UNIVERSITAT POLITÈCNICA DE CATALUNYA

DEPARTAMENT D'ENGINYERIA TELEMÀTICA

Tesi Doctoral

Security in Peer-to-Peer Communication Systems

P2PSIP Access Control Protection

Doctorand:
Diego Suárez Touceda

Directors:
Dr. José María Sierra Cámara
Dr. Miguel Soriano Ibáñez

Barcelona, Juny 2011

Tesi Doctoral

Security in Peer-to-Peer Communication Systems
P2PSIP Access Control Protection

Doctorand: **Diego Suárez Touceda**

Directors: **Dr. José María Sierra Cámara**
Dr. Miguel Soriano Ibáñez

Membres del tribunal qualificador:

Signatura

President:

Vocal 1:

Vocal 2:

Vocal 3:

Secretari:

Qualificació:

Barcelona, de de

.

A miña familia e amigos, por facer que a música nunca remate...

Those who dance are considered insane
by those who can't hear the music.

George Carlin

ABSTRACT

P2PSIP (Peer-to-Peer Session Initiation Protocol) is a protocol developed by the IETF (Internet Engineering Task Force) for the establishment, completion and modification of communication sessions that emerges as a complement to SIP (Session Initiation Protocol) in environments where the original SIP protocol may fail for technical, financial, security, or social reasons. In order to do so, P2PSIP systems replace all the architecture of servers of the original SIP systems used for the registration and location of users, by a structured P2P network that distributes these functions among all the user agents that are part of the system. This new architecture, as with any emerging system, presents a completely new security problematic which analysis, subject of this thesis, is of crucial importance for its secure development and future standardization.

Starting with a study of the state of the art in network security and continuing with more specific systems such as SIP and P2P, we identify the most important security services within the architecture of a P2PSIP communication system: access control, bootstrap, routing, storage and communication. Once the security services have been identified, we conduct an analysis of the attacks that can affect each of them, as well as a study of the existing countermeasures that can be used to prevent or mitigate these attacks. Based on the presented attacks and the weaknesses found in the existing measures to prevent them, we design specific solutions to improve the security of P2PSIP communication systems. To this end, we focus on the service that stands as the cornerstone of P2PSIP communication systems' security: access control. Among the new designed solutions stand out: a certification model based on the segregation of the identity of users and nodes, a model for secure access control for on-the-fly P2PSIP systems and an authorization framework for P2PSIP systems built on the recently published Internet Attribute Certificate Profile for Authorization.

Finally, based on the existing measures and the new solutions designed, we define a set of security recommendations that should be considered for the design, implementation and

maintenance of P2PSIP communication systems.

Keywords: *P2PSIP, P2P, VoIP, DHT, Security, Access Control, Authentication, Authorization, Attribute Certificates.*

Preface

> The inferno of the living is not something that will be; if there is one, it is what is already here, the inferno where we live every day, that we form by being together. There are two ways to escape suffering it. The first is easy for many: accept the inferno and become such a part of it that you can no longer see it. The second is risky and demands constant vigilance and apprehension: seek and learn to recognize who and what, in the midst of the inferno, are not inferno, then make them endure, give them space.
>
> Italo Calvino - Invisible Cities

In deference to my ancestors, I would like to write these lines in my grandparents language, my mother tongue, Galician.

Moito tempo vai xa dende que comezou esta aventura, mais aínda recordo aquel día como se fora hoxe. Ibamos camiño de Ortigueira co sorriso na boca, ledos coma nenos, sabendo que nos esperaban varios días de festa. Eu conducía o supercinco, aquela máquina máxica. De súpeto, sonou o teléfono. Non o meu, ese xa sonara varias veces antes, pero eu no me decatara. A chamada era urxente, "para o coche e chama a teus pais". Así foi, voteime a un lado, parei o bólido e chamei. Meu pai non daba crédito, ¿ cómo era posible que levasen tanto tempo chamándome ó móvil e eu non o collera ? Ás veces, a él tamén lle molesta que escoite a música, esa que nos fai salir da liña, esa que nos fai bailar. José María Sierra quería falar conmigo. Era da universidade, a UC3M de Madrid. Eu non sabía quen era, non coñecía a ningén que se chamase así. Iso non é raro, acostumo a olvidar o nome da xente con bastante facilidade, alarmante ás veces. De feito, lévame entre cinco e dez segundos olvidar o nome de alguén que me acaban de presentar. Nin que dicir ten o que pasa cando me presentan a cinco persoas á vez. Eu intento concentrarme nunha sola, como cas mulleres, monogamia e cabezonería. A ver se a base de insistir un logra o seu obxetivo. Non soe funcionar. Como di un bo amigo meu, un olvida os nomes para deixarlle oco nos recordos ás experiencias. O mesmo pasa cas mulleres,

cando unha muller se vai e para facerlle sitio a unha mellor que está por chegar.

Pero neste caso, estaba case seguro de que non o coñecía. Ademáis, facía tempo que Madrid formaba parte do pasado, unha gran fase da miña vida, pero xa rematada. Fixen un último intento por recordar. ¿ Da UC3M ?, igual me confundían co meu irmán, él estudara nesa universidade. Eu solo estivera alí unha vez, na súa cerimonia de graduación. ¿ Marcharía sen pagar un café ? Nada tiña sentido, e iso gustábame.

O caso é que chamei, e as cousas comezaron a cobrar sentido. A primeiros de ese ano, miña nai me chamara un día para informarme sobre unha oferta de traballo que vira. A oferta non me interesaba moito, pero ó lado había unha referencia á convocatoria de becas FPI para realizar o doutorado. Eso si me interesou, así que enviei a solicitude para unha beca FPI na UPC, nun proxecto dirixido por Miguel Soriano. Según souben durante esa primeira chamada, el xa tiña as prazas cubertas, pero envioulle o meus datos o seu colega Jose María Sierra, que tiña unha vacante. Varias chamadas telefónicas despóis, acordamos reunirnos en Madrid a finales da seguinte semán, un tempo máis que necesario para que o meu corpo, despois do festival de Ortigueira, se adaptase de novo a realidade.

Foi algo increíble. Eu tiña moi claro que quería facelo doutorado, pero a miña situación daquela no lle preveía moito futuro a esa posibilidade. De feito, a realidade era que todo apuntaba a que ía ter que renunciar a elo para facer algo máis socialmente aceptado. Pero nun segundo, cunha chamada de télefono, todo cambiou. Algúns dirán que foi sorte, outros que foi casualidade. Pero o certo é que simplemente sonou a música dunha un gaita galega, e eu comecei a bailar...

Moitas cousas ocorreron antes disto, e moitas outras despóis. Repasalas todas sería un placer, pero tamén unha tarea intratable. Sen embargo, algunhas persoas, experiencias e lugares merecen unha mención especial, que non debo nin quero pasar por alto. A eles, pola súa importancia na miña vida e por extensión nesta tese, lles quero dedicar unhas verbas de agradecemento.

Os meus avós, ós que aínda están e ós que xa non, porque eles son os responsables de estas dúas maravillosas familias das que son parte, os Suárez e os Touceda. E especialmente a miña "abuela" Carmen, por tantos anos de "ese amor de la abuela", unha fonte inagotable de cariño, sacrificio e comprensión.

O meus pais, porque sen eles nada sería posible. Gracias por soportar a miña continúa irreverencia, despotismo e falta de gratitude. Gracias polo voso apoio e amor incondicional. Gracias, en definitiva, por facer de min unha persoa feliz.

A Lupo, o meu irmán preferido, por tantos anos e tantas experiencias xuntos. Tamén á cuñi.

Os meus irmáns da estrada, onde o Candil de Silvia ilumina a nosa amizade: Simal, Tonecho, Pernas, Machino, Mou, Goldar, Unai, Tinto, Javichu, Sito, Richi, Chucurú, Corbatero, Julio, Moncho, Uzal, Zyppy, Carlitos, Tallón, Roi, Orella, Carbia, Mimelo, Gañete, Marque, Manolito, Collazo, Dico ... Porque somos grandes, moi grandes!!!! Tamén as miñas irmás, Lara, Desi, Marora, ... e a outros moitos da zona, especialmente a Picholas e María que estaban no coche conmigo cando todo comezou, porque aínda que a nosa vida non se cruce moi asiduamente, pasamos moitos momentos memorables xuntos.

Ós afortunados que pasaron e pasarán tantos veráns conmigo, porque durante uns meses vivimos no paraíso: Martiño, Marta, Angel, Bea, Juampa, Ethel, Noelia, Tonazo, Chaito, Jano, Dieguito, Obispo, María, Anxo, Raquel, Gersom, Silvia ...

Ós amigos de Madrid, por facer de esta marabillosa e á vez monstruosa cidade, un lugar habitable: Cormac, Oli, Roi, Mariam, Andrés, Guille, Omar, Yoli, Pili, Tegra, Paloma, Julio, Lorena, Juanjo, ...

Os meus compañeiros de Erasmus, porque alí se abreu a miña pequena ventana ó mundo: Flo, Rosveisa, Vir Short, Vir Long, Bill, Leo, e tantos outros.

Ó interrail e os meus dous compañeiros de viaxe, por tantas experiencias. Temos que repetir esa viaxe!!!!!

A Mexico, porque aínda non desfixera ás maletas e xa me sentía coma na casa. Gracias a toda a xente que coñecín alí, especialmente a Jorge, Julián, Ivan, Paulino e Kiwi.

A India, a Nepal, a Sudamérica e a súa xente por abrirme os ollos a outra realidade... e os incautos que me acompañaron na viaxe, por una experiencia inigualable.

A San Vicente, Sabucedo e Ortigueira, porque existen sitios onde un pode sentir a maxia...

A Carlos Mex, pola súa hospitalidade e paciencia durante a miña estancia no ITESM.

A Miguel Soriano, polo seu papel crucial tanto no comenzo coma no desenlace desta aventura.

A José María, porque aquel descoñecido que unha vez me chamou por teléfono, xa é moito máis que un xefe ou un director de tese, é un amigo. E a todos aqueles que me acompañaron e ós que eu acompañei nesta viaxe por augas turbulentas: Joaquín, Antonio, Miguel, Fernando, Oscar I, Mildrey, Pablo, Nacho, Oscar II, Fidel, ... *we shall overcome some day...*

Gracias, en xeral, a todos os que me sorriron ou me fixeron sorrir algunha vez.

Entonces duerme, y si te sientes mejor, quiero que sueñes conmigo, que sueñes conmigo...

CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY OF TERMS

AA	Attribute Authority
AC	Attribute Certificates
ACK	ACKnowledges
ACL	Access Control List
AES	Advanced Encryption Standard
ARPANET	Advanced Research Projects Agency NETwork
CA	Certificate Authority
CAPTCHA	Completely Automated Public Turing test to tell Computers and Humans Apart
CCN	Centro Criptológico Nacional
CIA	Confidentiality, Integrity, Availability
CPU	Central Processing Unit
CRL	Certificate Revocation List
DAC	Discretionary Access Control
DCA	Distributed Certificate Authority
DDACL	Dynamic Distributed Access Control List
DDNS	Dynamic Domain Name System
DDoS	Distributed Denial of Service
DHT	Distributed Hash Table
DNS	Domain Name System
DoS	Denial of Service
DS	Drop Strategies
DTLS	Datagram transport Layer Security
ENUM	TelephonE NUMber mapping

H.225	Call signaling protocols and media stream packetization for packet-based multimedia communication systems
H.245	Control protocol for multimedia communication
H.323	Packet-based multimedia communications systems
HTTP	Hypertext Transfer Protocol
IAS	Incoming Allocation Strategies
IAX	Inter-Asterisk-eXchange
ID	IDentifier
IETF	Internet Engineering Task Force
IM	Internet Messaging
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
IPSec	Internet Protocol Security
IRC	Internet Relay Chat
ISP	Internet Service Provider
ITU	International Telecommunication Union
kind-ID	Data Kind ID
LAN	Local Area Networks
MAC	Mandatory Access Control
NAT	Network Address Translation
NIST	National Institute of Standards and Technology
node-ID	Node IDentifier
NRM	Network Rating Model
NSA	National Security Agency
NVP	Network Voice Protocol
OSI	Open System Interconnection
P2P	Peer-to-Peer
P2PSIP	Peer-to-Peer Session Initiation Protocol
P2PSL	Peer-to-Peer Security Layer
PBX	Private Branch eXchange
PC	Personal Computer
PGP	Pretty Good Privacy
PK	Public Key
PKC	Public Key Certificate
PKI	Public Key Infrastructure

PMI	Privilege Management Infrastructure
PrK	Private Key
PSTN	Public Switched Telephone Network
QoS	Quality of Service
RELOAD	REsource LOcation And Discovery
resource-ID	Resource IDentifier
RBAC	Role-Based Access Control
RFC	Request For Comments
RTCP	Real-Time Control Protocol
RTP	Real-time Transport Protocol
S/MIME	Secure / Multipurpose Internet Mail Extensions
SCTP	Stream Control Transmission Protocol
SDES	SDP Security DEScriptions for Media Streams
SDP	Session Description Protocol
SHA-1	Secure Hash Algorithm
SIP	Session Initiation Protocol
SIP-URI	Session Initiation Protocol Uniform Resource Identifier
SIPS-URI	Session Initiation Protocol Secure Uniform Resource Identifier
SLP	Service Location Protocol
SOS	Secure Opinion Server
SPIM	SPam over Internet Messaging
SPIT	SPam over Internet Telephony
SPPP	SPam over Presence Protocol
SRTP	Secure RTP
STUN	Session Traversal Utilities for NAT
TCP	Transmission Control Protocol
TDES	Triple Data Encryption Standard
TELCO	TELecommunications Operator
TLS	Transport Layer Security
TLS-PSK	Pre-Shared Key ciphersuites for TLS
TLS-SRP	Secure Remote Password (SRP) Protocol for TLS
TTL	Time To Live
TTP	Trusted Third Party
UA	User Agent
UDP	User Datagram Protocol
VMS	Voice Mail Service

VoIP	Voice over IP
ZRTP	Media Path Key Agreement for Unicast Secure RTP
DTLS-SRTP	DTLS Extension to Establish Keys for the SRTP

CHAPTER 1

INTRODUCTION

> One never knows what one is going to do.
> One starts a painting and then it
> becomes something quite different.
>
> Pablo Picasso

1.1 Hypothesis

Over the years, communication systems have been taking an increasingly important role in our society. This evolution has been accompanied by more complex and powerful communication systems, and in which security systems have been gaining increasing importance. It can be pointed out as an inflection of the significance of these systems, the invention of the electric telegraph in the first half of the nineteenth century, which appearance marks the beginning of the modern telecommunications systems. Since that time, and with the emergence of its flagship device, the phone, there has been a real revolution in these systems, mainly due to the advent of computer communication networks (which most prominent representative is Internet), wireless technologies and mass communication systems such as radio and television. As noted, the evolution of these systems has been supported by the establishment of security improvements, either in specific telephony protocols or protocols that provide the basis for the transmission of information. Thus, the use of scramblers in conventional telephony (providing confidential communications) have given way to complete architectures to provide the required security services (confidentiality, integrity, authentication, authorization, ...); as those currently used in mobile communications.

