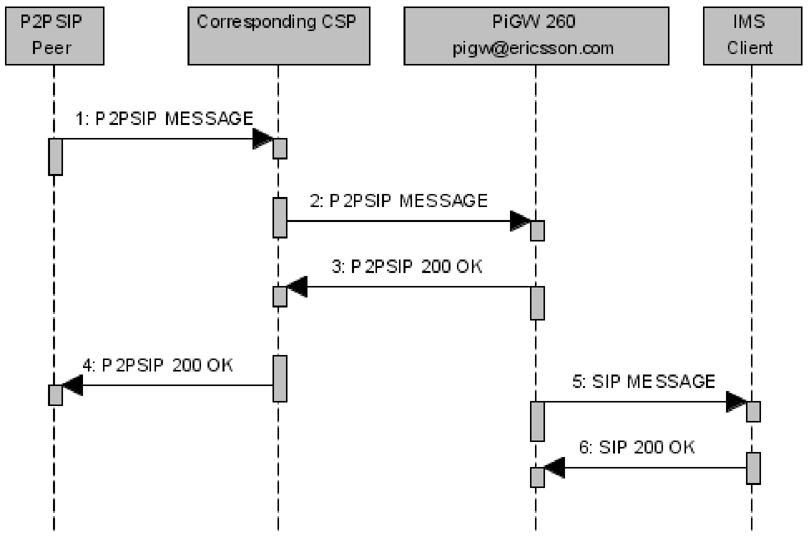


Figure 6-5: P2PSIP Peer sends message to IMS Client

Figure 6-6 shows the error handling use scenario when P2PSIP peer is unreachable. Possible message exchange among source peer, CSP, intermediate peers and destination peer is shown in the following description:

1. P2PSIP peer sends “P2PSIP MESSAGE” to its corresponding CSP.

2. Corresponding CSP forwards “P2PSIP MESSAGE” to PIGW (for example, 260 as P2PSIP ID and pigw@ericsson.com as IMS ID).

3. PIGW sends “P2PSIP MESSAGE” to P2PSIP network.

4. Message retransmission after a certain time (Time to Live (TTL) is defined).

5. PIGW replies to IMS client with “UNREACHABLE PEER” message.

6. IMS client returns “SIP 200 OK” to PIGW.

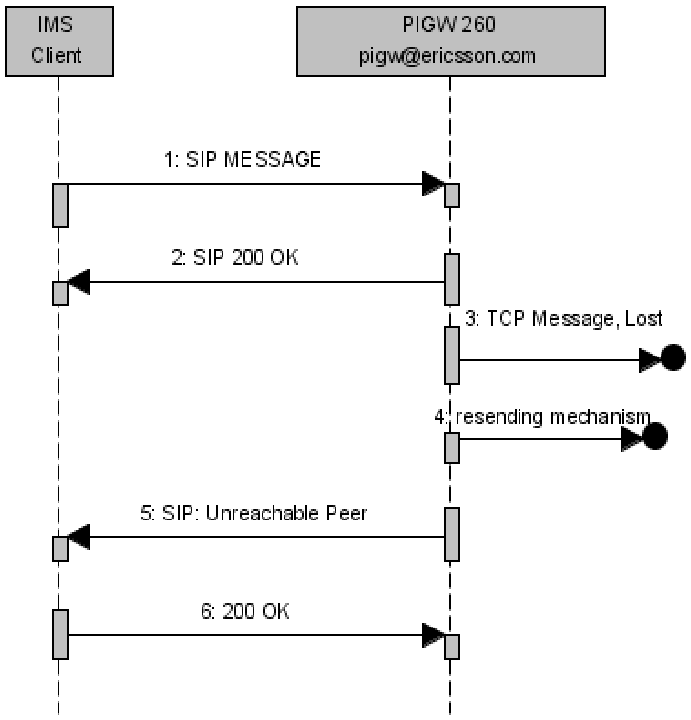


Figure 6-6: Handling Unreachable P2PSIP Peer

## 6.7. SIMULATION

We simulate a P2PSIP overlay of 512 P2PSIP peers, with 496 P2PSIP normal peers, 15 CSPs and a PIGW peer. After that, we import IMS application server function to PIGW, which then acts as an IMS application server (with ims id: greetings@ericsson.com) and a P2PSIP peer (with id: 260). Apache Derby is selected as the embedded database implementation for P2PSIP peers, CSPs, and PIGW.

Ericsson SDS 4.1 (Service Development Studio) is used as development tool for simulating IMS environment. It contains a network simulator (developed from Sun GlassFish communication server) for simulating IMS core network and testing agents for simulating IMS clients [10, 11]. Figure 6-7 shows a typical IMS client testing agent, which is capable to create, send, and receive IMS messages. It also provides client configuration, for instance, defining listening port, choosing TCP or UDP transport protocol, and saving the debug file.

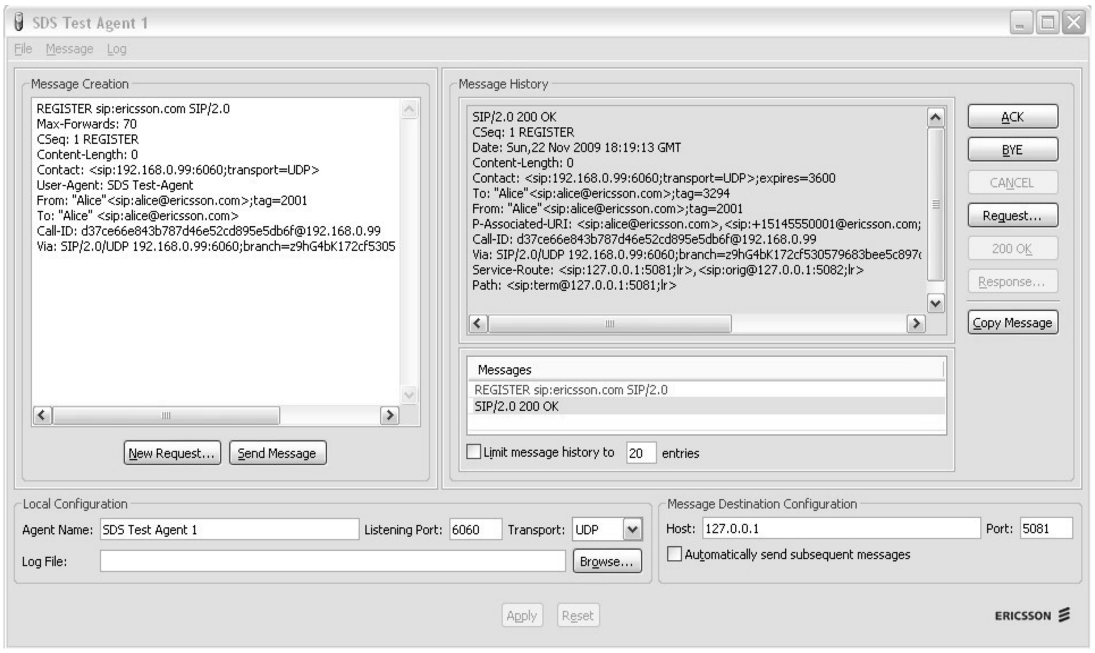
We implement text based instant messaging service to show availability of proposed system with two use scenarios: IMS client sends message to P2PSIP peer and P2PSIP peer sends message to IMS client. We manually define "P2PSIP MESSAGE" as request and "P2PSIP 200 OK" as response, as shown respectively in the following:

```
MESSAGE alice@ericsson.com P2PSIP/2.0
Max-Forwards: 70
CSeq: MESSAGE
Content-Length: 20
Contact: 586
From 586
To: alice@ericsson.com
Call-ID: 517846
Via: 586 158.36.228.48:9586; 512 158.36.228.48:9512

How are you?
```

```
P2PSIP/2.0 200 OK
To: alice@ericsson.com
From: 586
Contact: 586
CSeq: 200 OK
Content-Length: 0
From 586
Via: 586 158.36.228.48:9586; 512 158.36.228.48:9512
```

The system is deployed separately on a platform with Windows XP professional system, 2*2.4G Intel Core CPU and 3G memory. Wireshark [29] is used to monitor the message transmission. The testing shows that the system works well.



```
REGISTER sip:ericsson.com SIP/2.0
Max-Forwards: 70
CSeq: 1 REGISTER
Content-Length: 0
Contact: <sip:192.168.0.99:6060;transport=UDP>
User-Agent: SDS Test-Agent
From: "Alice"<sip:alice@ericsson.com>;tag=2001
To: "Alice" <sip:alice@ericsson.com>
Call-ID: d37ce66e843b787d46e52cd895e5db6f@192.168.0.99
Via: SIP/2.0/UDP 192.168.0.99:6060;branch=z9hG4bK172cf5305
```

```
SIP/2.0 200 OK
CSeq: 1 REGISTER
Date: Sun,22 Nov 2009 18:19:13 GMT
Content-Length: 0
Contact: <sip:192.168.0.99:6060;transport=UDP>;expires=3600
To: "Alice"<sip:alice@ericsson.com>;tag=3294
From: "Alice"<sip:alice@ericsson.com>;tag=2001
P-Associated-URI: <sip:alice@ericsson.com>,<sip:+15145550001@ericsson.com;
Call-ID: d37ce66e843b787d46e52cd895e5db6f@192.168.0.99
Via: SIP/2.0/UDP 192.168.0.99:6060;branch=z9hG4bK172cf530579683bee5c897
Service-Route: <sip:127.0.0.1:5081;lr>,<sip:orig@127.0.0.1:5082;lr>
Path: <sip:term@127.0.0.1:5081;lr>
```

```
REGISTER sip:ericsson.com SIP/2.0
SIP/2.0 200 OK
```

Limit message history to: 20 entries

Agent Name: SDS Test Agent 1
Listening Port: 6060
Transport: UDP
Log File:
Host: 127.0.0.1
Port: 5081

Figure 6-7: Ericsson IMS Test Agent

## 6.8. EVALUATION

### Number-of-Hops and Delay Measurement

We assume that $N$ is the number of P2PSIP peers in the overlay, including $S$ CSPs. We first consider the number of hops in Use Scenario 1 (Figure 6-4 in Section 6.7). According to Chord routing algorithm, the average number of hops of "PingRequest" is $\log_2(N/S)$. In addition to 3 hops among IMS clients, PIGW, CSP and SOS server, the average number of hops in Use Scenario 1 is $3+\log_2(N/S)$. As to Use Scenario 2 (described in Figure 6-5 in Section 6.7), the number of hops is equal to 3.

Then we measure delays of two use scenarios (in Section 6.7). We first select an IMS client (with ims id: alice@ericsson.com) as the initiator and randomly select 20 P2PSIP peers as destinations. We send the request and measure the latency between "SIP MESSAGE" sent out from IMS client and "P2PSIP 200 OK" received in PIGW260. We get the average delay equal to 326ms. Using similar method, we get the delay for Use Scenario 2 as 408ms. According to the simulation result of num-of-hops and delays, we believe that the proposed interconnecting system architecture is feasible.