Undoubtedly, one of the great advances in this field is the use of IP (Internet Protocol) [1] based computer networks as infrastructure for the establishment of communication services;

mainly messaging, telephony and videoconferencing. VoIP (Voice over Internet Protocol) systems have some advantages over the traditional telephone service, such as a lower cost of rate-setting and maintenance, a better utilization of the unused resources of the data network, improvements in functionality due to the integration of other services and increased mobility. In contrast, the security vulnerabilities associated with these networks from the beginning, sometimes have discouraged its use until the appropriate security mechanisms could be established to equate them to traditional telephone networks.

The origins of VoIP date back to 1973 with the development of the protocol NVP (Network Voice Protocol) [2] that was used experimentally to perform real-time voice communications in ARPANET (Advanced Research Projects Agency Network). Later, with the passage of time, several proposals have been appearing, existing currently three main alternatives: H.323, IAX and SIP.

H.323 (packet-based multimedia communications system) [3] is a specification of the ITU (International Telecommunication Union) for streaming audio and video over packet networks, comprising various standards such as H.225 (call signaling protocols and media stream packetization for packet-based multimedia communication systems) [4] and H.245 (control protocol for multimedia communication) [5].

IAX (Inter-Asterisk-eXchange) [6] is a protocol for the creation, modification and termination of media sessions over IP networks. Developed for Asterisk [7] (software application that provides functionalities of a PBX -Private Branch eXchange-), its main objective is to offer voice over IP, but it can also be used to establish other multimedia sessions such as videoconferences, TV broadcasting, etc.

Finally, and within the framework of this thesis, we focus on SIP (Session Initiation Protocol). SIP is an application level protocol, developed by the IETF (Internet Engineering Task Force) for the establishment, termination and modification of communication sessions, that can use either TCP (Transmission Control Protocol) [8], UDP (User Datagram Protocol) [9] or SCTP (Stream Control Transmission Protocol) [10] at the transport layer. This protocol is described in the RFC (Request For Comments) that defines it [11] and in posterior updates as [12] (remote notification of events), [13] (replacement of TDES -Triple Data Encryption Standard- [14] by AES -Advanced Encryption Standard- [15] as the default cryptographic protocol), [16] (improvements in SIP transactions) and [17] (improvements in the recognition of the identity of the participants in a SIP dialog). There is also a bibliography of RFCs that explain other features the protocol implements and that can be found in [18].

SIP is based on the classical client-server connectivity model. In this model, participants' roles are clearly defined and are not interchangeable: one or more servers offer a range of services used by a group of clients. Although this model is a valid solution in most scenarios,

there are environments in which the protocol is not feasible for technical, financial, social or security reasons. Some of these environments are [19]: small organizations without the technical resources to install their own server that do not want their internal communications to pass through external servers, limited or lack of connectivity, ad-hoc groups, government censorship, or high scalability requirements.

Based on the presented scenarios, it appears the idea of replacing the client-server architecture underlying SIP for a P2P (Peer-to-Peer) architecture. P2P networks are designed to take advantage of dispersed network resources and enable participants to act as servers or clients (without the need for a fixed role). Also, their main characteristic is the direct sharing of resources (CPU -Central Processing Unit- cycles, storage, content, connectivity, etc..) among users without the need for a central server that mediates among them. Based on this connectivity model, a multitude of applications have emerged, such as distributed computing systems as Seti@home [20] [21] or Gnome@home [22] [23], distributed database systems as PIER [24] or Piazza [25], or content distribution applications such as Napster [26], Gnutella [27] or Freenet [28]. In the case of communication systems, this new model replaces the entire server architecture of the original SIP protocol by a P2P network that distributes SIP functions among all the entities participating in the system. This model is called P2PSIP (Peer-to-Peer Session Initiation Protocol) [29].

The new architecture of P2PSIP communication systems presents a completely different security problematic in comparison with traditional SIP systems:

- Access control and users' authentication in SIP are granted by the system's infrastructure of servers. However, the decentralized nature of P2PSIP makes access control and users' authentication more challenging tasks.

- In SIP, users join the system by contacting a trusted server (Registrar) while in P2PSIP they must follow a bootstrapping process contacting other users (non trusted) of the system.

- Trusted SIP Proxies are the responsible for forwarding the users' requests in SIP, while in P2PSIP the users' messages are forwarded by a non trusted P2P overlay.

- In SIP, users' contact information and private resources (such as voicemails) are stored by trusted servers, while in P2PSIP they are stored by a non trusted P2P overlay.

- The final media communication process is established in both systems using the SIP protocol, being valid in same cases the existing SIP protection mechanism for the communication service. However, the introduction of a new P2P architecture in P2PSIP substantially change the security problematic in others.

In relation with existing P2P systems, despite that at first glance it may appear that requirements on P2PSIP services are no different than those for file-sharing services with an extra communication layer, P2PSIP presents completely new security issues, as described in [30, 31], that suggest a specific research:

- *Resource consumption*: In file-sharing systems each user indexes hundreds or thousands of files while in P2PSIP systems users only store a limited number of user's contact information and several voice-mails or offline messages.
- *Availability*: Resources in P2P file-sharing systems are files available at multiple locations. This abundance of resources overcame failures involving single instances and, therefore, attacks targeting a single resource need to be addressed to the distributed index in order to succeed. In turn, in P2PSIP systems resources are unique user's contact information and the attacks can be also directed to the user's device directly.
- *Integrity*: In P2P file-sharing systems attackers intend to corrupt files while in P2PSIP systems they try to impersonate users.
- *Confidentiality*: Shared files are usually readable by all users, while users' communications are usually meant to be confidential.
- *Bit-rate and Latency*: Realtime traffic requires a minimum constant bit-rate and low latency, while file-sharing tolerates unstable network conditions.
- *Peer lifetime*: In file-sharing systems users usually do not stay online very much longer than the time they need to get the file they are looking for. However, in P2PSIP systems, they usually stay online longer either to call or to be reachable by other users.

Motivated by the exposed above, a thorough study of the new security issues of P2PSIP communication systems is needed. In addition, we are at the right time to carry it out, because P2PSIP systems are still in development phase and our research could help positively to a better specification, in terms of security, of them.

1.2 State of the Art

The analysis of the state of the art of this thesis starts with an introduction to computer networks security within the security problematic of information systems. Then, it follows a detailed description of the security in SIP and P2P networks. Finally, we will finish our review of the state of the art analyzing the security of the existing proposals for Peer-to-Peer Communication Systems that have recently appeared.

1.2.1 Computer Networks Security

Information security is based on three pillars known by the acronym CIA (Confidentiality, Integrity, Availability) [32]: Confidentiality (prevention of the disclosure of information to unauthorized individuals or systems), Integrity (ability to keep information free from unauthorized modifications) and Availability (access to information by authorized users in a convenient format and at a reasonable time).

The methods used to ensure these properties have changed a lot over the years, as technology advanced. In the early days of computing, computers were huge, rare and very expensive. Because they were completely isolated, information security was limited to system's physical security: prevention of any damage the hardware might suffer and users' access control to the room the computer was located in. But times change. First came the first devices that allowed direct communication with the computer server from different parts of the building where it was, and later the modems, that through the standard telephone line, guaranteed user access not only from adjoining rooms but from cities located miles away.

The new opportunities offered by communication networks to access computers remotely, programs and data sharing, coupled with the technological advances; completely changed the computing landscape. The first computer networks came, and were used by corporations to automate and store online data about their customers, operations, etc. A phenomenon that quickly moved on to universities and colleges. Finally, with the appearance of PCs (Personal Computers), computers were becoming more common and accessible to the general public, and the interconnection among them resulted in the networks we know today (mainly Internet) and in which a user can perform almost any action she wants: to do online shopping, to access her bank details and transactions, to read the newspapers in its digital version, to listen to music, to watch TV, to make the tax return, etc.

This evolution of computers reflects a different security problematic than in its infancy. We started with a scenario in which computers were very rare, were isolated physically, engaged in some very specific tasks and were accessible to only a small group of experts. Now we are in a scenario in which computers are very common, are interconnected and can be accessed remotely from anywhere in the world; in which any user, regardless of her experience, has access to them and is able to handle them, and in which the stored data and the operations performed are the cornerstone of our society and whose modification with malicious purposes could have catastrophic consequences. Unfortunately, these facts have not gone unnoticed by criminals who have multiplied their illicit activity, as it can be seen in the incident reports prepared by the CERT/CC (Computer Emergency Readiness Team Coordination Center) [33]. With this scenario, it is not difficult to imagine how little by little the international community (governments, corporations, universities, etc.) have been increasing interest in information security.

To this concern, it's worth mentioning the work developed by the IETF in RFCs 1244 [34] and 2196 [35], being the latter an update of the first, that presents a guide to develop computer security policies and procedures for sites that have systems on the Internet. For its part, NIST (National Institute of Standards and Technology) special publication 800-12 [36] outlines the advantages of using certain security controls and the situations in which each one is appropriate. Topics covered include: security policy management, risk management, security accreditation and guarantees, access control, disaster response plans, physical security, audits, etc. Initially, the document was aimed at those responsible for sensitive federal systems, but most of the practices described may also apply to the private sector.

Looking deeper into the field of network security, it stands out the ITU recommendation X.800 [37] detailing the security services used to protect the interconnection of systems and the security mechanisms needed to implement them. In addition, the recommendation defines in which layer of the OSI model (Open System Interconnection) can be applied each service, and includes a chapter dedicated to security management.

Most of the network security studies that appeared in the literature in the mid 90's were very specialized on an specific security sector (industrial, governmental or military), fact that made almost impossible to use them outside the sector in which they had been developed. Based on this problem, it emerged the idea of trying to develop a common framework for the study of network security that allowed to joint the efforts being carried out by the research communities of the different sectors. To this end, the NSA (National Security Agency) created the NRM (Network Rating Model) in 1995 identifying nine key security attributes [38]: Privacy, Integrity, Accountability, Availability, Reliability, Connectivity, Recovery, Liability and Uncertainty. To conclude, it is worth mentioning the RFC4949 [39] that contains a glossary of terms used in the field of network security.

1.2.2 SIP Security

The traditional SIP architecture follows a client-server model, where SIP signaling messages are of request/response type. The SIP specification [11] defines six methods: REGISTER for registering contact information, INVITE, ACK, and CANCEL for setting up sessions, BYE for terminating sessions, and OPTIONS for querying servers about their capabilities. Responses are numerical (formed by 3 HTTP-like -Hypertext Transfer Protocol- [40] digits).