**Security Assessment**

The proposed system is capable to provide secure services. Let us look at two use scenarios represented in Figure 6-4 and Figure 6-5.

When IMS client initiates the session (Figure 6-4), requested message travels through PIGW, CSP and a few intermediate peers. Since PIGW and CSP are trusted entities, that means, security breaches mainly exist in “PingRequest” multicast process. For example, the “PingRequest” multicast may be modified, dropped, or replayed by malicious intermediate peers. With the implementation of subjective logic based trust enhancement, we reduce security vulnerability of this part of the system.

P2PSIP request, initiated by a peer (Figure 6-5), needs to travel through two intermediate peers before reaching IMS client (peer->CSP->PIGW->IMS client). Since CSP and PIGW are trusted entities, the session process is trusted.

However, similar with previous proposal (in Section 5.3), it might happen that, in the worst case, all “PingRequest” multicast messages are intercepted by a few malicious peers. In this case the request will fail.

# Chapter 7. Secure Architecture for "P2PSIP Client"

In this chapter, we propose a thin client based approach to improve security of P2PSIP systems. We introduce a special type of entity, called "P2PSIP Client" and propose a solution based on Thin Client P2PSIP Gateway (TC-PPSG), which acts as a normal peer in P2PSIP network, and a HTTP application server in Internet. The main issues considered here are: security, identity mapping, "push" technology, etc.

## 7.1. INTRODUCTION

In this Chapter, we study a special type of entity, called "P2PSIP client", which is now discussed in IETF P2PSIP Working Group [18]. P2PSIP client is the node who participates in overlay but does not provide distributed transport and storage functions. In order to access services, a possible "client protocol" is defined in Internet-drafts proposed in [105, 118]. Figure 7-1 illustrates the basic concept. The client protocol provides a mechanism for exchanging information between client and a normal peer. The normal peer (for instance, peer 20 in Figure 7-1) acts as "association peer", helping client (the triangle) to access P2PSIP services. On one hand, it stores client related information so that peers in the overlay is able to locate the client; and on the other hand, it handles client request for transport service functions (for example, send P2PSIP request).

Client protocol is IP layer based request-response protocol, which defines how client finds and interacts with its associated peer; how client creates, maintains, and terminates session with a normal peer. Client protocol is supposed to support a function subset included in peer protocol.

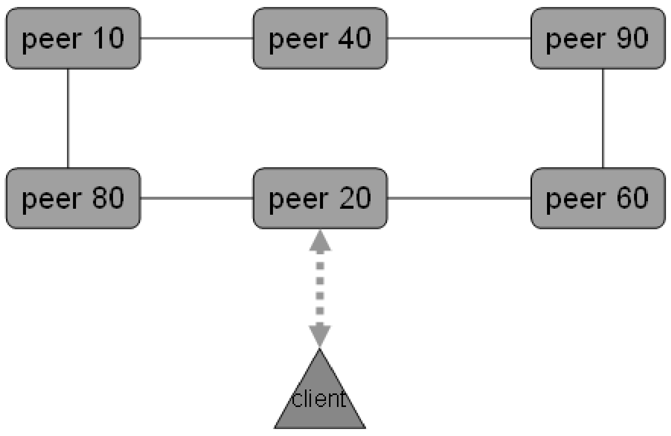


Figure 7-1: Client Protocol Approach

However, client protocol has a few disadvantages. Firstly, it does not consider at all security and privacy issues. On one side, associated peer has to serve client based on its requests, no matter whether client is malicious or not; on the other side client is uncertain about the trustworthiness of its associated peer. Since peer and client distrust each other, subsequent data transmissions may result in violation of data confidentiality, integrity and privacy.

Secondly, client protocol does not propose appropriate roaming mechanism. Roaming might happen when client notices a better hosting service. For example, a moving portable device may find a new associated candidate peer with faster IP connection. On the other hand, the overloaded associated peer might suggest its clients turn to a better option.

Thirdly, the proposed client protocol lacks convincing prototype implementations that validate the concept.

In this chapter, we study an alternative solution, which is based on thin client model. Besides, some approaches proposed in Chapter 5 can be reused in system architecture for security enhancement. These approaches include CSP based security, subjective logic based trust enhancement, Enrolment and Authentication server, PKI-certificate security, etc.

## 7.2. THIN CLIENT COMPUTING

Thin Client computing offers the promise of easier-to-maintain computational services with reduced total cost of ownership [125]. In this mode of computing, most of the functionality is located on the server side, while the client device only performs very simple display and query functions. One typical example of this mode of computing is using browsers to explore and access services. With the help of web browsers, it is possible to communicate with the Application Server somewhere in the network.

Thin client mode computing greatly reduces development cost, easier operation effort because most of operations are performed on server side. Besides, the system architecture provide better security (e.g. protect against theft, damage, malware, spyware, and viruses, etc) due to the fact that application related data are stored in central server instead of thin client.

Portable devices today are equipped with various kinds of browsers. So, these devices could become "thin clients" using thin client computing mode. In the following sections, we study web-browser-based thin client computing mode to access P2PSIP services. The web interface is provided through a slim web browser. The thin client supported functionality should be able to translate the P2PSIP signaling into HTTP messages and present relevant content as HTML-pages.

## 7.3. SYSTEM ARCHITECTURE

In this section, the proposed thin client based architecture is described. After that, we specify related solutions, including TC-PPSG internal, security, identity mapping, "push" technology, etc.

### 7.3.1 Architecture Overview

The proposed architecture involves three main parts: P2PSIP client, Thin Client P2PSIP Gateway (TC-PPSG), and P2PSIP overlay (including peers, CSP peers, E&A servers, SOS servers, described in Chapter 5), as shown in Figure 7-2.

P2PSIP client, which can be a mobile phone, laptop, PC, etc., is connected to the Internet. It contains a web browser for accessing Internet. TC-PPSG is the interconnecting gateway that is an application server in Internet and a normal P2PSIP peer in P2PSIP network. P2PSIP overlay is assumed to be secure overlay that includes security enhancement approaches described in Chapter 5.

For locating a peer/resource in P2PSIP overlay, client sends a HTTP request to the TC-PPSG. TC-PPSG parses HTTP message, catches useful information (e.g. source, destination, public IP, etc), generates P2PSIP message, and forwards request to the overlay. The overlay is responsible to locate the destination peer. Finally, the session between client and normal peer can be established.

The reversal session to locate a P2PSIP client is similar.

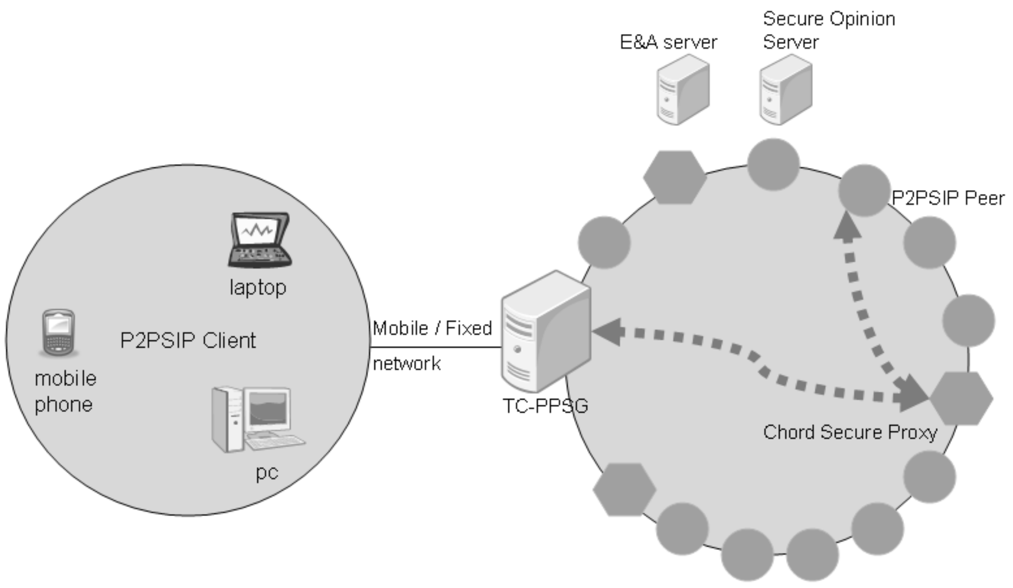


Figure 7-2: Thin Client P2PSIP Architecture

## 7.3.2 TC-PPSG Internal

Thin Client P2PSIP Gateway (TC-PPSG) is the key inter-working unit, acting as a bridge between client and peer. Note that TC-PPSG is deployed as P2PSIP application server with the functionality of a normal P2PSIP peer, and as an Internet application server. There are four main components inside a TC-PPSG unit (see Figure 7-3):

- Thin Client inter-working component. This part consists of a servlet container for receiving HTTP request and replying corresponding response. It communicates with ID Management component for authenticating user ID. It forwards HTTP messages to P2PSIP interworking component. It also communicates with O&M Provisioning component for manual configuration and provisioning.

- P2PSIP Interworking. This part contains a translation logic component that handles the translation task between P2PSIP and HTTP messages; a forwarding logic that sends message out to P2PSIP network properly or forwards to Thin Client Interworking component.

- ID Management. This component manages client identity. It contains an ID database that stores user identifiers, passwords and P2PSIP identities. It assists the other three components with ID authentication and configuration.

- O&M Provisioning. System administrator, for maintenance, manual configuration and provisioning, operates this component.

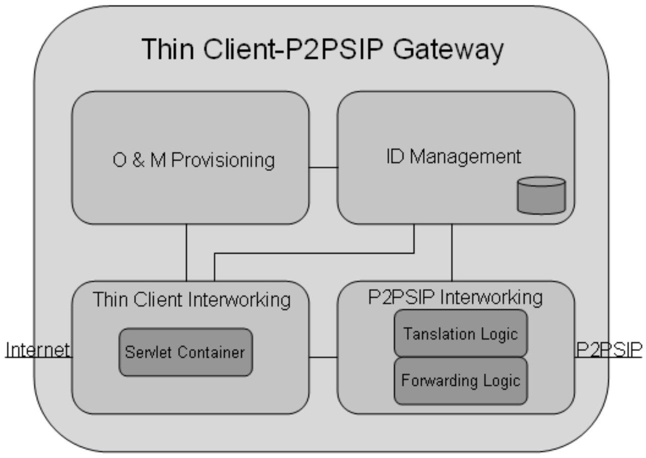


Figure 7-3: TC-PPSG Structure

### 7.3.3 Security and Privacy

All connections must be encrypted for providing data confidentiality and integrity.

Since TC-PPSG is assumed to be a trusted entity and the only contact point for clients, it is capable to protect confidentiality of client data (e.g. client ID, password, IP, etc) from leaking.

P2PSIP overlay security is provided by solutions described in Chapter 5 or referred to in the publications [121, 119, 122, 124].