The management of the signaling process is distributed among five types of logical entities: The *User Agent* (UA) which acts as a terminal in the SIP communication, the *SIP Proxy* which is responsible for forwarding the requests from the User Agent to the next SIP Server, the *Redirect Server* whose role is to respond to the resolution of names and the location of the user when the destination is outside of the domain of the user who calls, the *Registrar* which is responsible for

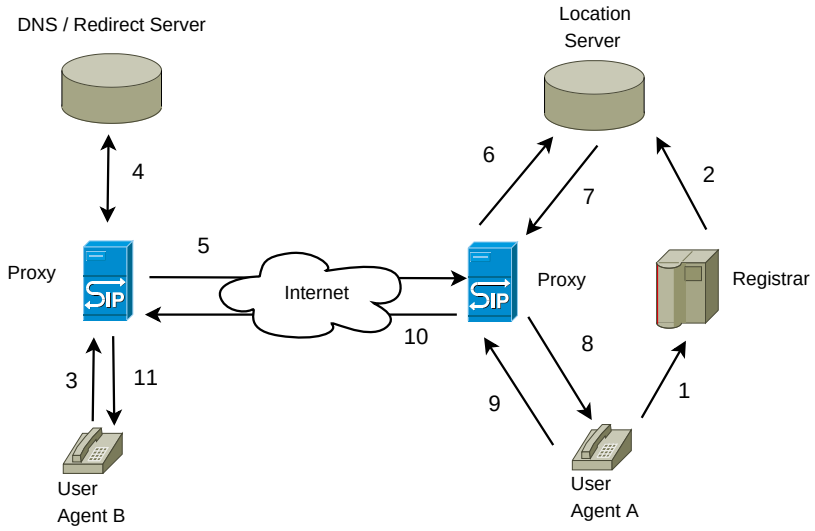


Figure 1.1: *SIP Architecture.*

the registration of users and the *Location Server* which is responsible for storing the searches carried out by the *Registrar*. Within this architecture, users are identified by a unique SIP-URI (Session Initiation Protocol Uniform Resource Identifier) of the type *SIP:user@domain*.

Figure 1.1 shows the architecture of SIP. In this example, user agents A and B are in different domains and are associated to different SIP proxies. Initially, A registers herself with the Registrar (1), using a REGISTER message, and the Registrar stores A's contact information into the Location Server (2). When B starts a call by sending an INVITE request to her SIP Proxy (3), this one consult the DNS (4) and forward the request to A's SIP Proxy (5). Then, the latter obtains A's IP address contacting the Location Server (6,7) and sends B's INVITE request to A (8). Finally, A replies back to B's request using the same route (9,10,11). The flow of messages exchanged between the two users of the system in order to establish and finish a communication is represented in Figure 1.2.

In relation to SIP security, Section 26th of the standard [11] describes the more common attacks that can be launched against the SIP protocol:

- *Registration hijacking*: The absence of cryptographic assurance of a request's origin could let a malicious user A to impersonate another user B of the system, being able, for example, to de-register A's contact information and then register her own device B as the appropriate contact address, thereby directing all requests for the affected user A to the attacker's device B.

- *Impersonating a server*: The absence of a cryptographic mechanism allowing the users of the system to authenticate the servers to whom they send requests, raises a possibility for an attacker to impersonate the remote servers.

- *Tampering with message bodies*: SIP message bodies should be cryptographically protected to prevent intermediate servers to access or modify their contents.
- *Tearing down sessions*: If principals in a session cannot be certain of a request's originator, an attacker could insert a BYE request in a session and tear it down.
- *Denial of service and amplification*: SIP creates a number of potential opportunities for DDoS (Distributed Denial of Service) attacks that must be recognized and addressed by the implementers and operators of SIP systems.

Also, the SIP standard describes some of the security mechanisms that can be used to prevent the mentioned attacks:

- *Transport and network layer security:* TLS (Transport Layer Security) [41] or IPSec (Internet Protocol Security) [42] can be used to provide confidentiality and integrity to the signaling messages. Also, certificates can be used to authenticate them.
- *Secure identifiers SIPS-URI*: SIPS-URI (Session Initiation Protocol Secure Uniform Resource Identifier) allows resources to specify that they should be reached securely.
- *HTTP authentication*: HTTP Digest authentication scheme in SIP allows replay protection and one-way authentication.
- *S/MIME*: S/MIME (Secure/Multipurpose Internet Mail Extensions) can provide end-to-end confidentiality and integrity for message bodies, as well as mutual authentication.

The paper [43] makes a good review of these attacks supplemented with other problems such as spam in VoIP networks, as well as defining the necessary security requirements to establish a secure session: confidentiality and message integrity, authentication of SIP entities, availability of agents and SIP servers, and traffic privacy. A larger study on spam in communication networks can be found in [44], where three types of spam that may affect this kind of networks are defined: SPIT (Spam over Internet Telephony), SPIM (Spam over Internet Messaging) and SPPP (Spam over Presence Protocol). Besides, it presents a discussion of the measures that can be taken to combat spam: content filtering, black and white lists, consent-based communications, reputation systems, address obfuscation, limited-use addresses, Turing tests, computational puzzles, payments at risk and legal action.

In relation to the eternal dilemma of finding a balance between performance and security, we can find a study on the loss of performance induced by making a VoIP transmission through a secure communication channel established using IPSec in [45]. In turn, the paper [46] studies the process of authentication in SIP and its performance, while the paper [47] analyzes the security for deploying IP telephony in the critical infrastructure.

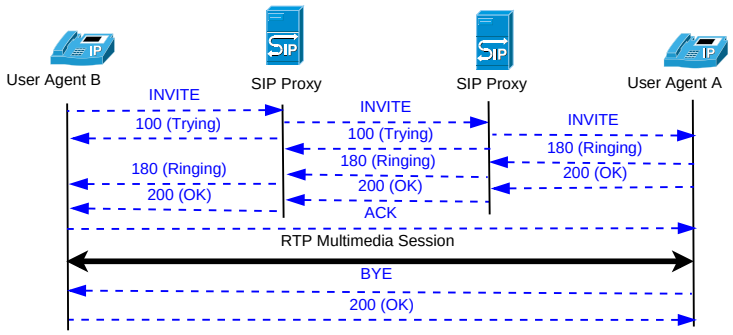


Figure 1.2: *Connection Flow in SIP.*

Finally, it is worth mentioning two very important documents. NIST Special Publication 800-58 [48] discusses the security considerations to be taken when designing a VoIP network, from its planning and maintenance, to the study of the QoS (Quality of Service), the implementation of IPSec tunneling for secure communications, or the use of external security measures such as firewalls and NAT (Network Address Translation). For its part, CCN (spanish Centro Criptológico Nacional) guide CCN-STIC-414 [49] studies the architecture of a secure VoIP network and analyzes some solutions for the attacks that these networks can suffer.

1.2.3 P2P Security

According to [50], P2P networks can be classified based on two parameters: their degree of centralization and their structure. In relation to their degree of centralization we can find completely decentralized networks, partially centralized and hybrid. Regarding their structure there are two kinds of well differentiated networks: structured and unstructured. In this thesis, we will focus primarily on structured P2P networks because they have the necessary requirements to be used as a substrate for SIP: load balancing, and efficient message routing and search. Examples of structured P2P networks are Chord [51], Kademlia [52], CAN [53], Pastry[54] or Tapestry [55]. A survey and comparison of these different schemes is presented in [56].

Structured P2P networks maintain a DHT (Distributed Hash Table) and make each node responsible for a specific part of the system content. These networks use hash functions and assign values to each content and node in the network. Thus, whenever a node wants to search certain resources, it determines the node responsible for them and directs the search to it. One of the most popular structured P2P network, and that we use as an example to illustrate the operation of this kind of architecture, is Chord [51]. This decision has not been picked at random, but because Chord is the structured P2P network used in the proposal [57] of the new P2PSIP communication system currently being developed by the P2PSIP working group of the IETF. In Chord, when a node joins the network, it receives an identifier (node-ID) that is obtained by calculating the hash of its IP address. The same hash function is used to create the resource-IDs

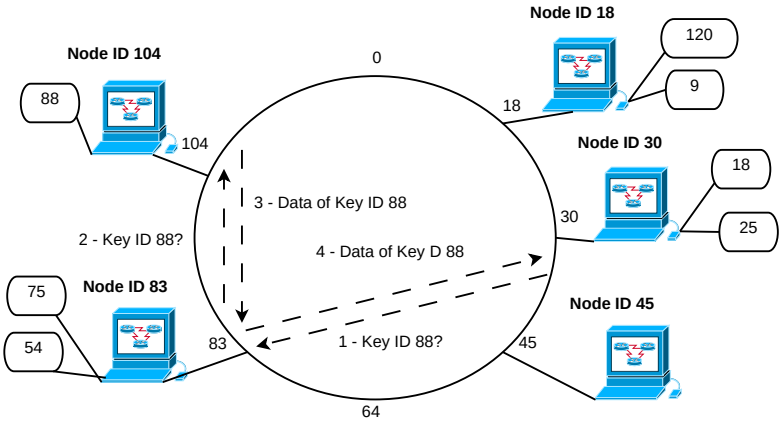


Figure 1.3: *Chord Architecture.*

for any content (information) to be stored in the Chord network. Chord uses a logical ring as the underlaying structure for the routing of messages and the searching for resources. Within this ring, nodes are ordered clockwise, from 0 to 2^{m-1} (being m the size in bits of the identifiers), according to their node-ID. Each node is responsible for storing all the resource-IDs that are equal to or less than its own identifier but larger than the identifier of its predecessor in the ring. Also, with routing purposes, each node maintains a routing table with its predecessor and its successor in the ring, and a set of links to nodes located at different parts of the ring called fingers.

In order to join the network, and after calculating its node-ID by hashing its IP address, a node contacts one or more bootstrap nodes and is routed to the node with the least ID (identifier) greater than its node-ID. Then, the joining node, its predecessor and successor exchange messages to update their routing tables. Something similar happens to store resources on the network, which are routed to the nodes responsible for the content of the resources.

A example of the architecture of Chord is presented in Figure 1.3 that shows the flow of messages for a content search in a ring Chord with $m = 7$ and five nodes in the network. If a node with node-ID 30 wants to find a resource with resource-ID 88, it first contacts the node in its routing table with the closer and smaller node-ID than the searched resource-ID (1), node 83 in this case. Then, node 83 forwards the query to its successor (node 104) that is the responsible for the resource (2). Finally, node 104 replies with the data requested (resource-ID 88) to node 83 (3), which in turn forwards the reply to the requesting node (4).

Once described the operation of this kind of networks, it is easy to infer that exist a number of services that could be used maliciously. Accordingly, it is necessary to establish mechanisms that provide an adequate level of security. One of the early works on security in the area of P2P networks was published in [58] related to the security of Naspter [59] and Gnutella [27], presenting a short analysis of the security of each protocol and discussing the advantages that

each one has against the other. Another interesting paper in this area is [60], that presents a exhaustive survey of the attacks that may be faced by structured P2P networks, divided into three sections:

- *Routing attacks*: Incorrect lookup routing, incorrect routing updates and partition during bootstrap.
- *Storage and retrieval attack*: Denial of the existence of a resource.
- *Miscellaneous attacks*: Inconsistent behavior, overload of targeted nodes and rapid joins and leaves.

For its part, in [61] is presented a quantitative analysis of the solution to possible attacks against the routing system, the storage of the P2P network and the allocation of identifiers. The problem of assigning identifiers is also studied in [62, 63]. A completely different point of view is shown in the technical report [64] that carries out a description of P2P systems' vulnerabilities and organizes them based on the layer of the system they affect. Following the same line, the paper [65] defines secure routing as the basis for building secure applications over P2P networks. For the implementation of secure routing three problems should be solved:

- *Secure node-ID assignment*: It prevents an attacker from choosing the value of the identifiers assigned to the nodes she controls. It proposes as solution the use of central certifying authorities that assign to each node a certificate tied to its identifier.
- *Secure routing table maintenance*: It controls that the number of malicious nodes that appear in the routing tables of correct nodes does not exceed the fraction of malicious nodes across the network. The proposed solution is to use constrained routing tables.
- *Secure message forwarding*: It ensures that at least one copy of each message reach its destination with high probability. The use of a routing failure test in conjunction with diverse routing is proposed as solution.

Also, this paper studies the use of self-certified data as a complement to the use of secure routing.

The paper [66] presents a review of the concepts discussed in the paper described above to achieve secure routing. Also, it studies using self-certified data and a system of quotas to protect the storage system, and presents a review of trust in P2P overlays. In turn, [67] proposes to exploit redundancy for achieving a secure routing in the presence of faulty nodes and describes how to create the structures (routing tables) necessary to carry it out. Another proposal for secure routing is presented in [68], that based on the use of certificates distributed within the network, allows a node to be able of verifying the authenticity of the responses to the messages

it sends. In [69], another alternative is presented that diversifies the amount of trust that is placed on the neighboring nodes when routing a message, with the objective of minimizing the ratio of incorrect routing.

The paper titled *The Sybil attack* [70] shows that, without a logically centralized authority, it is impossible to limit the number of identities a user can obtain to access the network except under extreme and unrealistic assumptions of resource parity and coordination among entities. Subsequent papers, such as [71, 72], have come up with measures to reduce the effect of this kind of attack based on the use of cryptographic puzzles.

Papers [73, 74] describe the *Eclipse attack* in which an attacker, or a coalition of them, tries to intervene most of the traffic of the network to eclipse the view nodes have ones to each others, allowing an attacker, in the worst case, to control most of the network's traffic. The simplest way, but not the unique, to launch an *Eclipse attack* is through a *Sybil attack* that allow an attacker to introduce a sufficient number of nodes in the network to control the traffic flowing through it.