### 7.3.4 Identity Mapping

Interconnecting P2PSIP with Internet requires the mapping function between user Internet identity and its P2PSIP identity. There are two options for handling user identities, “Connecting gateway” in which only TC-PPSG is connected into the overlay and “Connecting client and gateway” in which both TC-PPSG and P2PSIP client are connected into the overlay. Principally, we support both of the options.

In the first option, TC-PPSG is registered into P2PSIP overlay while client is hidden. The mapping model is as illustrated in Figure 7-4. P2PSIP Client “borrows” the identity of TC-PPSG (with overlay ID 260) to send and receive P2PSIP messages. For example, Internet-oriented user identities (e.g. Alice, Bob, and Coco, etc) are mapped to TC-PPSG identity (ID 260). This approach would cause no problem when the session is initiated from P2PSIP client, however, the reverse session (when P2PSIP peer want to talk with P2PSIP client) might require additional functionality to distinguish corresponding P2PSIP client. A possible option is to define a P2PSIP extension header to carry client information. The extension header starts with a header field “Client”, and value field recording user identity. An example is represented in Section 7.5.

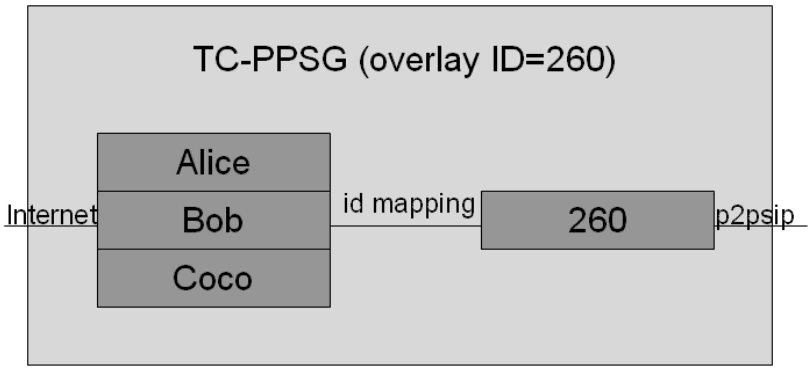


Figure 7-4: Identity Mapping Approach

The second approach maps client identity to a specific P2PSIP identifier. In this approach, TC-PPSG applies specific P2PSIP identity for each client (for instance,

)

apply to E&A server, etc). Client identity is defined to be the next hop successor of TC-PPSG. For instance (See Figure 7-5), TC-PPSG applies three IDs (261, 263, 270), which are logically its own direct successors in the overlay. According to Chord overlay algorithm, all data traffic going to these IDs have to travel through TC-PPSG. Therefore, as the only contact point, TC-PPSG is capable to relay the connections for its clients.

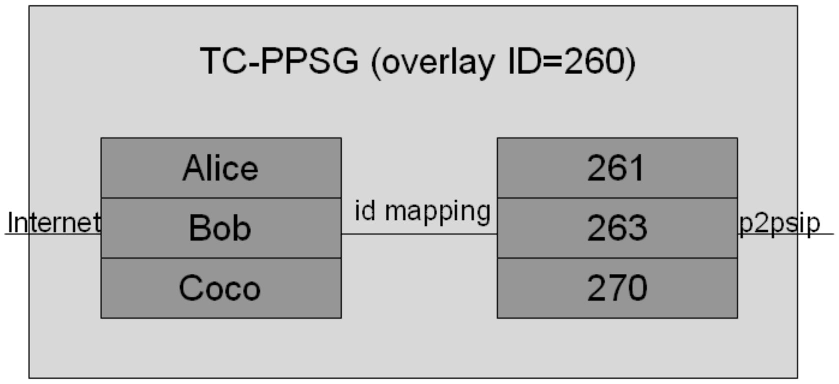


Figure 7-5: Identity Mapping Alternative

### 7.3.5 "Push" Technology

One important technical issue is that thin client browser should be able to receive updated information automatically so that from end user point of view, the message is "pushed" into browser. One simple solution is HTML auto-refresh mechanism [30]. Each time client needs the data, it requests server expressly to reload the whole web page. The interaction between client and server is shown in Figure 7-6.

Auto-refresh functionality can be achieved by either setting "META" attribute of HTML (for example, "<meta http-equiv="refresh" content = 30>" means refreshing the page each 30 seconds) or calling embedded page script function, both of which have been widely implemented in traditional web applications.

Although this kind of mechanism requires higher bandwidth and wastes unnecessary data traffic, it is suitable for most of the Fixed / Mobile browsers currently exist.

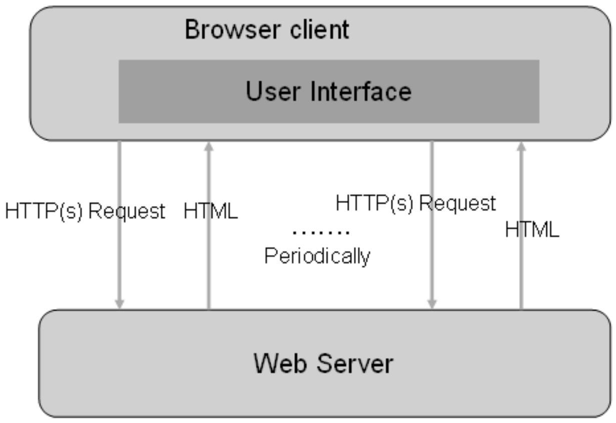


Figure 7-6: Web Auto-Refresh Mechanism

Another more sophisticated solution is based on AJAX (Asynchronous JavaScript and XML) [50], which interweaves technologies including JavaScript, Document Object Model (DOM), Cascading Style Sheets (CSS), and Dynamic HTML (DHTML). In this mechanism, client browser uses JavaScript to periodically interact with the server by reloading dynamic "tag" elements. Updated element and value are encapsulated in XML document and transferred back by HTTP protocol. The interaction process is shown in Figure 7-7. Compared with web refresh mechanism, this approach greatly reduces network traffic; however, only a few advanced browsers (e.g. Opera Mobile [17], IE mobile [12], etc) currently support AJAX.

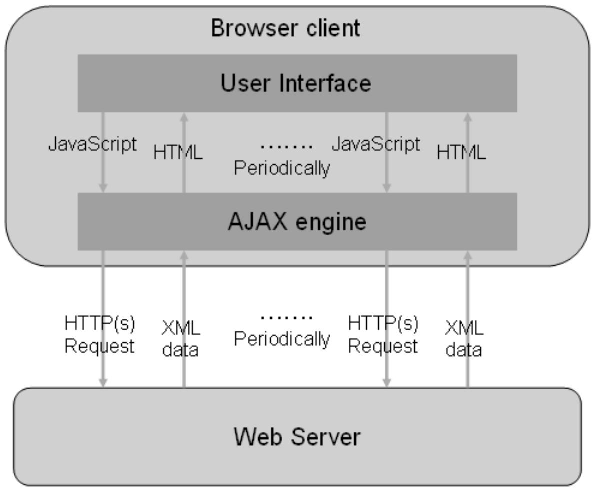


Figure 7-7: AJAX mechanism

In Thin Client P2PSIP Architecture, we propose to use both of above technologies. The first approach is supported by most of resource-limited clients. The second approach can be implemented in advanced browsers that support AJAX technology. During the connection establishment, TC-PPSG should be able to detect client device/browser type, capability and then intelligently choose different "Push" technologies.

## 7.4. USE SCENARIOS

In this section, we describe the detail of use scenarios that are supported by the proposed thin client based P2PSIP architecture. We define "P2PSIP MESSAGE" as the request and "P2PSIP 200 OK" as the response.

### 7.4.1 Client Registration

Use case 1 (see Figure 7-8) describes P2PSIP client registration process to TC-PPSG. Note that we only present "Connecting gateway" option. Possible messages flows are:

1) Source peer sends "HTTP Registration" request to TC-PPSG. The request should include at least user name and password.

2) TC-PPSG validates user name and password. For instance, it checks if user name is unique or not, and if password is not too simple. If everything is OK, it returns a "200 OK" response. Otherwise, it will return an error message.

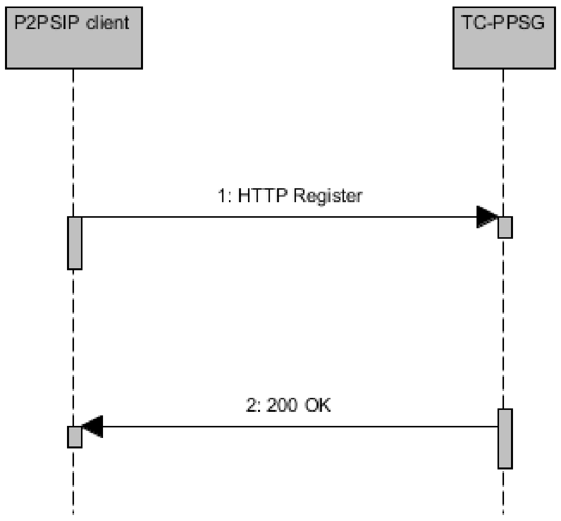


Figure 7-8: Client Registration

## 7.4.2 Instant Messaging

Use case 2 (see Figure 7-9) describes how P2PSIP client and peer contact each other by sending instant message. Note P2PSIP client has been registered to TC-PPSG. Possible messages flows are:

1) Source peer sends "HTTP message" to TC-PPSG. The instant message is encapsulated inside HTTP body.

2) TC-PPSG translates HTTP request to "P2PSIP MESSAGE", which is then forwarded to the overlay. Finally, it reaches destination peer.

3) Destination peer replies with "P2PSIP 200 OK".

4) TC-PPSG replies a "HTTP 200 OK" to client browser.

5) Destination peer, who wants to talk with client, initiates a "P2PSIP MESSAGE" to TC-PPSG.

6) TC-PPSG replies with "P2PSIP 200 OK".

7) Instant message is pushed to client browser.

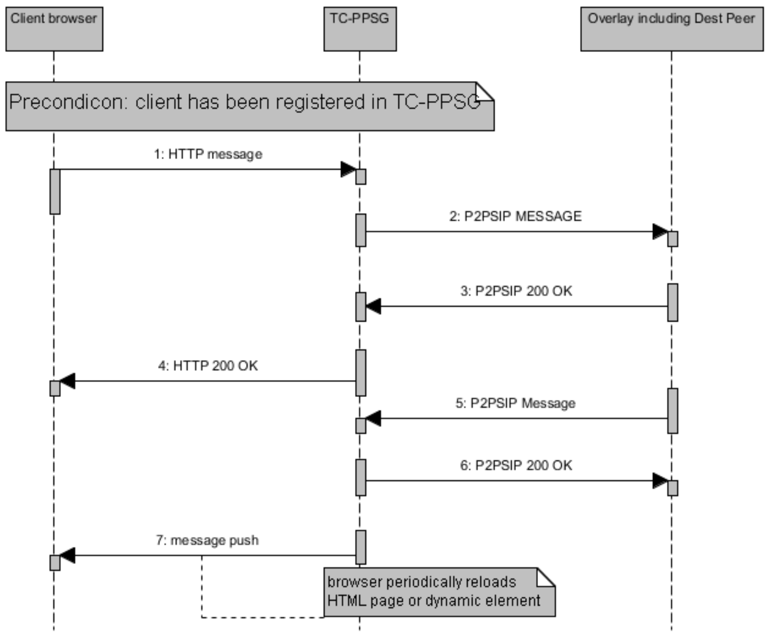

Client browser
TC-PPSG
Overlay including Dest Peer
Precondicon: client has been registered in TC-PPSG
1: HTTP message
2: P2PSIP MESSAGE
3: P2PSIP 200 OK
4: HTTP 200 OK
5: P2PSIP Message
6: P2PSIP 200 OK
7: message push
browser periodically reloads HTML page or dynamic element

Figure 7-9: Instant Messing Use Scenario

### 7.4.3 Voice over IP

Use case 3 (see Figure 7-10) describes how P2PSIP client initiates VoIP session with a P2PSIP peer. Note client browser has been equipped with advanced technologies (e.g. flash support, etc) that support VoIP. Besides, client has been registered to TC-PPSG. Possible messages flows are:

1) Source peer sends "HTTP invite" request to TC-PPSG. Session related information (opening port, codec, etc) is included in HTTP body.

2) TC-PPSG translates HTTP request to "P2PSIP INVITE", and then forwards it to the overlay. Finally, the request reaches destination peer.

3) Destination peer replies with "P2PSIP 200 OK".

4) "P2PSIP 200 OK" is translated to "HTTP 200 OK", which is forwarded to client browser.

5) Session negotiation and establishment.

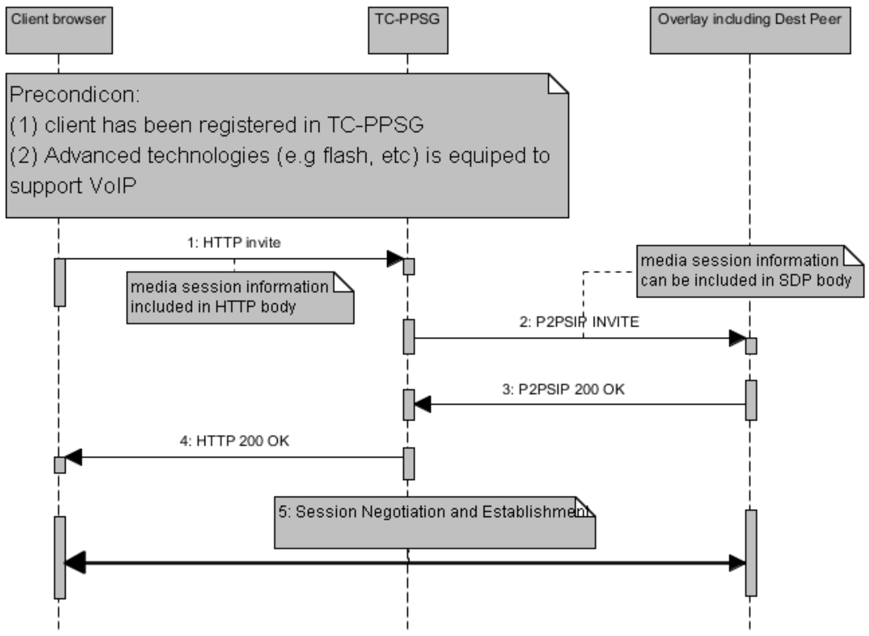

Client browser
TC-PPSG
Overlay including Dest Peer
Precondicon:
(1) client has been registered in TC-PPSG
(2) Advanced technologies (e.g flash, etc) is equiped to support VoIP
1: HTTP invite
media session information included in HTTP body
media session information can be included in SDP body
2: P2PSIP INVITE
3: P2PSIP 200 OK
4: HTTP 200 OK
5: Session Negotiation and Establishment

Figure 7-10: VoIP Use Scenario

## 7.5. IMPLEMENTATION

Our previous P2PSIP implementation (the overlay contains 512 P2PSIP peers, including with 16 CSP peers, and 496 normal peers) [120, 122] is extended to model thin client architecture. We extend the functionality of an existing peer to be TC-PPSG, which is deployed as a HTTP application server in Internet, and a P2PSIP peer in P2PSIP network. Apache Derby is used to simulate embedded database inside TC-PPSG.

We build a text based instant message use scenario. A P2PSIP "MESSAGE" is defined as request message (an example is presented below). The "MESSAGE" starts with a few request header fields useful for initiating session, such as client id, source, destination, and routed intermediate parties; and follows by the main message body.

```
MESSAGE 260 P2PSIP/2.0
Client: alice
Max-Forwards: 70
CSeq: MESSAGE
Content-length: 70
Contact: 586
From: 586
To: 260
Call-ID: 4857294
Via: 512 192.168.0.99:9512;

Hello Alice, How are you?
```

After that, a corresponding "200 OK" is defined as success response, as below:

```
P2PSIP/2.0 200 OK
To: 586
From: 260
Contact: 586
CSeq: 200 OK
Content-Length: 0
Via: 260 127.0.0.1:9260
```

We use Openwave simulator V7 [16], Opera mini browser [17], and IE 8.0 browser [12] to test the system. The browsers are capable to initiate a "MESSAGE" request and reach a P2PSIP peer. However, the transmission of an instant message from a peer to a client would take at most 30 seconds before it will be displayed in the browser due to the refresh time interval. Generally, the functionality works well although the displayed HTML UI is not user friendly due to small size of mobile screens.

## 7.6. EVALUATIONS

Table 7-1 presents the comparison of our proposed thin client solution with current proposal "client protocol".

***Table 7-1 Two Solutions Comparison***

| | Client protocol solution | Thin Client System architecture |
|---|---|---|
| Reliability | No guarantee because associated peer and client are not trusted each other. Malicious associated peer is capable to discard, replay, and modify client request. | Guaranteed because TC-PPSG is the trusted entity. |
| Use scenario supported | All use scenarios as P2PSIP peer, such as Instant Messing, Presence services, VoIP, etc. | Supports most of use scenarios. Depends on browser capabilities. For example, need flash supported browser to support VoIP. |
| Authentication | Currently no. Associated peer has to serve for client. | Client Internet ID. Only registered client can use services. |
| Integrity | Certificate supports digital signature. | Certificate supports digital signature. |

| | | |
|---|---|---|
| Data confidentiality | Certificate based encryption and decryption. | Certificate based encryption and decryption. |
| Privacy | No guaranteed. Malicious associated peer can intercept the connection and spy sensitive privacy | TC-PPSG is trust entity that protects sensitive privacy for all clients. |
| Implementation | Very difficult to implement. Need two version applications, one for associated peer and one for client. | Implementation only in TC-PPSG. |
| End devices support | Targeted at most devices that support basic TCP/UDP protocols. Still need some CPU processing power, bandwidth, to support the applications. Client has to install the application before using it. | Supported by most devices that equip with browser.<br><br>No need to download application, however, needs to register first. |
| Mobility | Not defined yet. This makes roaming difficult to achieve. | No problem because of none associated peer is included. |
| Debug and maintenance | Difficult. Debug needs to consider both associated peer and client side. If the vulnerability is found, all the peers and clients need update their applications | Easy to debug and maintenance. All the functions are in TC-PPSG entity. |

The comparison shows client protocol solution is better in supporting P2PSIP use scenarios. However, thin client based system architecture is superior in other aspects. Due to lack of credential entities, client and associated peer distrust each other all the time, and this creates security, privacy and reliability problems. Currently, there is no efficient mechanism proposed in "client" solution. In contrast, our proposed thin client solution offers guarantee on system reliability and privacy protection because TC-PPSG is regarded as trusted entity. Also, our previously described approaches (e.g. CSP based security, subjective logic based trust enhancement, etc) are also used to enhance security in P2PSIP overlay.

Besides, the concept of client protocol has difficulty in implementation, debugging and later maintenance because all functionalities require simultaneous changes on both associated peers and client sides. Even worse, because the client protocol is not mature, there is no convinced prototype to support the proposals. In contrast, our solution inherits the advantage of thin client computing on fast development and easy maintenance since all functions are done only on TC-PPSG side.

Furthermore, our approach is independent of device location and therefore provides better roaming capability (as long as the Internet connection is available) than client protocol based approach.

In summary, we believe that our proposed solution, currently, is better than "client protocol" based proposal, after consideration and comparison of technical aspects.

# Chapter 8. Conclusions and Future works

This Chapter concludes thesis work. It starts with retrospect of P2P research, followed by description of our research work. Finally, open issues and possible solutions are also presented.

## 8.1. CONCLUSIONS

P2PSIP is a promising future trend. It is supposed to be a communication protocol that supports a set of real-time multimedia services, such as presence, instant messaging, Voice over IP (VoIP), etc. Unfortunately, P2PSIP is far from mature and still needs time for solving critical technical challenges. In the thesis, we mainly answer four critical questions.

Firstly, we propose and analyse a few approaches to reduce delays during session initiation, and therefore improve system efficiency. The improvement includes revision of lookup algorithm, geographical association, cache mechanism, hierarchical architecture, optimized routing, etc. The following mathematical analysis proves performance improvement of proposed approaches. According to testing results, we conclude that cache mechanism is probably one of best improvement options.

Secondly, we consider security issues and propose new solutions to enhance the system security. Solutions include centralized proxy based approach, and subjective logic based trust enhancement. Both solutions contain centralized proxy/server elements for management of security functions and parameters. Although this contradicts to decentralization requirement of P2PSIP, we believe it is necessary for security and privacy enhancement. We also combine these two solutions to achieve optimal security protection.

Thirdly, we illustrate shortcoming of current proposal of “client” protocol and study an alternative solution, which is based on Thin Client architecture. The proposed Thin Client P2PSIP Gateway (TC-PPSG) acts as a gateway for handling translation tasks between P2PSIP overlay and Internet.

Finally, a possible inter-working system model is introduced to interconnect P2PSIP network and future All-IP based IMS network. The system architecture includes the inter-working solution, and use security mechanisms according to previous proposals..

## 8.2. FUTURE WORKS

The research of P2PSIP is still in the initial stage. There are still many critical issues to be addressed. Following areas are critical for future adoption of P2PSIP:

**Extended Usage Scenario**

The study of P2PSIP usage scenario is one of the important future works. Although a few use scenarios have been presented and implemented to validate proposed solutions, they are mostly focused on session initiation phase. This thesis work considers little about further services after initiation, such as session modification and negotiation tasks. Besides, some other advanced use scenarios, such as presence services, Voice over IP (VoIP), etc., need to be investigated.

**DHT overlay flexibility**

Another challenge is DHT overlay flexibility problem Currently, Chord algorithm is suggested as the only mandatory overlay technology for supporting P2PSIP communication. However, many overlay algorithms (e.g. Kademlia, CAN, etc) are supposed to be coexisting in near future. Therefore, one important issue is to investigate whether solutions proposed in the thesis suitable for other overlay environments based on different algorithms.