In [75], authors describe how to make a structured P2P network more robust and scalable using the cuckoo rule: when a new node wants to join the network it moves all the nodes located in its area to new random points within the space of identifiers. In turn, the paper [76] describes P2PSL (Peer-to-Peer Security Layer): a new security layer that sits between the P2P application and the underlying layers. For its part, the work [77] describes rational attacks in which a node attempts to maximize its consumption of the system's resources while minimizing the amount of its own resources which are consumed by the other nodes. Besides, it presents a taxonomy of this kind of attacks and discusses various types of trust and reputation systems as a measure to avoid them. In the other hand, articles [78, 79] attempt to define a framework for admission control in P2P networks based on decentralized voting mechanisms: all the group members must vote to accept or not the entry of each potential new candidate.

Finally, an overview of techniques reported in the literature for making DHT-based systems resistant to three of the main attacks that can be launched by malicious nodes participating in the DHT (the *Sybil attack*, the *Eclipse attack*, and routing and storage attacks) is present in the recently published paper [80].

1.2.4 P2PSIP Security

One of the first telephony applications based on P2P networks was Skype [81, 82]. As is well known, Skype is an application based on Kazaa's architecture [83] offering telephony and instant messaging services among its users, that also allows to establish connections with the traditional telephony network PSTN (Public Switched Telephone Network). Unfortunately, Skype uses a proprietary protocol and a hybrid infrastructure that depends on online central servers.

In 2005 the IETF established the P2PSIP working group with the scope of developing a standard for SIP communication systems over P2P networks. The initial works of the group were based on two projects: one from Columbia University [84] and another from College of William and Mary [19]. These proposals have undergone many changes, among which stands out the development of a new binary protocol RELOAD (REsource LOcation And Discovery) to control the underlying overlay network, rather than using SIP messages to handle it, as was initially proposed. Three have been the main motivations for this change[29]:

- *Adaptability*: The initial architecture was totally focused on SIP making very difficult that it could be adapted to other protocol.
- *Security*: It was necessary to adapt the routing protocol to address the new security scenario where routing messages pass through unreliable nodes and not through reliable proxies as in SIP.
- *Performance*: The performance of a binary protocol is better than of a text-encoded protocol such as SIP, which has a high overhead.

RELOAD follows the previously described (Figure 1.3) P2P model where users' contact information is stored in the overlay network. Before entering into a P2PSIP system, each user is assigned a SIP username and node-ID. The user's node-ID determines the location of the network where the user is placed and the resources she is responsible for, while the place where the user can store her resources is determined by the hash of her username {*resource-ID = Hash(username)*} and node-ID {*resource-ID = Hash(node-ID)*}. When a user A of the system wants to start a multimedia session with other user B, she first calculates the resource-ID where B's resources are stored by hashing B's username and then sends a request to the P2PSIP network asking for the calculated resource-ID to get B's contact information. Once A has received B's contact information, she starts a multimedia session with B by sending a SIP INVITE request as described in Figure 1.4.

Currently, the P2PSIP working group is focused on six IETF drafts: the first [29] presents a general framework for P2PSIP, the second [57] describes RELOAD, the third [85] demonstrates the use of SIP over RELOAD, the forth [86] defines extensions to the RELOAD P2PSIP base protocol to collect diagnostic information, the fifth [87] describes how the default topology plugin of RELOAD can be extended to support self-tuning, that is, to be adaptable to changing operating conditions such as churn and network size. Finally, the sixth [88] provides a service discovery mechanism for RELOAD.

The cornerstone of RELOAD security is the possession of each participating node in the network of one or more public key certificates (PKCs). In [57] two models are presented depending on the requirements and the scenario where the P2PSIP system is going to be deployed.

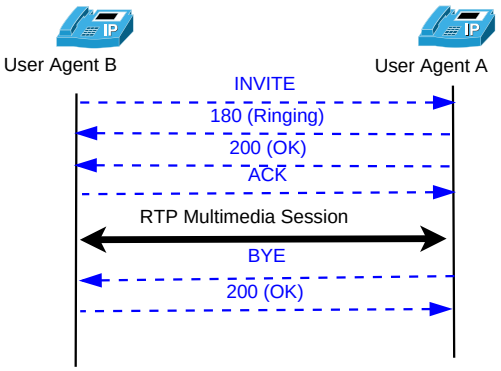


Figure 1.4: *Connection Flow in P2PSIP.*

The first proposal is modeled around the idea of using a central server acting as CA (Certification Authority). First time a user wants to join the network, she must present her credentials (usually a login and a password) to the server. If the access is granted, the server will request to the user a username that identifies her uniquely within the network and a public key (PK). Once checked that the username has not been assigned before, the server accredits the user with a long-term X.509 certificate [89], signed with the PrK (Private Key) of the CA, linking the user public key with one or more usernames and one or more unique node-IDs randomly generated. This certificate serves for multiple purposes:

1. Identifies the user and her node by her usernames and node-IDs respectively, so that when others nodes/users communicate with her can know, checking her certificate, that they are really contacting the correct node/user.

2. Determines the places of the network where the new node can be placed via its node-IDs.

3. Specifies the location of the network where the node/user can store her resources, determined by the hash of her usernames {*Resource-ID = Hash(username)*} and node-IDs {*Resource-ID = Hash(node-ID)*}, hence controlling the access to the network resources. This access is managed through the definition of data kinds, as described later.

4. Allows adding integrity and authentication to messages by means of digital signature performed with user's public key, and communication confidentiality using the same key to establish TLS or DTLS (Datagram transport Layer Security) [90] sessions.

These certificates are stored within the network in the location reserved for the resources of each user, forming a list in which new certificates are added at the end, fact that facilitates their renewal.

The other model presented in [57] is based on the lack of a central CA. Access to the network is controlled by means of a shared secret key, used to key TLS-PSK (Pre-Shared Key ciphersuites

for TLS) [91] or TLS-SRP (Secure Remote Password -SRP- Protocol for TLS) modes [92]. Thus, a user who does not have the key cannot establish a TLS communication with the other members of the network and hence access to it. In this case, certificates are self-generated and self-signed by the users themselves that use as node-ID the hash of their PK.

The first model is the recommended for the vast majority of scenarios. The second model, however, it is appropriate for small groups that wish to form a private network without complexity and it should be only used in closed networks where users are mutually trusted.

In relation to its routing, RELOAD is a message-oriented request/response protocol. Messages are transported using TLS or DTLS. Each node establish a TLS or DTLS session with its fingers (nodes used to route its communications through the network) and uses its certificate to authenticate itself to the other participant of the connection, and to establish the symmetric session key to cipher the communication. Each message has three parts:

- *Forwarding header:* Generic header used to forward the message between peers. This header is the only information that an intermediate peer needs to examine. Among the whole bunch of attributes in this header, we present a description of the most relevant:
  - *Transaction-ID:* Unique ID that identifies the transaction.
  - *Via-list:* Contains the sequence of destinations through which the message has passed. It starts out empty and grows as the message traverse each peer.
  - *Destination-list:* Contains a sequence of destinations which the message should pass through. The last element of the list specifies the ultimate recipient of the message or the searched resource.
  - *TTL:* TTL (Time To Live) specifies the number of hops a message can experience before being discarded.
- *Message contents:* The message being delivered between peers.
- *Signature:* An optional digital signature performed by the sender over the message contents and parts of the message's header.

The basic routing mechanism used by RELOAD is symmetric recursive. In symmetric recursive routing, in order to route a message, each peer forwards the message closer to its destination. Once the message reaches its destination, the response is routed back using the reverse path. The use of iterative routing is also defined in the standard draft. Other extra routing alternatives, such as direct response and relay peer routing, are defined in [93].

The *destination-list* and *via-list* fields in the forwarding header are used to set the destination and to establish the return path of the message, respectively. Figure 1.5a shows an example of a

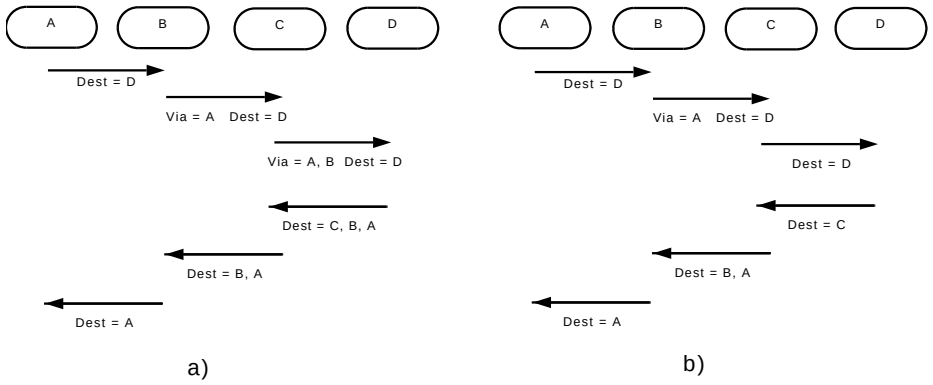


Figure 1.5: *a) RELOAD recursive symmetric routing. b) RELOAD recursive symmetric routing with peer C using a truncate list.*

message following a route from A to D through B and C; and the contents of the *destination-list* and *via-list* fields during the route. In this example, all intermediate peers use a full *via-list*. In turn, Figure 1.5b shows a message following the same route where peer C uses a truncate *via-list*, i.e., instead of adding the route followed by the message to its *forwarding header*, the peer saves the information internally (keyed by the *transaction-ID*) and returns the response message along the path from which it was received.

TLS or DTLS sessions provide the communication with finger-to-finger authentication, integrity and confidentiality. Authentication of the sender and integrity of the data is granted end-to-end in the request via the digital signature of the sender. Confidentiality of the request cannot usually be assured because the originating peer does not know which node stores the desired resource and, therefore, it cannot encrypt the data with the recipient public key. In turn, the peer recipient of the request can encrypt the response with the originating peer's public key and sign the response with its private key, granting end-to-end authentication, integrity and confidentiality of the response.

Storage in RELOAD is distributed among the nodes forming the P2P overlay. Each location within the overlay ID space is referenced by a resource-ID, being each node responsible for the resource-IDs located between its predecessor in the Chord ring and itself. Each location may contain multiple kinds of data identified by a kind-ID (data Kind ID). The definition of each data kind specifies rules for determining which certificates can access each resource-ID / kind-ID pair, controlling this way the data access. Examples of access control policies are: USER-MATCH (only users whose usernames hash to the resource-ID can write the resource) or NODE-MATCH (only users whose node-IDs hash to the resource-ID can write the resource). Authorization and integrity of the resources is granted by means of digital signature. RELOAD storage layer also implements data replication to avoid data-lost in case of node failure or malicious behavior.

In relation to general P2PSIP security, the work [94] reviews the security of P2PSIP archi-

tectures, identifying the following attacks:

- *Attacks on the ID mapping scheme*: It discusses the inappropriateness of some alternatives such as using the hash of the IP address or the hash of the IP address in conjunction with the port to calculate the user's ID. Also, it discusses the problem of authentication in P2PSIP systems, studying two possible solutions: using a central certifying authority or using a distributed trust algorithm.
- *Attacks on overlay routing*: It covers some of the proposals that can be found in the P2P literature, previously discussed in this document.
- *Bootstrapping*: If the admitting node contacted by the new joining node in order to bootstrap is malicious, the new node can be easily attacked.
- *Free-Riding*: Nodes use services, but fail to provide services to the network.
- *SPIT*: Spam over IP telephony networks.

The research carried out in [95, 96] presents a study on the assignment of identifiers, the use of certificates and the different routing methods that can be used in P2PSIP networks. Some of the recommendations presented are: using an offline certification authority for the assignment of IDs and certificates, and using recursive routing mainly due to its better performance in situations of non-transitivity. For its part, the paper [97] examines the use of self-certified SIP-URIs to authenticate the resources in the network. Finally, it is worth noting the paper [98] developed by the IETF working group that aimed to analyze the security requirements P2PSIP systems have. Unfortunately, the wording of this document has been abandoned and it is incomplete and out-of-date.

1.3 Thesis Methodology

As we have already stated in the hypothesis (Section 1.1) of this thesis, P2PSIP systems present a completely new security problematic in comparison to traditional SIP systems and file-sharing P2P networks. This, in conjunction with the absence of a specific analysis of the security of P2PSIP systems in the literature, supports the necessity of a thorough analysis of the new security issues P2PSIP systems present.

Based on this need, the objective of this thesis is the improvement of certain security services involved in P2PSIP communication systems. To this end, the realization of this thesis has been divided in five phases that will be detailed in the following paragraphs. In general terms, we begin with the identification of the problem to be studied and then we define the objectives and scope of the thesis. Subsequently, there will be a search in the literature to establish the

current state of the art of P2PSIP communication systems. Once completed the state of the art, we will develop a taxonomy of attacks that can disrupt the functioning of such systems and to study the existing countermeasures that can be taken to avoid them. Then, in base of this taxonomy, we are going to focus our study in the service that is the cornerstone of the security of P2PSIP communication systems, the access control, and develop specific measures to improve its security. Finally, a set of security recommendations that should be considered when designing, implementing and managing a communication system based on P2P networks will be developed.

Phase 1. Problem statement and state of the art

In this first phase, we identify the problem statement, the scope of the thesis and its objectives. Then we carry out a study about the theoretical aspects related to computer networks, communication systems, P2P networks and, finally, P2PSIP communication systems, mainly focused on their security issues.