Besides, DHT overlay should be capable to inter-work among different overlay technologies. In [74] authors propose that each overlay selects a “Super Node” to construct another DHT overlay and act as an relay peer.

### Service Discovery

Currently, the design of P2PSIP protocol does not pay much attention to service discovery mechanisms. For example, in peer bootstrap process, important elements, such as bootstrap peers, enrolment servers, STUN/TURN servers, etc., are assumed to be pre-configured. This might not be extendable especially because these elements might be not always reliable (for instance, some of them might become offline, or service unreachable temporally). Therefore, it is necessary to design common services discovery mechanisms suitable for P2PSIP overlay.

### Reliability

The implementation and result in thesis work is given an assumption: overlay is stable without much churn (peer leave/join the overlay frequently). However, in reality portable P2PSIP peers (with limited CPU power, unstable wireless connection, and small cache) are possible to cause much churn and expose negative effect in overlay, according to research of [69, 72, 87]. Let us consider an even worse case (lowest layer in Figure 3-8), in which most of overlay peers are unreliable portable devices. This typical case is expected to be quite inefficient. Therefore, it is necessary to make a detailed analysis of such typical use scenario to develop novel approaches with improved performance.

# APPENDIX A. SIP Methods

RFC 3261 defines six basic SIP methods for session initiation: “INVITE”, “100 Trying”, “180 Ringing”, “ACK”, “200 OK”, “BYE”. In this section, we show an example for each of SIP method. Corresponding interaction diagram is shown in Figure 2-7.

**INVITE**

```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bK776asdhds
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.atlanta.com
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 142
```

**100 Trying**

```
SIP/2.0 100 Trying
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Content-Length: 0
```

**180 Ringing**

```
SIP/2.0 180 Ringing
Via: SIP/2.0/UDP server10.biloxi.com;branch=z9hG4bK4b43c2ff8.1;received=192.0.2.3
Via: SIP/2.0/UDPbigbox3.site3.atlanta.com
```

```
    ;branch=z9hG4bK77ef4c2312983.1;received=192.0.2.2
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
Contact: <sip:bob@192.0.2.4>
CSeq: 314159 INVITE
Content-Length: 0
```

**ACK**

```
ACK sip:bob@192.0.2.4 SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds9
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>; tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 ACK
Content-Length: 0
```

**200 OK**

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP server10.biloxi.com;branch=z9hG4bKnashds8;received=192.0.2.3
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com
    ;branch=z9hG4bK77ef4c2312983.1;received=192.0.2.2
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bK776asdhds ;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.atlanta.com
CSeq: 314159 INVITE
Contact: <sip:bob@192.0.2.4>
Content-Type: application/sdp
Content-Length: 131
```

**BYE**

```
 BYE sip:alice@pc33.atlanta.com SIP/2.0
 Via: SIP/2.0/UDP 192.0.2.4;branch=z9hG4bKnashds10
 Max-Forwards: 70
```

```
From: Bob <sip:bob@biloxi.com>;tag=a6c85cf
To: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 231 BYE
Content-Length: 0
```

# APPENDIX B. P2PSIP System Specification

Each peer in P2PSIP system handles three functionalities: peer initiation function (for participating the overlay), request sending function (for initiating the request), and request handling function (for processing the received messages). The following is the detail algorithm specifications.

**Peer Initiation**

```
for each peer{

      //ask enrolment server for finger table
      getFingerTable(my_id);

      //create a embedded cache database
       createDerbyServer("peer " + my_id);
       createPeerCache("peercache "  + my_id);

     //listening on a specific port, which is defined as 9000+ peer ID
      openServerSocket(my_port);

}
```

**Sending Request**

```
//when source peer wants to initiate a request
private void jButtonActionPerformed(){

      //encapsulate a P2PSIP "INVITE" request
      String invite = generateInvite();

      //may be destination peer profile is already in cache history
      String dest_IP_port = getCacheRecord(dest_id);
      if(dest_IP != null){
            //destination peer is already in the cache, just send the request
            sendRequestDatagram(dest_IP_port);

        }else{

                //find a downstream peer according to Chord algorithm
                String downstream_peer = lookupDownstreamPeer();
                sendRequest(invite, downstream_peer);

            }

}
```

**Request Handling**

```
while receiving (message){

    //read the message, parse it
    int source_id = getSourceID(message);
    int dest_id = getSourceID(message);

    if(message is "INVITE") {

        if(i am intermediate peer){

            //find a downstream peer according to Chord algorithm
            String downstream_peer_id = lookupDownstreamPeer();
            forwardInvite(invite, downstream_peer_id);

        }
        if(i am destination peer){

            // generate "180 Ringing" response
            String resp_180Ringing = generate180Ring();
            sendResponse(resp_180Ringing, source_id);

        }

    }
```

```
        else if (message is “180 Ringing”){

            //only source peer is able to receive “180 Ringing”
            //do nothing currently

        }
    }
```

# APPENDIX C. Chord Secure Proxy (CSP) Specification

One of the critical tasks of Chord Secure Proxy (CSP) is to receive and handle request messages from P2PSIP peer. According to the parameters in request messages, CSP chooses different handling mechanisms. The algorithm is described below.

**Handling Request**

```
while receiving (message){

    //read the message, parse it
    int source_id = getSourceID(message);
    int dest_id = getSourceID(message);
    int call_id = getCallID(message);
    if(message is "INVITE") {
        //temporally store message into cache
        insertDerbyRecord(message, call_id);

        // generate "PingRequest" message.
        //set source id to my_id.
        String pingRequest = generatePingRequest(dest_id, my_id);

        //calculate a few successors as downstream peers,
        //according to Chord algorithm
        int [] downstream_peers = calculate_downstream();

        //multicast pingRequest to a few successors
        sendMulticast(pingRequest, source_id);

    }else if (message is "200 OK"){
       //destination peer returns the response,
       //should forwards original "INVITE" message
```

```
        sendRequest(message, dest_IP);

        //delete the internal cache record
         deleteDerbyRecord(call_ id);
    }}
```

# APPENDIX D. Secure Opinion Server (SOS) Specification

Secure Opinion Server (SOS) is deployed as a HTTP application server. The main functionality component is a servlet "OpinionResponse", which receives a list of route candidates, applies subjective logic based trust calculations, and selects the most trustful route option. Below is the SOS deployment description.

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee" xmlns:web="http://java.sun.com/xml/ns/javaee/web-
app_2_5.xsd" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" id="WebApp_ID" version="2.5">
  <display-name>secureopinionserver</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.htm</welcome-file>
    <welcome-file>default.jsp</welcome-file>
  </welcome-file-list>
  <servlet>
    <description></description>
    <display-name>OpinionResponse</display-name>
    <servlet-name>OpinionResponse</servlet-name>
    <servlet-class>no.uia.opinionservlet.OpinionResponse</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>OpinionResponse</servlet-name>
    <url-pattern>/OpinionResponse</url-pattern>
  </servlet-mapping>
</web-app>
```

Three major functionality algorithms are described in the following:

**Opinion Calculation**

```
  while receiving (message){

       //get the destination peer profile (id, ip, port)
       int destProfile = getDestProfile(message);

       //read the xml message, get a list of route options
       String [] route = getRouteOptions(message);
       float [] trust = 0.0;
       float [] distrust = 0.0;
       float [] uncertainty = 0.0;

       float[] summary = 0.0;

       for each route[i] {
            //get its trust, distrust, and uncertainty value from Opinion DB.
            //should have the encourage rule for intermediate peers, introduced next
            trust[i] = getTrust(i);
            distrust[i] = getDistrust(i);
            uncertainty[i] = Uncertainty(i);

         //calculate a summary V value for each route.
         summary[i] = calculate(trust[i], distrust[i], uncertainty[i]);
    }

    //choose a route that owns highest value of summary[j]
    sendRoute(route[j], destProfile);

}
```

**Opinion Encouragement Function**

```
For each intermediate peer {

       //intermediate peers who have contributions to overlay should be encouraged.
```

```
          increaseTrust();
          decreaseDistrust();
          decreaseUncertainty();

}
```

**Opinion Punish function**

```
For each 10 minute{

        //Check each opinion DB record
        if(updated){
             // do nothing
        }else{

             //this peer is regarded have no contribution to overlay
             // decrease trust value, increase distrust value, decrease uncertainty value
             // according to the definition
             decreaseTrust();
             increaseDistrust();
             increaseUncertainty();

        }

}
```

# APPENDIX E. P2PSIP IMS Gateway (PIGW) Specification

P2PSIP IMS Gateway (PIGW) is deployed as an IMS Application server, as well as a normal P2PSIP peer. We achieve the functionality of PIGW through the revision of a P2PSIP node (with ID 260) in previous implementation. The following shows the deployment description.

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee" xmlns:web="http://java.sun.com/xml/ns/javaee/web-
app_2_5.xsd" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" id="WebApp_ID" version="2.5">
  <display-name>peer260</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.htm</welcome-file>
    <welcome-file>default.jsp</welcome-file>
  </welcome-file-list>
  <servlet>
    <description></description>
    <display-name>Pigw260</display-name>
    <servlet-name>Pigw260</servlet-name>
    <servlet-class>no.uia.httpservlet.Pigw260</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>Pigw260</servlet-name>
    <url-pattern>/Pigw260</url-pattern>
  </servlet-mapping>
  <servlet>
    <description></description>
    <display-name>TestingServlet</display-name>
    <servlet-name>TestingServlet</servlet-name>
    <servlet-class>no.uia.httpservlet.TestingServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>TestingServlet</servlet-name>
    <url-pattern>/TestingServlet</url-pattern>
  </servlet-mapping>
</web-app>
```

Besides, PIGW contains a SIP servlet, named “HanldingSipServlet”, for receiving SIP messages from IMS network, handling the translation task between IMS and P2PSIP. The servlet deployment description is:

```
<?xml version="1.0"?>
<!DOCTYPE sip-app PUBLIC "-//Java Community Process//DTD SIP Application 1.0//EN"
"http://www.jcp.org/dtd/sip-app_1_0.dtd">

<sip-app>
<servlet>
  <servlet-name>HandlingSipServlet</servlet-name>
  <display-name>HandlingSipServlet</display-name>
  <description></description>
  <servlet-class>
    no.uia.sipservlet.HandlingSipServlet
  </servlet-class>
</servlet>

<servlet-mapping>
<servlet-name>HandlingSipServlet</servlet-name>

<pattern>
  <or>
  <equal>
    <var>request.method</var>
    <value>MESSAGE</value>
  </equal>
</or>
</pattern>
```

```
</servlet-mapping>

</sip-app>
```

The major functionality of PIGW is to translate data signals between P2PSIP and IMS network. The detail is described in the following:

**Message Translation**

```
while receiving (message){

    if(message is from peer){

        //ask enrollment server about mapping of peer ID to SIP URI
        String sourceSipUri = getMapping(source_id);
        String destSipUri = getMapping(dest_id);