Phase 2. Study of the attacks on P2PSIP communication systems and their countermeasures

From the study on security issues held in the previous phase, we develop a taxonomy of attacks that can disrupt the operation of communication systems based on P2P networks, to then analyze the existing countermeasures to prevent them and their viability.

Phase 3. Design of security solutions to improve the access control of P2PSIP communication systems

Based on the taxonomy of attacks presented and the deficiencies in the existing countermeasures that can be taken to prevent these attacks, we carry out the design of new specific solutions to improve the security of P2PSIP communication systems. In order to do so, we focus on the service that stands as the cornerstone of P2PSIP communication systems' security: access control. In particular, we address three points:

- Design of a new certification model based on the segregation of the identity of users and nodes.
- Design of a new model for secure access control for on-the-fly P2PSIP systems.
- Design of a new authorization framework for P2PSIP communication systems based on attribute certificates.

Phase 4. Security recommendations for the development of P2PSIP communication systems

After the study on the security of P2PSIP communication systems and the design of specific solutions to improve it, we develop a set of security recommendations that should be considered for the design, implementation and maintenance of P2PSIP communication systems. These recommendations will be not unique, but vary depending on the scenario and its security requirements.

Phase 5. Conclusions

Finally, after the process of security analysis, description of recommendations and evaluation, we present the conclusions of the research conducted in this thesis.

1.4 Thesis Roadmap

Chapter 2 presents a taxonomy of the attacks that can be launched against P2PSIP communication systems and studies the defenses that can be taken to prevent them.

Chapter 3 describes a new certification model for P2PSIP communication systems based on the segregation of the identity of users and nodes.

Chapter 4 introduces a new access control model for on-the-fly P2PSIP communication systems.

Chapter 5 studies the advantages of a new certification framework for P2PSIP communication systems based on the use of attribute certificates.

Chapter 6 develops a set of recommendations for the design, implementation and maintenance of P2PSIP communication systems.

Chapter 7 displays the conclusions of the research conducted in this thesis.

1.5 Main Contributions

The main contributions of this thesis are reflected in the publications we present below.

In relation to the security analysis of network applications:

2011 D. Touceda, J.M. Sierra, A. Izquierdo and H. Schulzrinne, "Survey of Attacks and Defenses on P2PSIP Communications,"Communications Surveys and Tutorials", IEEE, Accepted for publication. *JCR Impact Factor yet to be determined*

2007 Diego Suarez, Joaquín Torres Márquez, Mildrey Carbonell and Jesús Téllez Isaac, "A new domain-based payment model for emerging mobile commerce scenarios". DEXA Workshops 2007: 713-717. *CORE RANK B*

In the relation to new models of certification for P2P systems.

2011 D. Touceda, J. Camara, L. Villalba, and J. Marquez, "Advantages of identity certificate segregation in P2PSIP systems," Communications, IET, vol. 5, pp. 879–889, Apr. 2011. *JCR Impact Factor: 0.751*

Related to the proposal and evaluation of new models of authentication:

2011 D. Touceda, J.M. Sierra and M. Soriano, "Secure Access Control for on-the-fly P2P Systems", Computer Networks, Elsevier, Under Review. *JCR Impact Factor: 1.201*

2009 Viedma Astudillo, M., Téllez Isaac, J., Suarez Touceda, D., and Plaza López, H., "Evaluation of a Client Centric Payment Protocol Using Digital Signature Scheme with Message Recovery Using Self-Certified Public Key". In Proceedings of the international Conference on Computational Science and Its Applications: Part II (Seoul, Korea, June 29 - July 02, 2009). Lecture Notes In Computer Science, vol. 5593. Springer-Verlag, Berlin, Heidelberg, 155-163. *CORE RANK C*

In relation to a more secure and flexible authorization:

2011 D. Touceda, J.M. Sierra and M. Soriano, "On Authorization for P2P networks", Computer Communications, Elsevier, Under Review. *JCR Impact Factor: 0.933*

2008 Carbonell, M. Torres, J. Suarez, D. Sierra, J.M. Tellez, J., "Secure e-payment protocol with new involved entities". In Collaborative Technologies and Systems (CTS), 2008, 103-111. *CORE RANK C*

CHAPTER 2

ATTACKS AND DEFENSES ON P2PSIP

> To know your Enemy, you must become your Enemy.
>
> Sun Tzu - The Art Of War

As we have already introduced, P2PSIP presents a completely new security problematic in comparison to traditional SIP and P2P file-sharing systems. Motivated by this new problematic, in this section we conduct an analysis of the attacks that can be launched against a P2PSIP system and the defenses presented in the literature to prevent them, with the intention of achieving a better understanding of the new security challenges P2PSIP communication systems present.

Our analysis starts with the identification of the different services (Figure 2.1) forming the new P2PSIP communication system's architecture:

- **Access Control:** Access control is the service in charge of deciding which entities are allowed to join the system, and use its resources, and which ones are not. Once this decision

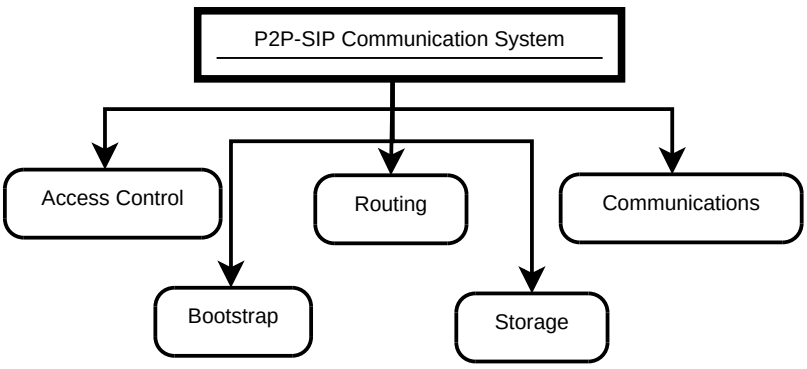


Figure 2.1: *P2PSIP Communication System Services.*

has been taken, the service must assign to each user a unique ID that identifies her within the network. Also, it should link the user ID with her permissions on the system's resources. Without a robust access control system, the whole security of a P2PSIP network can be compromised.

- **Bootstrap:** Bootstrapping is the process through which a node contacts other nodes (or servers) already connected to the network in order to initialize its status and be able to operate within the system. During this process, among other actions, the new node places itself in the location of the network indicated by its node-ID, informs its neighbors about its presence in order to initialize its routing table and to store the resources it is responsible for.

- **Routing:** The routing service is in charge of delivering all the messages exchanged between the nodes of a P2PSIP network. These messages range from users' contact information requests/answers to control and informational messages to maintain the overlay.

- **Storage:** The storage service saves the contact information of the network's users in order to permit them to communicate with each other. Unlike client-server networks, where this task is performed by a dedicated server, in P2P networks it is distributed among the nodes of the system. Also, this service is responsible for storing private and public users' resources such as voicemail messages, public certificates, etc.

- **Communication:** The role of this service is to establish communications, mainly messaging, telephony and videoconferencing. The underlaying P2P architecture permits to offer these services without the requirement for permanent proxy or registration servers.

In the following sections we analyze the security challenges P2PSIP communication systems present. Based on the distinction of the services offered by a P2PSIP communication system, we describe the attacks that can affect them. For each attack, a summary of the defenses that can be adopted to secure the affected service is presented. Also, an analysis of miscellaneous attacks that do not target any specific service but the whole system is included. Figure 2.2 overviews these attacks.

2.1 Access Control

The first line of defense of a P2PSIP system against possible attacks is access control. Without a robust access control system, the whole security of a P2PSIP system can be compromised. In this section we analyze the security of this service by presenting the existing attacks and defenses discussed in the literature. A summary of this security analysis is shown in Table 2.1

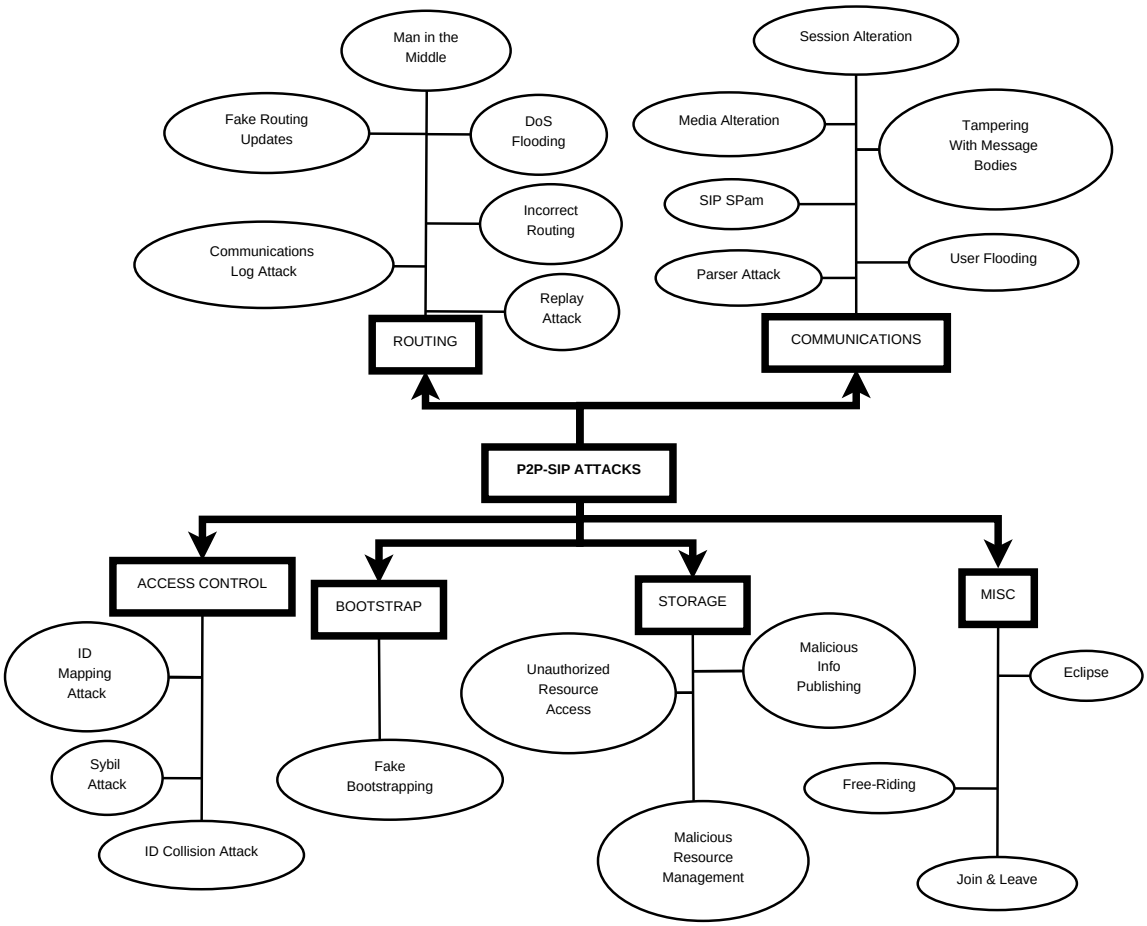


Figure 2.2: *Attacks against P2PSIP Communication Systems.*

that presents the existing attacks against this service, the security requirement at risk on each attack, and the defenses for each attack and their references in the literature.

2.1.1 ID Mapping Attack

Every participating node in the P2PSIP system is assigned a random node-ID that indicates in which part of the overlay it is located. Based on this identifier, the node is made responsible for a space of resource-IDs within the network. This responsibility includes tasks like storing all resources (users' contact information, voicemails, etc.) with any resource-ID within the node's resource-ID space, making those resources available to the rest of users in the system or establishing an access control policy to decide which users can store, access, modify or remove them. With this being said, it is easy to see how easily a malicious user could monitor, prevent the access to or corrupt the resources assigned to the node she controls (*Malicious Resource*

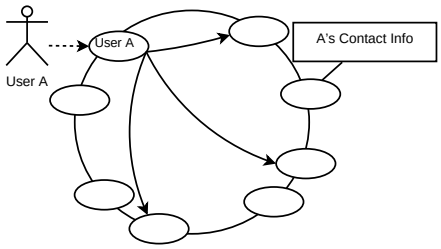
Table 2.1: *Access Control Summary of Attacks and Defenses.*

ACCESS CONTROL			
Attacks	**Security Requirement at Risk**	**Defenses**	**References**
ID Mapping Attack	Integrity	Hashing Based node-ID	[19] [51] [52] [54] [60] [61] [62] [65] [94] [96] [99]
		Node-ID Rotation Based	[62] [63] [75] [100]
	Availability	Centralized node-ID	[57] [61] [65] [82] [96]
Sybil Attack	Integrity	Limitation of IP Addresses	[19] [51] [52] [54] [60] [61] [62] [65] [94] [96] [99]
		Limitation of Computational Resources	[65] [71] [70] [101] [102]
		Limitation of Human Resources	[102] [103]
		Decentralized Trust Model	[104] [105]
		Threshold Cryptography	[106] [107] [108] [109] [110] [111] [112]
		Identity Based Access Control	[113] [114]
		Social Networks	[115] [116] [117]
	Availability	Shared Secret	[41] [57] [90] [91] [92] [118]
		Centralized Control	[57] [65] [66] [70] [82] [96]
		Trusted Devices	[119] [120]
ID Collision Attack	Integrity	Centralized node-ID	[94]

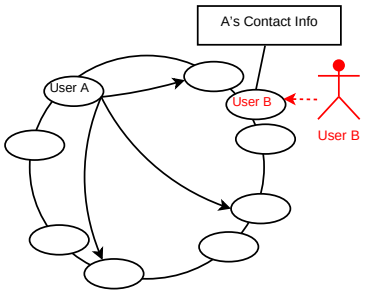
Management Attack[1]).