        //replace peer ID to SIP URI, encapsulate a P2PSIP message
        String imsMessage = generateSIPMessage();

        //according to SIP URI, IMS core is able to locate destination UA
        sendMessage(imsMessage);
    } else if(message is from IMS network){

        //ask enrollment server about mapping of SIP URI to peer ID
        int sourceID = getMapping(source_uri);
        int destID = getMapping(dest_uri);

        //replace peer ID to SIP URI, encapsulate a P2PSIP message
        String p2psipMessage = generateP2PSIPMessage();

        //according to SIP URI, IMS core is able to locate destination UA
        sendMessage(p2psipMessage);

    }
}
```

# APPENDIX F. Thin Client P2PSIP Gateway (TC-PPSG) Specification

Thin Client P2PSIP Gateway (TC-PPSG) is deployed as an HTTP application server, as well as a normal P2PSIP Peer. We achieve the functionality of TC-PPSG through the revision of a P2PSIP node (with ID 260) in previous implementation. It contains several HTTP servlets for receiving HTTP request and generating corresponding response. Here is the deployment description file.

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee" xmlns:web="http://java.sun.com/xml/ns/javaee/web-
app_2_5.xsd" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" id="WebApp_ID" version="2.5">
  <display-name>peer260</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.htm</welcome-file>
    <welcome-file>default.jsp</welcome-file>
  </welcome-file-list>
  <servlet>
    <description></description>
    <display-name>Pigw260</display-name>
    <servlet-name>Pigw260</servlet-name>
    <servlet-class>no.uia.httpservlet.Pigw260</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>Pigw260</servlet-name>
    <url-pattern>/Pigw260</url-pattern>
  </servlet-mapping>
  <servlet>
    <description></description>
    <display-name>TestingServlet</display-name>
    <servlet-name>TestingServlet</servlet-name>
    <servlet-class>no.uia.httpservlet.TestingServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>TestingServlet</servlet-name>
    <url-pattern>/TestingServlet</url-pattern>
  </servlet-mapping>
  <servlet>
    <description></description>
    <display-name>ValidateServlet</display-name>
    <servlet-name>ValidateServlet</servlet-name>
    <servlet-class>no.uia.httpservlet.ValidateServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>ValidateServlet</servlet-name>
    <url-pattern>/ValidateServlet</url-pattern>
  </servlet-mapping>
  <servlet>
    <description></description>
    <display-name>RegisterServlet</display-name>
    <servlet-name>RegisterServlet</servlet-name>
    <servlet-class>no.uia.httpservlet.RegisterServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>RegisterServlet</servlet-name>
    <url-pattern>/RegisterServlet</url-pattern>
  </servlet-mapping>
  <servlet>
    <description></description>
    <display-name>InstantMsgHttpServlet</display-name>
    <servlet-name>InstantMsgHttpServlet</servlet-name>
    <servlet-class>no.uia.httpservlet.InstantMsgHttpServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>InstantMsgHttpServlet</servlet-name>
    <url-pattern>/InstantMsgHttpServlet</url-pattern>
  </servlet-mapping>
```

```
</web-app>
```

One major task of TC-PPSG is to translate data signals between P2PSIP and Internet, as described in the following:

**Message Translation**

```
when receiving (HTTP Post){

        int callID = getCallID(message);
        String msg = getMessage(message);

        //ask enrollment server about mapping of SIP URI to peer ID
        int sourceID = getMapping(source_uri);
        int destID = getMapping(dest_uri);

        //encapsulate a P2PSIP message
        String p2psipMessage = generateP2PSIPMessage(source ID, destID, callID,
msg);

        //find a corresponding CSP that is responsible for the destination peer
        // and send p2psip message to the overlay
        int csp = lookupCSP();
        sendMessage(p2psipMessage, csp);

    }
}
when receiving (p2psip message){

        //parse useful information
        int callID = getCallID(message);
        String msg = getMessage(message);

        //ask enrollment server about mapping of SIP URI to peer ID
```

```
            int sourceUri = getMapping(source_id);
            int destUri = getMapping(dest_id);

            //store into database
            insertMsgRecord("Instant message", sourceUri, destUri, msg) ;

        }
    }
```

# APPENDIX G. Publication List

1. *Xianghan Zheng, Vladimir Oleshchuk, Hongzhi Jiao, A System Architecture for SIP/IMS-based Multimedia Services, Novel Algorithms and Techniques in Telecommunications, Automation and Industrial Electronics. pp.543-548, Springer, 2008.*

2. *Xianghan Zheng, Vladimir Oleshchuk, Providing Privacy in P2PSIP-based Communication Systems, in Proceedings of Norsk informasjonssikkerhet- konferanse, Sep, 2008: Kristiansand, Norway.*

3. *Xianghan Zheng, Vladimir Oleshchuk, Improving Chord lookup protocol for P2PSIP- based Communication Systems, in Proceedings of the 2009 International Conference on New Trends in Information and Service Science (3rd NISS), pp.1309-1314, IEEE, 2009: Beijing, China.*

4. *Xianghan Zheng, Vladimir Oleshchuk, A Secure Architecture for P2PSIP-based Communication Systems, in Proceedings of the 2nd International Conference on Security of Information and Networks (SIN 2009), pp.75-82, ACM, 2009: Gazimagusa, North Cyprus.*

5. *Xianghan Zheng, Vladimir Oleshchuk, Trust-based Framework for Security Enhancement of P2PSIP Communications Systems, in Proceedings of 4th International Conference for Internet Technology and Secured Transaction (ICITST-2009), pp.253-260, IEEE, 2009: London.*

6. *Xianghan Zheng, Vladimir Oleshchuk, A survey on peer-to-peer SIP based communication systems. Peer-to-Peer Networking and Applications, Springer, Jan, 2010.*

7. *Xianghan Zheng, Vladimir Oleshchuk, Improvement of Chord overlay for P2PSIP-based Communication Systems, International Journal of Computer Networks & Communications (IJCNC). Vol.1, No.3, Oct, 2009.*

8. *Xianghan Zheng, Vladimir Oleshchuk, The design of hierarchical, efficient, and secure P2PSIP communication systems, Information Theory and Practices. Security and Privacy of Pervasive Systems and Smart Devices. pp.253-260, LNCS volume 6033, Springer, 2009.*

9. *Xianghan Zheng, Vladimir Oleshchuk, Secure Interworking between P2PSIP and IMS, 2010 International Symposium on Collaborative Technologies and Systems (CTS 2010). pp.481-488, IEEE, 2010: Chicago, USA.*

10. *Xianghan Zheng, Vladimir Oleshchuk, Trust Enhancement of P2PSIP Communication Systems, International Journal for Internet Technology and Secured Transactions (IJITST), Vol. 4, Inderscience, 2010.*

# Reference

[1] *Android.com - Android at Google I/O*, pp. http://www.android.com/ (accessed July 2010).

[2] *Apache Derby*, pp. http://db.apache.org/derby/ (accessed July 2010).

[3] *Apple*, pp. http://www.apple.com/ (accessed Aug 2010).

[4] *BitComet - A free C++ BitTorrent/HTTP/FTP Download Client*, pp. http://www.bitcomet.com/ (accessed June 2010).

[5] *BitTorrent*, pp. http://www.bittorrent.com.

[6] *Business Communications Solutions from Avaya.*, pp. http://www.avaya.com/ (accessed July 2010).

[7] *Cisco Systems Inc*, pp. http://www.cisco.com/ (accessed July 2010).

[8] *CounterPath Corporation | Home*, pp. http://www.counterpath.com (accessed June 2010).

[9] *eBay Inc. Corporate Fact Sheet: Q3 2009*, pp. http://ebayinkblog.com/wp-content/uploads/2009/10/FINAL-eBay-Inc-Fact-Sheet-Q309.pdf (accessed July 2010).

[10] *Ericsson Service Development Studio (SDS) – 4.1*, pp. http://www.ericsson.com/developer/sub/open/technologies/ims_poc/tools/sds_41 (accessed July).

[11] *Ericsson White Paper - Ericsson Service Development Studio (SDS) 4.1 Technical Description*, Feb, 2009, pp. www.ericsson.com (accessed July).

[12] *Internet Explorer 8 Windows 7 Security Features Malware Privacy*, pp. http://www.microsoft.com/windows/internet-explorer/default.aspx (accessed July 2010).

[13] *Mobile Phones | Choose the Best Phone for You | Windows Mobile*, pp. http://www.microsoft.com/windowsmobile/en-my/default.mspx (accessed June 2010).

[14] *Napster - Wikipedia, the free encyclopedia*, pp. http://en.wikipedia.org/wiki/Napster (accessed July 2010).

[15] *Nokia - Nokia on the Web*, pp. http://www.nokia.com/ (accessed July 2010).

[16] *Openwave | Mobile Internet, Messaging, Mobile Analytics, Video Optimization*, pp. http://www.openwave.com/ (accessed June 2010).

[17] *Opera Mini && Opera Mobile browsers*, pp. http://www.opera.com/mobile/ (accessed July 2010).

[18] *P2PSIP*, pp. http://www.p2psip.org (accessed July 2010).

[19] *Peer-to-Peer*, pp. http://www1.cs.columbia.edu/~salman/peer/ (accessed July 2010).

[20] *SETI@home*, pp. http://setiathome.berkeley.edu/ (accessed July 2010).

[21] *The SIP Center - A portal for the commercial development of SIP Session Initiation Protocol*, pp. http://www.sipcenter.com (accessed July 2010).

[22] *Skype official website - download Skype free now for free calls and internet calls*, pp. http://www.skype.com/intl/en/ (accessed June 2010).

[23] *Trojan Horse - Wikipedia, the free encyclopedia*, pp. http://en.wikipedia.org/wiki/Trojan_Horse (accessed July 2010).

[24] *Twinkle - SIP softphone for Linux*, pp. http://www.twinklephone.com.

[25] *Usenet*, pp. http://en.wikipedia.org/wiki/Usenet (accessed July).

[26] *Welcome to Nortel*, pp. http://www.nortel.com (accessed July 2010).

[27] *Welcome to the UPnP Forum !*, pp. http://www.upnp.org (accessed July 2010).

[28] *Windows Live Messenger*, pp. http://explore.live.com/windows-live-messenger?os=winxp (accessed Aug 2010).

[29] *Wireshark . Go deep.*, pp. http://www.wireshark.org/ (accessed July 2010).

[30] *XHTML™ 1.0 The Extensible HyperText Markup Language (Second Edition)*, pp. http://www.w3.org/TR/2002/REC-xhtml1-20020801 (accessed July 2010).

[31] *Yahoo! Messenger - Chat, Instant message, SMS, Video Call, PC Calls*, pp. http://messenger.yahoo.com/ (accessed June 2010).

[32] E. Adar and B. Huberman, *Free Riding on Gnutella*, First Monday, 5 (Oct, 2000).