A big problem arises when an attacker is free to choose her node-ID in the P2PSIP system, because, in this way, she could also choose the space of resources of the network she controls. After a good user A has joined the system (Figure 2.3a), a malicious user B could choose her node-ID to be made responsible for A's contact information (Figure 2.3b) to, for example, deny any attempt from other users to contact A. Even when resource replication is implemented, a coalition of malicious nodes could easily position themselves at all the node-IDs where the replicas are stored. Likewise, a coalition could similarly choose their node-IDs to maximize their chances of appearing in the routing table of a victim node (Figure 2.3c), controlling the victim's access to the network (known as *Eclipse Attack* and described in Section 2.6.3) preventing her to establish any multimedia communication with the other users of the system. This ability of an attacker to freely choose her node-ID is what most of the literature call *ID Mapping Attack*. Below, we describe the different approaches presented in the literature to combat this attack.

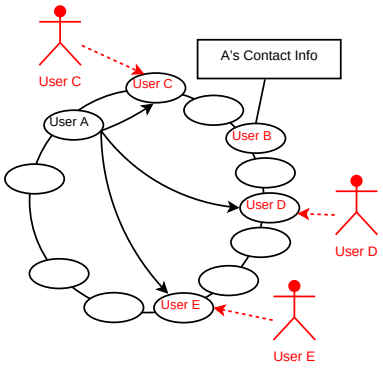
[1]The *Malicious Resource Management Attack* is related to the storage layer. Its description and the mechanism to mitigate it, like replication and authentication of resources, are described in Section 2.4.3. It is introduced here just as a background to better understanding the drawbacks of suffering an *ID Mapping Attack*.



(a) User A joins the system. Initializes her routing table and stores her contact information.



(b) Malicious user B chooses her node-ID to be made responsible for A's contact information and joins the system.



(c) Malicious users C, D and E choose their node-IDs to appear in A's routing table and join the system.

Figure 2.3: *ID Mapping Attack.*

Hashing Based node-ID Assignment

An alternative, used in Chord [51], Kademlia [52] and Pastry [54], is to hash the IP address of the node to calculate its node-ID using a secure hashing function like SHA-1 (Secure Hash Algorithm) [99]. The use of a one-way hashing function to derive the node-ID provides a random spread of nodes around the ID space and guarantees that it is not feasible to invert it, i.e. to find the IP address mapping a specific node-ID. Unfortunately, due to the fact that the number of nodes in the network is normally much smaller than the ID space, an attacker does not need to find an IP mapping a specific ID, but one mapping inside the desired interval in the node-ID space [65]. A quantitative analysis and an experimental validation of this problem is presented in [61], showing that this can be achieved offline in $O(n)$ operations being n the number of non-malicious nodes in the network. Despite being very difficult for a node to have access to such

amount of addresses with IPv4 (Internet Protocol version 4) (unless the attacker has access to the range of addresses assigned to a university or a large company), it is not when IPv6 (Internet Protocol version 6) is in use or when the node IP address is dynamically assigned from a large pool (e.g. an ISP -Internet Service Provider- dynamic address pool for its clients). Another drawback of this approach is the fact that every IP address is exactly mapped to one specific ID. Despite being a simple security measure against *Sybil Attacks* (described in Section 2.1.2), it does not work well with nodes behind a NAT, sharing a common public IP address.

Some solutions presented in the literature to allow users sharing the same IP address to join the system suggest including the client port used by the connection in the node-ID calculation, so that different users sharing the same IP address use different ports to get different node-IDs. The first approach, used in SOSIMPLE [19] and also discussed in [62], hashes a concatenation of the IP address and the client port to calculate the node-ID. The main drawback of this approach, as found in [94] and [96], is that facilitates an *ID Mapping Attack*. Even when the attacker has only one IP address available, she can try ports at will until the hash of *[IP:Port]* matches the desired value. The second approach suggested in [96] to mitigate the problems introduced by the first one, hashes the IP address to calculate an initial node-ID and then substitutes the least significant 16 bits of the node-ID with the port number of the client. This way, the ID space of the network available for each IP address is contiguous and just 16 bits long. However, to perform an *ID Mapping Attack* is still easier than when only the IP address is used to feed the hashing function. Also, this approach does not work well when replication is implemented in contiguous nodes, as done in Chord [51].

Besides hashing the IP address of a node, Pastry [54] suggests another alternative, also presented in [60], for node-ID generation: to hash the node's public key. Despite getting a random distribution of nodes through the ID space and uniquely associating the node-ID of the user with her public key, it is still possible to launch an offline attack against this alternative, as noted in [62]. All an attacker has to do is calculating different public keys until one hash to the desirable node-ID.

Node-ID Rotation Based Assignment

The proposal presented in [100] computes the identifiers by hashing the IP address of a node and a random number obtained from a randomness service. Also, after a period of time called epoch, nodes should get another random number, recalculate their identifiers and join the network again (probably at a new location). This technique is combined with periodic resets of the nodes' routing tables, and a rate limitation of routing updates. Despite achieving a random spread of nodes around the network and making the network resistant against *Eclipse Attacks*, this approach inherits some of the problems related to node-ID assignment based on IP addresses,

like not working well with nodes behind a NAT. It also adds a new drawback, the latency introduced by the continuous process of nodes' arrivals and departures (churn).

A similar strategy, called k-rotation[2], that presents the same advantages is discussed in [63]. When a node wants to join the network, it is introduced in a random position, replacing an old node. Then, the replaced node is moved to a new random position replacing another node, continuing this process for *k-1* rounds. Afterwards, a new free position is created for the last replaced node. The main drawback of this approach is the high cost introduced in the network by the join operation that could be exploited by an attacker (or a coalition of attackers) to launch a *Join and Leave Attack* (described in Section 2.6.3) requesting a huge number of simultaneous joins. Other examples of this kind of strategies can be found in [62] and [75].

Centralized node-ID Assignment

A centralized scheme is presented in [61] that suggests using a publicly known and trusted bootstrap server that assigns a random node-ID and issues a certificate with a short life-time to a node every time it joins the network. This approach prevents the use of offline attacks against the node-ID generation but it is susceptible to an attacker that joins and leaves the network until she gets assigned the desirable node-ID. The cost of this attack has been showed to be *O(n)* contacts to the bootstrap server [61], being n the number of non-malicious nodes in the network. Despite having a higher cost than an offline attack, it is not enough to prevent an attacker from gaining control over a specific target. One way to prevent this could be to implement weak security checks through the bootstrap server, like detecting frequent attempts by a node from a single IP to join the system. Unfortunately, this approach has another drawbacks like a single point of failure in the bootstrap server that should be contacted every time a node joins the network, and inability to maintain persistent node information as node changes its node-ID every time it joins the network. A similar classical approach is to use several login servers, like Skype [82]. This scheme reduces the impact of possible single point of failure but increases the maintenance and deployment cost of the infrastructure.

Another server based technique introduces an offline centralized CA in the system. A first generic approach for P2P systems [65] suggested cryptographic node-ID certificates binding a node-ID, chosen randomly by the server, to a public key generated by the client and an IP address. Since we want to support user mobility, binding the user's certificate to an IP address is not a good idea for P2PSIP. Other approaches, [57] and [96], suggest binding a user public key to zero or more (typically one) usernames and to zero or more (typically one) node-IDs randomly generated by the server. Also, all users receive a copy of the CA's public key in order to validate other users' certificates and publish the public portion of their certificates in the

[2] k is a configuration parameter.

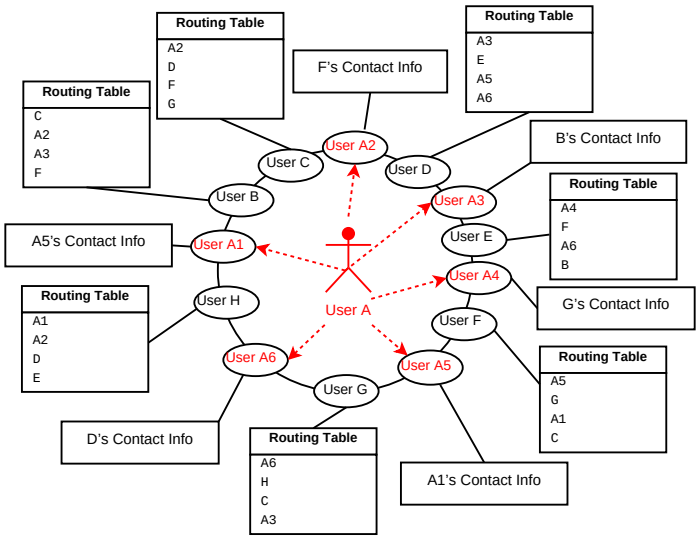


Figure 2.4: *Sybil Attack where malicious user A joins the network with several identities.*

network to allow secure offline messages (like voicemail). The CA ensures that node-IDs are chosen randomly from the ID space, and prevent nodes from forging them. The main drawback of this approach is having a single point of failure in the CA server. Nevertheless, the fact that it must be contacted by nodes only the first time they join the network to obtain a signed certificate and periodically to renew it, minimizes this problem. Again, it is also possible the establishment of an infrastructure of several CA servers to reduce the impact of a possible single point of failure.

2.1.2 Sybil Attack

Even when replication is securely implemented and a user can not freely choose her node-ID, an attacker could still launch a *Sybil Attack* to take control over a portion of the P2PSIP system. In a *Sybil Attack*, a unique entity (a user) presents to the system different identities, each one associated to a node-ID. This way, if there is not a limitation on the number of identities she can obtain, an attacker could control an unlimited number of nodes and, hence, a big part or even the whole P2PSIP system. As we can see in Figure 2.4, the more nodes a user controls the more probability she has of controlling the other users' communications (by appearing in their routing tables) and having control over their contact information.

This attack was first described in [70], that also shows that, without a logically centralized authority, *Sybil Attacks* are always possible except under extreme and unrealistic assumptions of resource parity and coordination among entities. Below, we present the defenses that can be taken against the attack in question.

Limitation of IP Addresses

Using the hash of the IP address of a node to derive its node-ID provides a slight protection, limiting the number of identities a user can present to the number of IP addresses she has available. Unfortunately, with the problems that using a hash of the node's IP address to calculate its node-ID has, this mechanism cannot be considered secure by itself.

Limitation of Computational Resources

One approach to slow down the rate a user can obtain node-IDs is to exploit the limitation of available resources (computation, storage, etc.). Most of the research found in the literature about this area is related to the use of cryptographic puzzles, first described in [101]. The idea is to make every node solve some sort of computational puzzle to join the network and each certain time interval once they are inside, thus making it costly for an attacker to introduce many nodes in the network. One example of this mechanism is presented in [65]. Their approach requires new nodes to generate a key pair with the property that the SHA-1 hash of the public key (used as node-ID) has the first p [3] bits equal to zero. Also, they suggest the possibility of binding IP addresses with node-IDs by requiring nodes to find a string x such that *SHA-1(SHA-1(IP, x), node-ID)* has p' bits equal to zero. Nodes would be required to present such an x to be accepted by the others. Finally, each certain time interval node-IDs are invalidated by setting a different initialization vector for the hash function and, thus, every node would have to solve a new puzzle periodically. Similar mechanisms to mitigate the Sybil attack using cryptographic puzzles can be found in [71] and [102]. The main problem of cryptographic puzzles (like any other mechanism based on the exploitation of limited resources) is that they only work well under the assumption that all entities operate under nearly identical resource constraints [70]. In networks like P2PSIP, where users can be connected from several kind of devices with such a different amount of resources available (from mobile VoIP phones with low resources to very powerful personal computers), the cost of solving the challenge should be acceptable to the slowest device making easier for an attacker with high computational resources to present multiples identities to the network.

Limitation of Human Resources

A CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) is a program that mainly protects websites against bots by generating and grading tests that humans can pass but current computer programs cannot [103]. CAPTCHAs can be used to ensure that there is a human being behind every operation in the network and, this way, to limit

[3] p and p' are configuration parameters

the number of identities an entity can present in the network. Nevertheless, this solution is not effective in P2P contexts because of the difficulty of imposing this cost in a fully decentralized system. It is impractical to perform a CAPTCHA for every one of the thousands nodes you interact with [102].

Decentralized Trust Model

The PGP (Pretty Good Privacy) "Web of Trust" Model [104] is a decentralized alternative for authentication based on the human evaluation of identities. Each user manually signs the PKs of the users she directly knows giving them certain values of trust. These values of trust can be used by other users to indirectly trust those users. Apart from other issues inherent to its design, the main problem with using the PGP model on the present context is that it is designed to link identities with public keys, but does not prevent an entity from having several identities. The paper [105] presents the application of the "Web of Trust" model to ad-hoc networks with the same drawbacks.