[33] S. A. Baset and H. Schulzrinne, *An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol*, *Proceedings of 25th Conference on Computer Communications (INFOCOM'06)*, IEEE, Barcelona, Spain, April, 2006.

[34] B. Bhattacharjee, S. Chawathe, V. Gopalakrishnan, P. Keleher and B. Silaghi, *Efficient Peer-To-Peer Searches Using Result-Caching*, Peer-to-Peer Systems II 2735 (Oct, 2003), pp. 225-236.

[35] D. A. Bryan, B. Lowekamp and C. Jennings, *dSIP: A P2P Approach to SIP Registration and Resource Location*, *IETF Internet Draft (draft-bryan-p2psip-dsip-00)*, Feb, 2007, pp. http://tools.ietf.org/html/draft-bryan-p2psip-dsip-00 (accessed June 2010).

[36] D. A. Bryan, B. B. Lowekamp and C. Jennings, *SOSIMPLE: A Serverless, Standards-based, P2P SIP Communication System First International Workshop on Advanced Architectures and Algorithms for Internet Delivery and Applications (AAA-IDEA'05)* IEEE, Orlando, USA, 2005, pp. pp. 42-49.

[37] D. A. Bryan, B. B. Lowekamp and M. Zangrilli, *The Design of a versatile, secure P2PSIP communications architecture for the public internet*, *IEEE international Symposium on Parallel and Distributed Processing*, IEEE, Miami, USA, April, 2008, pp. 1-8.

[38] D. A. Bryan, P. Matthews, E. Shim, D. Willis and S. Dawkins, *Concepts and Terminology for Peer to Peer SIP*, IETF Internet Draft (draft-ietf-p2psip-concepts-02) (July, 2008), pp. http://www.p2psip.org/drafts/draft-ietf-p2psip-concepts-02.html (accessed July 2010).

[39] J. Byers, J. Considine and M. Mitzenmacher, *Simple Load Balancing for Distributed Hash Tables*, Peer-To-Peer Systems, 2735 (2003), pp. 80-87.

[40] G. Camarillo, P. Nikander, J. Hautakorpi, A. Keranen and A. Johnston, *HIP BONE: Host Identity Protocol (HIP) Based Overlay Networking Environment*, IETF Internet Draft (draft-ietf-hip-bone-07) (June, 2010).

[41] F. Cao, D. A. Bryan and B. B. Lowekamp, *Providing Secure Services in Peer-to-Peer Communications Networks with Central Security Servers*, *International Conference on Internet and Web Applications and Services (ICIW)*, IEEE, Guadeloupe, Frech Caribbean, Feb, 2006, pp. 105-110.

[42] I. Clarke, O. Sandberg and B. Wiley, *Freenet: A Distributed Anonymous Information Storage and Retrieval System*, Designing Privacy Enhancing Technologies, 2009 (2001), pp. 46-66.

[43] F. Dabek, *A Distributed Hash Table*, *PHD Thesis of Massachusetts Institue of Technoloty*, Sep, 2005, pp. http://pdos.csail.mit.edu/papers/fdabek-phd-thesis.pdf (accessed July 2010).

[44] F. Dabek, J. Li, E. Sit, J. Robertson, M. F. Kaashoek and R. Morris, *Designing a DHT for low latency and high throughput*, *Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation*, USENIX Associationi, Berkelay, USA, 2004, pp. 7-7.

[45] V. Darlagiannis, N. Liebau, O. Heckmann, A. Mauthe and R. Steinmetz, *Cacheing Indices for Efficient Lookup in Structured Overlay Networks*, Agents and Peer-to-Peer Computing 4118 (2006), pp. 81-93.

[46] J. R. Douceur, *The Sybil Attack*, Peer-To-Peer Systems, 2429 (2002).

[47] P. Eronen and H. Tschofenig, *Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)*, IETF RFC 4279 (Dec, 2005).

[48] A. Fessi, H. Niedermayer, H. Kinkelin and G. Carle, *A cooperative SIP Infrastructure for Highly Reliable Telecommunication Services*, *Proceedings of the 1st international conference on Principles, systems and applications of IP telecommunications (IPTCOMM'07)*, ACM, New York, USA, July, 2007, pp. 29-38.

[49] P. Ganesan and G. S. Manku, *Optimal routing in Chord*, *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms* ACM, New Orleans, USA, 2004, pp. 176-185.

[50] J. Garrett, *Ajax: A New Approach to Web Applications.* , pp. www.adaptivepath.com/publications/essays/archives/000385.php (accessed July 2010).

[51] G. Ge and E. J. Whitehead, *Bamboo: an architecture modeling and code generation framework for configuration management systems*, *Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering*, ACM, Long Beach, CA, USA, 2005.

[52] D. Greenfield, *SIP Goes Peer-to-Peer*, *New Telephony Magazine (now IT Architect)*, Charlotte Wolfer, 1 May, 2005.

[53] M. Handley, V. Jacobson and C. Perkins, *SDP: Session Description Protocol*, *IETF RFC 4566*, July, 2006.

[54] M. Handley, H. Schulzrinne, E. Schooler and J. Rosenberg, *SIP:Session Initiation Protocol*, *IETF RFC 2543*, March, 1999.

[55] J. Hautakorpi and G. Camarillo, *Evaluation of DHTs from the viewpoint of interpersonal communications*, *Proceedings of the 6th international conference on Mobile and ubiquitous multimedia*, ACM, Oulu, Finland, 2007, pp. 74-83.

[56] J. Hautakorpi, A. Salinas, E. Harjula and M. Ylianttila, *Interconnecting P2PSIP and IMS*, *Proceedings of the Second International Conference on Next Generation Mobile Applications, Services and Technologies*, Wales, UK, Sept, 2008, pp. 83-88.

[57] F. Hong, M. Li, J. Yu and Y. Wang, *PChord: Improvement on Chord to Achieve Better Routing Efficiency by Exploiting Proximity*, *Proceedings of 25th IEEE International Conference on Distributed Computing Systems Workshops (ICDCSW'05)*, IEEE, Columbus, USA, June, 2006, pp. 806-811.

[58] C. Jampathom, *P2PSIP Security*, *Master Thesis of Helsinki University of Technology*, 2008, pp. Http://nordsecmob.tkk.fi/Thesisworks/Cheevarat-final.pdf (accessed July 2010).

[59] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset and H. Schulzrinne, *REsource LOcation And Discovery (RELOAD) base Protocol*, IETF Internet Draft (draft-ietf-p2psip-base-09) (July, 2010), pp. http://tools.ietf.org/html/draft-ietf-p2psip-reload-00 (accessed June 2010).

[60] J. Jiang, R. Pan, C. Liang and W. Wang, *Bi-Chord: An IMproved Approach for Lookup Routing in Chord*, *Advances in Databases and Information Systems*, Springer, 2005, pp. 338-348.

[61] J. Jiang, F. Tang, F. Pan and W. Wang, *Using bidirectional links to improve peer-to-peer lookup performance*, Journal of ZheJiang University, 7 (2006), pp. 945-951.

[62] X. Jiang, H. Zheng, C. Macian and V. Pascual, *Service Extensible P2P Peer Protocol*, IETF Internet Draft (draft-jiang-p2psip-sep-01) (Feb, 2008), pp. http://tools.ietf.org/html/draft-jiang-p2psip-sep-01 (accessed July 2010).

[63] A. Josang, *An Algebra for Assessing Trust in Certification Chains*, *Proceedings of the Network and Distributed Systems Security (NDSS'99)*, The Internet Society, 1999.

[64] A. Josang, *A LOGIC FOR UNCERTAIN PROBABILITIES*, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 9 (June, 2001), pp. 279-311.

[65] A. Josang, R. Hayward and S. Pope, *Trust network analysis with subjective logic*, *Proceedings of the 29th Australasian Computer Science Conference*, Australian Computer Society, Inc., Hobart, Australia, 2006, pp. 85-96.

[66] D. Karger, E. Lehman, T. Leighton, R. Panigrahy, M. Levine and D. Lewin, *Consistent hashing and random trees: distributed caching protocols for relieving hot spots on the World Wide Web*, *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, ACM, El Paso, Texas, United States, 1997.

[67] M. Kinateder, R. Terdic and K. Rothermel, *Strong pseudonymous communication for peer-to-peer reputation systems*, *Proceedings of the 2005 ACM symposium on Applied computing*, ACM, Santa Fe, New Mexico, 2005, pp. 1570-1576.

[68] J. Koskela, *A HIP-based peer-to-peer communication system*, *Proceedings of International Conference on Telecommunications (ICT 2008)*, St. Petersburg, Russia, 2008, pp. 1-7.

[69] S. Krishnamurthy, S. El-Ansary, E. Aurell and S. Haridi, *A Statistical Theory of Chord Under Churn in Peer-to-Peer Systems IV*, Peer-To-Peer, 3640 (2005), pp. 93-103.

[70] L. Le and G.-S. Kuo, *Hierarchical and Breathing Peer-to-Peer SIP System*, *IEEE International Conference on Communcations (ICC'07)*, IEEE, Glasgow, Scotland, 2007, pp. 1887-1892.

[71] B. Leong, B. Liskov and E. D. Demaine, *EpiChord: Parallelizing the Chord lookup algorithm with reactive routing state management*, Computer Communications, 29 (May 2006), pp. 1242-1259.

[72] J. Li, J. Stribling, T. M. Gil, R. Morris and M. F. Kaashoek, *Comparing the Performance of Distributed Hash Tables Under Churn*, Peer-To-Peer Systems, 3279 (2005), pp. 87-99.

[73] J. Maenpaa and G. Camarillo, *Service Discovery Usage for REsource LOcation And Discovery (RELOAD)*, IETF Internet Draft (draft-ietf-p2psip-service-discovery-01) (July, 2010), pp. https://datatracker.ietf.org/doc/draft-ietf-p2psip-service-discovery/ (accessed July 2010).

[74] I. Martinez-Yelmo, A. Bikfalvi, R. Cuevas, C. Guerrero and J. Garcia, *H-P2PSIP: Interconnection of P2PSIP domain for global multimedia services based on a hierarchical DHT overlay network*, Computer Networks, 53 (March, 2009), pp. 556-568.

[75] M. Matuszewski and E. Kokkonen, *Mobile P2PSIP - Peer-to-Peer SIP Communication in Mobile Communities*, *5th IEEE Consumer Communications and Networking Conference (CCNC)*, IEEE, Las Vegas, USA, Jan, 2008, pp. 1159-1165.

[76] P. Maymounkov and D. Mazieres, *Kademlia: A Peer-to-Peer Information System Based on the XOR Metric*, *Proceedings of 1st International Workshop on Peer-to-Peer Systems (IPTPS02)*, Springer, Cambridge, USA, 2002, pp. 53-65.

[77] V. A. Mesaros, B. Carton and P. V. Roy, *S-Chord: Using symmetry to improve lookup efficiency in Chord*, *Proceedings of International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'03)*, CSREA Press, Las Vegas, USA, 2003, pp. 1752-1760.