Threshold Cryptography

The paper [106] proposes a distributed access control, based on a (n, t) threshold cryptography scheme [107], where trust is distributed between a set of users of the network forming a DCA (Distributed Certificate Authority). This DCA consist of n users which share a public/private key pair. The public key is known by all the users M of the network, while the private key is divided in n shares, one for each user member of the DCA, having the property that any t shares can reconstruct it, but any *t-1* cannot. Similar proposals have later appeared, such as [108], that distribute the trust among all the users of the network instead of among only a few. These proposals are very suitable for very hostile environments because they guarantee that to forge the access control mechanism it is necessary the collaboration of at least t malicious users. Nevertheless, they have several drawbacks: on the one hand most of the schemes are not really distributed, because they require a TTP (Trusted Third Party). The ones based on the Shamir's secret sharing scheme [109], as the presented in [106], for the creation of the private key and the distribution of the shares. Others based on the scheme introduced in [110], like the ones presented in [111], do not require a TTP during the bootstrapping process but assume that there exist an offline CA that issues long term certificates to each user. Also, they are not scalable. All the identities must be known a priori (each user is assumed to have a certificate assigned by an offline CA), and the threshold values (n, t) are prefixed and should be chosen carefully because re-keying is very costly. Finally, the operational cost of the scheme is very high in both computational and communication overhead. Several rounds of communications and complex mathematical operations are needed to admit a new user. On the other hand, the fully

distributed threshold schemes have an even worst performance and also, without a synchronous broadcast system, it is not clear if fully distributed secret update and redistribution protocols are possible while defending against all known attacks [112].

Identity Based Access Control

Instead of using the typical PKI (Public Key Infrastructure) scheme based on certificates, the identity-based cryptography ID assignment scheme presented in [113] is based on the idea of identity-based cryptosystems first presented in [114]. Within this scheme, three different proposals are described: one based in an external TTP, other based in a centralized bootstrap peer and another in a bootstrap peer that delegates some administrative task in trusted assignor peers. In the three proposals users are identified by their IP and assigned a random ID by one of the administrative entities, depending on the proposal. This scheme improves the IP based access control by making the ID not be straight deductible from the IP but it heritages all its others problems. Also, it adds the drawbacks of having to include centralized authorities in the scheme.

Social Networks

The paper related to SybilGuard [115] presents a mechanism based on social networks to limit the effect of *Sybil Attacks*. It is based on the fact that Sybil nodes (that depend on the same attacker) are usually strongly connected ones to each others but weakly connected to rest of the (honest) network. A random walk algorithm is used to detect these anomalies in the nodes' social links and limit the number of Sybil nodes in the network. An improved version of SybilGuard, called SybiLimit was described later in [116]. In [117] a similar algorithm is presented whose authors claim to outperform the two described before. However, these algorithms are only measures that limit the impact of *Sybil Attacks* and their possible effectiveness in real environments is not well documented.

Shared Secret

As an alternative for small groups that wish to form a private network without complexity, the RELOAD protocol [57] also presents the possibility of controlling the access to the network by means of a shared secret key, used to key TLS-PSK [91] or TLS-SRP modes [92]. Thus, a user who does not have the key cannot establish a TLS [41] or DTLS [90] communication with the other members of the network and hence access to it. This method, usually used in ad-hoc networks [118], limits the number of entities (users) that can access the network but not the number of identities they can have.

Centralized Control

To establish defenses against the *Sybil Attack* becomes easier when ID assignment is based in a centralized certificate authority [70]. One option presented in [65] is to bind node-IDs to real world identities. This can be done easily in the private networks of companies or universities requiring the user to introduce her credentials in order to get her node-ID and certificate to access the network. In more open nature networks, some kind of unique identification as the passport number, driving license ID, a limited e-mail address or credit card number (to note some) can be used. Another approach also suggested in [65, 66] and [96] is to charge a minimal cost for each certificate. Thus, while the certificates may be very low in cost, obtaining the amount necessary to corrupt an overlay would be very costly. RELOAD's draft [57] presents an alternative based on the typical username/password access control. But this time, instead of having to provide their credentials each time they access the network like with the login server of Skype [82], users only have to do it once to receive the certificate grating them access to the system.

Trusted Devices

Access to the network may be also restricted by requiring each user to have a specific trusted hardware device that grants her identity in the system, as suggested in [119] and [120]. This is a good countermeasure against external intrusions and *Sybil Attacks* in private networks but not very suitable for large networks due to the cost of providing each user with a trusted device.

2.1.3 ID Collision Attack

The last factor that should be taken into consideration in the ID assignment process is to prevent duplicate node-IDs [94]. This is very unlikely to happen when the node-ID is derived from the hash of the IP address or the public key of the node, but not impossible (and very difficult to prevent if it happens) in a very large P2PSIP system. This concern can be easily solved when a central authority is involved in the assignment (CA, bootstrap server) by checking that the new assigned ID was not given out before.

In cases where no central authority is involved in the ID assignment users should check while entering the network that other user does not already have their same node-ID, and, if that happen; calculate a new one. However, this check does not prevent an attacker to, once performed the check, keep on using the collided node-ID to impersonate other user.

Table 2.2: *Bootstrapping Summary of Attacks and Defenses.*

BOOTSTRAPPING			
Attacks	**Security Requirement at Risk**	**Defenses**	**References**
Fake Bootstrapping A.	Confidentiality	Peer-Caches	[57] [81] [121] [122]
	Integrity	Random Address Probing	[60] [65] [122] [123] [124]
		Network Layer Mechanism	[60] [65] [122] [123] [125] [126]
		Out of Band Mechanisms	[27] [123] [127] [128] [129]
		Centralized Bootstrapping	[27] [53] [57] [122] [123] [130] [131] [132]
	Availability	Global Bootstrap Service	[133] [134] [135]

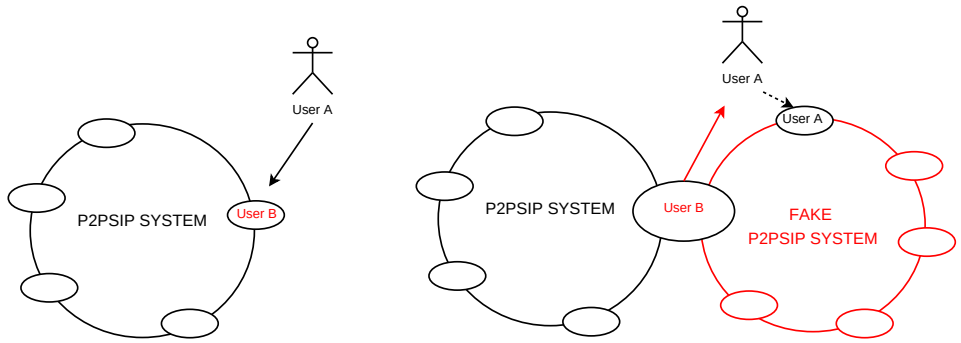
2.2 Bootstrapping Scheme

Each time an authorized user wants to operate and access to the resources of a P2PSIP system, she needs to bootstrap into the network. A secure and efficient bootstrapping mechanism is crucial to the security of a P2PSIP system, since without it the users' access to the network could be denied or maliciously modified to monitor their actions and/or prevent their access to certain resources.

Following, we present the main attack that can be launched to disrupt this process and the bootstrapping techniques presented in the literature to prevent it. Table 2.2 summarizes these defenses.

2.2.1 Fake Bootstrapping Attack

Before entering a P2PSIP system, every new participating user has to contact a user that is already member of it in order to initialize her state (Figure 2.5a). If the contacted user (bootstrap user) is malicious, she can completely corrupt the view of the P2PSIP system as seen by the new user [65]. Also, a malicious bootstrap user may connect the new user to a different fake system formed by other faulty users [60] (Figure 2.5b). This way, the attacker may monitor the victim's behavior (with whom she is trying to establish multimedia communications) or serve her with fake content (negate the presence of online users, give fake user's contact info to it, etc.). Following, we present the different bootstrapping mechanisms described in the literature and used to prevent the *Fake Bootstrapping Attack*.



(a) New user A contacts malicious user B in order to bootstrap in the P2PSIP system.

(b) Malicious user B redirects user A to a fake P2PSIP system formed by other faulty users.

Figure 2.5: *Fake Bootstrapping Attack.*

Peer-Caches

One approach to bootstrap is the use of peer-caches. If the user has already been in the network, each time she leaves the system she stores the address of all the peers she has contacted during the session in a cache. Next time the user wants to log in the network, she tries to contact one of the nodes in the cache in order to bootstrap. This mechanism was suggested for Gnutella [121] and later included in Skype [81] and RELOAD [57]. The study [122] shows that this mechanism works well with short periods of disconnection, but should be combined with another methods in order to bootstrap after long periods or for the first time.

Random Address Probing

New joining nodes may try to discover bootstrap peers by randomly probing to connect to different IP addresses (using a well-known port) within the address space until they find a node of the network [123]. For this solution to be suitable, the number of users of the system should be large enough so that the probability of finding a node of the network would be reasonable. As noted in [122] and [123], this approach does not work well with nodes behind a NAT or a firewall, and must be applied with care to avoid false security alerts since it uses the same propagation technique than some viruses and worms. Also, an extra security mechanism is needed to ensure that the contacted nodes are not malicious. The paper [65] recommends to contact a large number of bootstrap nodes to ensure that at least one is correct. Also, that research highlights the importance of using certificates to prevent malicious nodes to forge the node-IDs of trusted ones. An extra measure is presented in [60] that recommends to cross-check routing tables with other randomly chosen nodes in order to test the correctness of the bootstrapping process. The

proposal described in [124] presents an improvement over this approach that, using a heuristic that selects the most promising IPs, reduces the number of probes. Its main drawback is that a statistical profile of the nodes of the network is needed in order to create the heuristic, and such a profile may not be available.

Network Layer Mechanism

One possible approach is to create a multicast group for bootstrapping purposes [123]. All the nodes of the network join this group once they log on the system. This way, all a new joining node has to do to find candidate nodes for bootstrapping is to send a request to the group. Another similar approach is to send anycast messages querying for nodes of the overlay [125] or to use the SLP (Service Location Protocol) [126]. These protocols facilitate the bootstrap process, but the information stored at the multicast or anycast routers and at the central SLP directory services raises scalability and robustness (availability of the central directory services) questions [122]. If anycast is used or there is no restriction on the nodes that can join the multicast group, an extra measure, like the use of PKCs [60] and [65], should be implemented to authenticate the contacted nodes and prevent the attack.

Out of Band Mechanisms

In the first implementations of Gnutella [27] the address of active peers was exchanged through IRC (Internet Relay Chat) [127]. Later, due to the limitations of this mechanism, they implemented the use of web caches [128]. Nodes within the network periodically report suitable bootstrap nodes to HTTP-based caches. New joining users contact this web cache before entering the network in order to get the IP address of bootstrap nodes. The main problem of this approach is the web caches' content getting outdated very quickly as noted in [123] and [129] plus the difficulty to ensure that the bootstrap nodes added to the web cache are not malicious.

Centralized Bootstrapping

Several static nodes (servers) are placed within the network. These nodes collect information about the topology of the network and act as bootstrap nodes for the users intending to access the system. Static bootstrap nodes' contact information may be hard-coded within the user application [27], resolved from a DNS, [53] and [130], or a DDNS (Dynamic Domain Name System) [131]. The main problem of this approach is its scalability [123] and the bootstrap servers being a single point of failure [122]. Even if several bootstrap nodes exist, they may become subject to DoS attacks, as happened when the login nodes of Skype were overloaded [132]. On the other hand, if new nodes are allowed to become bootstrap servers dynamically as the network size raises, an extra control system should be implemented to ensure that the new

bootstrap servers are not malicious. The RELOAD protocol [57] presents an hybrid alternative, where once a user contacts the enrollment server in order to receive her user credentials to access the system, she also receives a multicast group of bootstrap peers and the IP address of some of them.

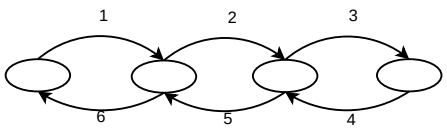
Global Bootstrap Service

Another approach presented in the literature, [133] and [134], is the implementation of a global bootstrap service. The idea is to merge the bootstrap information of several P2P networks and applications in a single global P2P bootstrap network. In order to access the desired application, a new user joins the global network and retrieves the specific bootstrap information of the service she wants to access. This approach shifts the problem of joining the P2PSIP system to joining the P2P bootstrap network [135]. The fact that the number of users of the global network is the sum of users of all the applications make feasible to use methods like random address probing to access it. However, authentication methods should be used to prevent a coalition of malicious nodes from creating a bogus bootstrap network. Also, it is not clear the efficiency this approach would have in the real world due to the overhead created for such an amount of nodes and information concentrated in the same network.

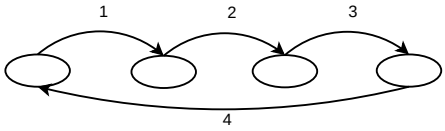
2.3 Routing Scheme

Message routing is one of the essential features a DHT based P2P overlay provides to communication systems using this kind of networks. The efficiency and correctness of the routing mechanism are of great importance for the behavior of a P2PSIP system.