[78] J. Mirkovic, G. Prier and P. Reiher, *Attacking DDoS at the source*, *Proceedings of 10th IEEE International Conference on Network Protocols*, IEEE, Paris, France, 2002, pp. 312-321.

[79] R. Moskowitz and P. Nikander, *Host Identity Protocol (HIP) Architecture*, *IETF RFC 4423*, May, 2006.

[80] R. M. Needham, *Denial of service*, *Proceedings of the 1st ACM conference on Computer and Communication Security*, ACM, Fairfax, USA, 1993, pp. 151-153.

[81] V. Oleshchuk, *Trust-based Framework for Security Enhancement of Wireless Sensor Networks*, *4th IEEE Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS 2007)*, Dortmund, Germany 6-8 Sept, 2007, pp. 623-627.

[82] V. Oleshchuk and V. Zadorozhny, *Trust-Aware Query Processing in Data Intensive Sensor Networks*, *International Conference on Sensor Technologies and Applications (SensorComm)*, Valencia, Spain, 14-20 Oct, 2007, pp. 176-180.

[83] A. Ornaghi and M. Valleri, *Man in the middle attacks*, pp. http://www.blackhat.com/presentations/bh-usa-03/bh-us-03-ornaghi-valleri.pdf (accessed July 2010).

[84] J. Postel, *Internet Control Message Protocol*, IETF RFC 792 (1981).

[85] S. Ratnasamy, P. Francis, M. Handley, R. Karp and S. Schenker, *A scalable content-addressable network*, *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM'01)*, ACM, San Diego, USA, Oct, 2001, pp. 161-172.

[86] S. Ratnasamy, R. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan and S. Shenker, *GHT: a geographic hash table for data-centric storage*, *Proceedings of the 1st ACm international workshop on Wireless sensor networks and applications*, ACM, Atlanta, USA, 2002, pp. 78-87.

[87] S. Rhea and D. Geels, *Handling churn in a DHT*, *Proceedings of the annual conference on USENIX Annual Technical Conference*, USENIX Association, Boston, USA, 2004, pp. 10-10.

[88] A. B. Roach, *Session Initiation Protocol - Specific Event Notification*, *IETF RFC 3265*, June, 2002.

[89] J. Rosenberg, *Interactive Connectivity Establishment: NAT Traversal for the Session Initiation Protocol*, IETF Journal, Volume 2 (Nov, 2006), pp. 14-19.

[90] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley and E. Schooler, *SIP:Session Initiation Protocol*, *IETF RFC 3261*, June, 2002.

[91] J. Rosenberg, H. Schulzrinne, C. Huitema and D. Gurle, *Session Initiation Protocol (SIP) Extension for Instant Messaging*, *IETF RFC 3428*, Dec, 2002.

[92] J. Rosenberg, J. Weinberger, C. Huitema and R. Mahy, *STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs)*, IETF RFC 3489 (March 2003).

[93] A. Rowstron and P. Druschel, *Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems*, *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg*, Springer, Heidelberg, Germany, 2001, pp. 329-350.

[94] R. Schlegel, S. Niccolini, S. Tartarelli and M. Brunner, *SPam over Internet Telephony (SPIT)*, *Global Telecommunications Conference (GLOBECOM 06)*, IEEE, San Francisco, USA, 2006, pp. 1-6.

[95] J. Seedorf, *Lawful Interception in P2P-Based VoIP Systems*, Principles, Systems and Applications of IP Telecommunications, 5310 (2008), pp. 217-235.

[96] J. Seedorf, *Security Challenges for P2P-SIP*, IEEE Network Special Issue on Securing Voice over IP, 20 (2006), pp. 38-45.

[97] J. Seedorf, F. Ruwolt, M. Stiemerling and S. Niccolini, *Evaluating P2PSIP under Attack: An Emulative Study*, *Proceedings of 2008 Global Telecommunications Conference (Globecom)*, IEEE, New Orleans, USA, 2008, pp. 1-6.

[98] D. K. Sepandar, T. S. Mario and G.-M. Hector, *The Eigentrust algorithm for reputation management in P2P networks*, *Proceedings of the 12th international conference on World Wide Web*, ACM, Budapest, Hungary, 2003, pp. 640-651.

[99] G. Shi, Y. Long, J. Chen, H. Gong and H. Zhang, *T2MC: A Peer-to-Peer Mismatch Reduction Technique by Traceroute and 2-Means Classification Algorithm*, Networking 2008Ad Hoc and Sensor Networks, Wireless Networks, Next Generation Internet, 4982 (2009), pp. 366-374.

[100] A. Singh, T. W. Ngan, P. Druschel and D. S. Wallach, *Eclipse attacks on overlay networks Threats and defenses*, *Proceedings of 25th IEEE International Conference on Computer Communications (INFOCOM 06)*, IEEE, Barcelona, Spain, 2006, pp. 1-12.

[101] K. Singh and H. Schulzrinne, *Peer-to-peer internet telephony using SIP*, *Proceedings of the international workshop on Network and operating systems support for digital audio and video*, ACM, Stevenson, Washington, USA, 2005, pp. 63-68.

[102] K. Singh and H. Schulzrinne, *SIPpeer: A Session Initiation Protocol (SIP) based Peer-to-Peer Internet Telephony Client Adapt*, Jan, 2005, pp. http://ww1.cs.columbia.edu/kns10/publication/sip-p2p-design.pdf (accessed June 2010).

[103] J. L. Snell, *Expected Value and Variance*, *INTRODUCTION TO PROBABILITY*, McGraw-Hill, pp. 210-230.

[104] H. Song, X. Jiang, R. Even and D. A. Bryan, *P2PSIP Overlay Diagnostics*, IETF Internet Draft (draft-ietf-p2psip-diagnostics-04) (July, 2010), pp. http://tools.ietf.org/html/draft-ietf-p2psip-diagnostics-04 (accessed July).

[105] H. Song, X. Jiang, H. Zheng and H. Deng, *P2PSIP Client Protocol*, IETF Internet Draft (draft-zheng-p2psip-client-protocol-01) (Aug, 2008), pp. http://tools.ietf.org/html/draft-zheng-p2psip-client-protocol-01 (accessed July 2010).

[106] S. Song, K. Hwang, R. Zhou and Y.-K. Kwok, *Trusted P2P Transactions with Fuzzy Reputation Aggregation*, IEEE Internet Computing 9(2005), pp. 24-34.

[107] E. H. Spafford, *The internet worm program: an analysis*, SIGCOMM Comput. Commun. Rev., 19 (1989), pp. 17-57.

[108] R. Sparks, *SIP:basics and beyond*, Queue, 5 (2007), pp. 22-33.

[109] I. Stoica, Morris, R., Liben-Nowell, D., Karger, D. R., Kaashoek, M. F., Dabek, F., Balakrishnan, H., *Chord: a scalable peer-to-peer lookup protocol for internet applications*, IEEE/ACM Transactions on Networking, 11 (2003), pp. 17-32.

[110] M. Stukas and D. C. Sicker, *An Evaluation of VoIP Traversal of Firewalls and NATs within an Enterprise Environment* Information Systems Frontiers, 6 (Sept, 2004), pp. 219-228.

[111] M. Swinburn and A. Jayawardena, *Cooperative Trust Model for a Decentralised Peer-to-Peer E-Market*, Innovations and Advanced Technologies in Systems, Computing Sciences and Software Engineering (2008), pp. 330-335.

[112] A. Vilei, G. Convertino and F. Crudo, *A new UPnP architecture for distributed video voice over IP*, *Proceedings of the 5th international conference on Mobile and ubiquitous multimedia*, ACM, Stanford, California, 2006.

[113] N. Wakamiya and M. Murata, *Overlay network symbiosis: evolution and cooperation*, *Proceedings of the 1st international conference on Bio inspired models of network, information and computing systems*, ACM, Cavalese, Italy, 2006.

[114] D. S. Wallach, *A survey of peer-to-peer security issues*, *Proceedings of the 2002 Next-NSF-JSPS international conference on Software security: theories and systems*, Springer, Tokyo, Japan, 2002, pp. 42-57.

[115] C. Wolfer, *Peer-to-Peer VoIP : Will it Ever Work New Telephony Magazine (Now IT Architect)*, Charlotte Wolfer, 24 May, 2005.

[116] M. Zangrilli and D. A. Bryan, *A Chord-based DHT for Resource Lookup in P2PSIP*, *IETF Internet Draft (draft-zangrilli-p2psip-dsip-dhtchord-00)*, Feb, 2007, pp. http://www.p2psip.org/drafts/draft-zangrilli-p2psip-dsip-dhtchord-00.html (accessed July 2010).

[117] B. Y. Zhao, J. Kubiatowicz and A. Joseph, *Tapestry: a fault-tolerant wide-area application infrastructure*, SIGCOMM Computer Communication Review, 32 (2002), pp. 81-81.

[118] H. Zheng and X. Jiang, *P2PSIP Client Protocol*, IETF Internet Draft (draft-zheng-p2psip-client-protocol-00) (Dec, 2007), pp. http://tools.ietf.org/html/draft-zheng-p2psip-client-protocol-00 (accessed July 2010).

[119] X. Zheng and V. Oleshchuk, *The Design of Secure and Efficient P2PSIP Communication Systems*, *Information Security Theory and Practices. Security and Privacy of Pervasive Systems and Smart Devices*, Springer, 2010, pp. 253-260.

[120] X. Zheng and V. Oleshchuk, *Improvement of Chord overlay for P2PSIP-based Communication Systems*, International Journal of Computer Networks & Communications (IJCNC), 1 (Oct, 2009), pp. 133-142.

[121] X. Zheng and V. Oleshchuk, *Providing Privacy Service for P2PSIP based Communication Systems*, *Proceedings of Norst Informasjonssikkerhetkonferanse (NISK)*, Kristiansand, Norway, 2008.

[122] X. Zheng and V. Oleshchuk, *A Secure Architecture for P2PSIP-based Communication Systems*, *Proceedings of the 2nd international conference on Security of information and networks*, ACM, Famagusta, North Cyprus, Oct, 2009, pp. 75-82.

[123] X. Zheng and V. Oleshchuk, *A survey on peer-to-peer SIP based communication systems*, *Peer-to-Peer Networking and Applications*, Springer, Jan, 2010.

[124] X. Zheng and V. Oleshchuk, *Trust-based Framework for Security Enhancement of P2PSIP Communication Systems*, *The 4th International Conference for Internet Technology and Secured Transactions (ICITST-2009)*, IEEE, London, UK, Nov, 2009, pp. 1-6.

[125] X. Zheng, V. Oleshchuk and H. Jiao, *A System Architecture for SIP/IMS-based Multimedia Services Novel Algorithms and Techniques In Telecommunications, Automation and Industrial Electronics*, Springer, Dec, 2007, pp. 543-548.

[126] S. Zoels, Z. Despotovic and W. Kellerer, *On hierarchical DHT systems - An analytical approach for optimal designs*, Computer Communications, 31 (Feb, 2008), pp. 576-590.