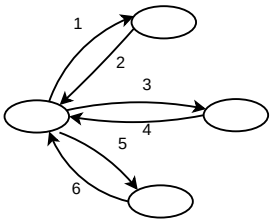
Routing can be performed in two different ways, each one having its advantages and drawbacks. In recursive routing, the source of the query forwards a request to the node closer in its routing table to the objective. The process is repeated by nodes receiving the query until it reaches the desired target. Once the destination is reached, the response can be sent back in two different ways: symmetric and asymmetric. Symmetric responses, Figure 2.6a, follow the same path in reverse than the queries they are related to, while asymmetric ones may follow a different path (a direct response is also possible, Figure 2.6b) to reach the originating node of the query. A mechanism that accumulates a history of the peers a query passes through, as the one described in [57], is necessary to reply symmetrically. The advantages of recursive routing are not requiring the sending or receiving peers to have a rich set of connections to other nodes in the overlay, reducing the total number of messages transmitted and its feasibility to be used with NAT networks. Its main drawbacks are the amplification of possible DoS attacks and the little control the initiating node has over the routing process.



(a) Symmetric Recursive Routing.



(b) Asymmetric Recursive Routing with Direct Response.



(c) Iterative Routing.

Figure 2.6: *Routing Types.*

In iterative routing, Figure 2.6c, the nodes receiving the query, instead of forwarding it, reply directly to the originating node with a closer node to the target. Then the initiator sends a new request to the new recommended node. The process follows until the target is reached. The advantages of iterative routing are that it consumes fewer resources for the intermediate peers (they only have to send redirect messages rather than forwarding requests and responses) and gives more control to the source node over the routing (offering robustness to some security attacks, described later in this section, that recursive routing is vulnerable to). Its main drawbacks are low global performance (greater number of total messages) and not working well with nodes behind a NAT. For further details on the advantages and drawbacks of the different routing types we refer the reader to [96] and [136].

The attacks that can be launched against the routing mechanism of a P2PSIP system and the defenses discussed in the literature to prevent them are discussed below. Table 2.3 summarizes the rest of this section.

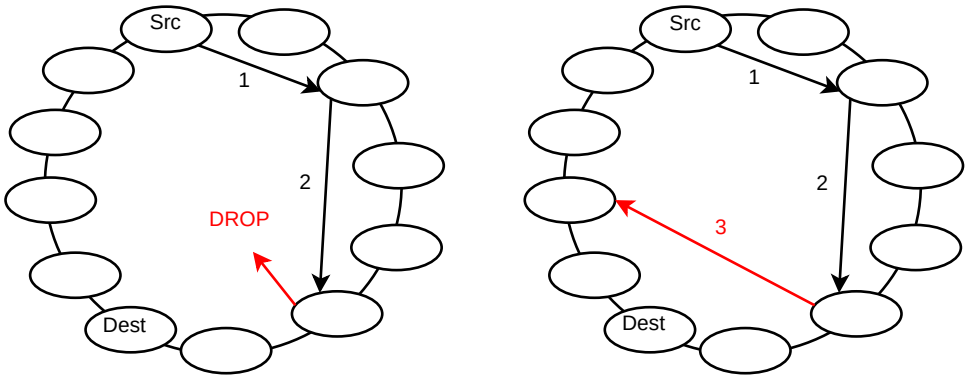
2.3.1 Incorrect Routing Attack

The routing primitive is necessary to perform almost all the main operations of a P2PSIP system: publishing or updating the users' contact information, leaving a voice-mail message intended for an offline user, retrieving the contact information of other users to start a multimedia communication with them or joining the system (finding the space of the network the new user's node should be inserted in). In the absence of faults, a message should be delivered to its destination after an average of h routing hops (the value of h depends on the specific DHT overlay used and the size of the network). But routing may fail if any of the *h-1* users along

Table 2.3: *Routing Summary of Attacks and Defenses.*

ROUTING			
Attacks	**Security Requirement at Risk**	**Defenses**	**References**
Incorrect Routing	Availability	Hop Testing	[65] [94]
		Alternate Routing Path	[61] [137]
		Parallel Routing	[52] [53] [65] [72] [138]
		Trust Diversity Routing	[69]
		Social Trust Routing	[139] [140]
Fake Routing Updates	Availability	Flexible Routing Tables	[54] [55] [60] [65] [141]
		Constrained Routing Tables	[51] [53] [60] [65]
		Mixed Routing Tables	[65] [67] [74]
		Periodic Routing Tables	[100]
		Cross Checking Tables	[60]
Man in the Middle	Confidentiality Integrity Availability	Encryption & Digital Signature	[15] [41] [42] [57] [60] [90] [142] [143] [144]
Replay Attack	Availability	Message's Identifiers	[57] [98] [145] [146]
Communications Log	Confidentiality	External A. Systems	[28] [147] [148] [149] [150] [151] [152] [153] [154]
		Routing Variations	[155] [156] [157]
		Headers Obfuscation	[28] [57] [157]
DoS Flooding	Confidentiality Availability	Balancing Techniques	[158]
		Pricing Techniques	[101] [159] [160] [161] [162]
		User Authentication	See Section 2.1

the route between the source and the destination are faulty (it may be also possible that the destination user of the message herself be faulty, however this concern can not be solved at the routing layer but at the storage layer as described later in this paper). Therefore, as presented in [65], the probability of success of the routing primitive when a fraction $\int$ of the users of the system are faulty is only $\left(1 - \int\right)^{h-1}$. This misbehavior of users dropping messages (Figure 2.7a) or routing them to the wrong place (Figure 2.7b) is what we call the *Incorrect Routing Attack*. Below, we present the different routing techniques used to prevent this attack.



(a) Intermediate node drops the message.

(b) Intermediate node routes the message to a wrong destination.

Figure 2.7: *Incorrect Routing Attack.*

Hop Testing

When iterative routing is in use, it may be possible for the sender to pick an alternative next hop when it fails to receive an entry from a node or to check whether the recommend next hop in a route is correct [65]. This test is based on the properties of the DHT routing algorithms that should answer every time with a closer node-ID to the objective and can be performed easily by checking that the next hop in the route is closer to the target than the previous one. Unfortunately, this check does not prevent a coalition of attackers to slow down the lookup process (replying with a closer node-ID to the target, to pass the test, but not the optimum one) and it is useless if one of the attackers can place itself close enough to the target, as described in [94].

Alternate Routing Path

In [61] three factors are identified that determinate the performance of the routing algorithm in the presence of malicious nodes: existence of multiple alternate paths between any two identifiers, cost of routing along alternate paths between any two identifiers, and ability to detect incorrect routing. Based on these assumptions, that paper presents a mechanism to defend the overlay against incorrect routing attacks. It works as follows: once the initiator node n of the query receives a reply from a node m claiming to be the responsible for a specific key, it checks that the node-ID of m is reasonably close to the resource-ID, that is, $dist(ID(m), k) < thr$, where threshold thr is configurable. If it is not or the reply is not received after a certain timeout, n launches the query again using an alternate path to the target. Simulations show a significant improvement in the probability of successfully routing a message using this mechanism. Authors

also suggest using a result cache to improve the performance of the routing primitives. The main difficulty of this approach is finding a threshold that minimizes the number of false positives and negatives. This technique can be used in conjunction with another algorithm modifications, such as BiChord [137], that add extra possible routing paths.

Parallel Routing

In [65] redundant routing is presented as a solution for incorrect routing. Their approach is divided into two phases. First, a node routes a message using normal recursive routing. Afterwards, a failure test is launched to detect whether the routing was correct. If it was, the routing algorithm ends successfully; if not, redundant routing is used to send the message again using diverse routes. The failure test, described in the context of Pastry [54] and assuming storage replication, is based on the assumption that the density of node-IDs is similar in the whole network, and checks that the density of node-IDs in the neighbor set of the sender is similar to the density of node-IDs close to the replica roots of the destination key. The main drawback of this approach is the lack of accuracy of the failure test presented.

A more aggressive approach is used in Kademlia [52]. Kademlia implements parallel routing by iteratively querying α nodes for the closest k nodes to the target, where α and k are system-wide concurrency parameters. In each step the returned nodes are merged into a sorted list from which the next α nodes are picked. S/Kademlia [72] extended this algorithm by making the parallel queries to use disjoint paths. This mechanism increases the probability of a successful routing but also floods the network with unnecessary messages. Another version of parallel routing using iterative queries is implemented in Epichord [138].

A different option is to use multiple replicated networks, such as CAN's realities [53]. Users have a different node-ID in each network and therefore each request follows a different path in each reality.

Trust Diversity Routing

The paper [69] presents an alternative based on the Chord iterative strategy with some modifications. The main difference is that every polled node returns its whole routing table instead of the closest node to the key. Once the querying node receives all the alternatives for the next routing hop, it can use different strategies to select the one to be used. This paper presents results for the standard Chord query strategy (select the closest node in the ID space), trust diversity (nodes keep a history of nodes they previously queried and try to balance nodes they use in queries with the goal of not putting too much "trust" in single nodes, but rather spread the nodes used for routing over the time), mixed routing (sorting the nodes in base of their closeness and their diversity and using the best ranked) and zigzag routing (closeness routing

and trust diversity routing are used alternatively). Closeness strategy shows to be the best when the rate of malicious nodes is relatively small while zigzag routing outperforms the rest strategies with high percentages of malicious nodes. The main drawback of this approach is the increase in size of the messages exchanged and the possible insecurity of a node returning its whole routing table when it is polled (malicious nodes may use this information to try to monitor, deny or eavesdrop its communications).

Social Trust Routing

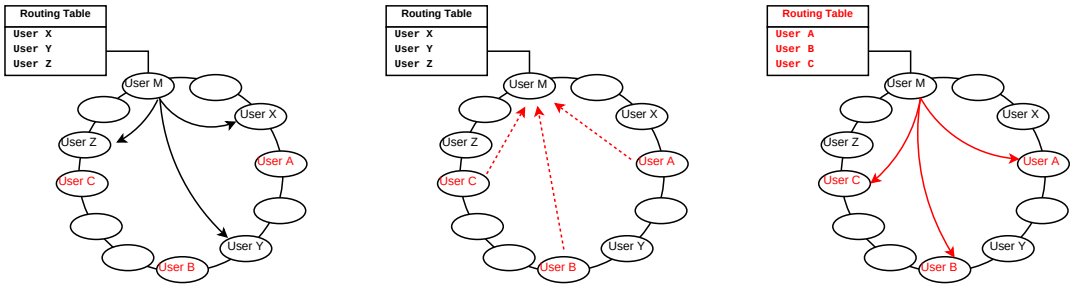
The approach presented in [139] and [140] uses social networks to take routing decisions. Trusted users from a social network are preferred for routing. This mechanism reduces the impact of this attack when is possible to route a message through trusted nodes but it usually also increases the number of hops needed to do it.

2.3.2 Fake Routing Updates Attack

In order to communicate with the rest of the overlay, each user participating in a P2PSIP system maintains a routing table formed by links to a set of peers called neighbors. All of a user's communications with the system are carried out through these links and, hence, the ability of a user to access the resources of the system is determined by the correctness of her node's routing table. As described in the bootstrap section, the routing table of a node is created when it joins the network. However, given the changing nature of the topology of the network, updates are also needed periodically to maintain the correctness of the table. These updates are done by contacting the set of neighbors in order to, among others, be sure they are still online or to discover new neighbors. Even when the routing primitive is securely implemented using one of the methods described in the section before, an attacker (or a coalition of them) could disrupt the routing operation of a node by corrupting its routing table with incorrect routing updates (Figure 2.8). In this way, the attacker (or attackers) could deny (by dropping them), monitor (by logging them) or delay (by using longer paths) the victim's attempts to establish multimedia communications with other users of the system or to update her state. This anomaly, that was first presented in [65] and [60], is what we call the *Fake Routing Updates Attack*. Below, we describe the different routing table types that can be used to prevent this attack.

Flexible Routing Tables

When a node receives a routing update, before inserting the new entry into its routing table, it must check that it fulfills the requirements of the system and also verify that the remote node is reachable [60]. Some DHT algorithms, like Pastry [54] and Tapestry [55], impose very weak constraints on the set of node-IDs that can fill each entry of the routing tables (especially at



(a) User M has a clean routing table with good behaving users X, Y and Z.

(b) Malicious users A, B and C send M fake routing updates.

(c) User M's routing table gets poisoned with malicious entries.

Figure 2.8: *Fake Routing Updates Attack.*

the top levels). This flexibility allows them to use methods to improve routing performance as Proximity Neighbor Selection (PNS) [141] but also makes very easy for an attacker (or a coalition of them) to fake the routing table of a well behaving node.

Constrained Routing Tables

DHT networks like Chord [51] and CAN [53] impose strong constraints on their routing tables' nodes: they need to have the closest node-ID to same point in the ID space. This alternative cannot take advantage of proximity routing but it makes harder for an attacker to fake the routing table of a node.

Mixed Routing Tables

A mixed solution is presented in [65]. Their proposal uses two routing tables: one that exploits network proximity information for efficient routing and another that constrains routing table entries. In normal operation, the first routing table is used to forward messages to achieve good performance. The second one is used only when the efficient routing technique fails. Following a similar approach, the paper [67] presents a new constraint based on PNS: only nodes with minimal network delay are selected as neighbors. This way, in order to fake the routing tables of a node, the attackers need to be geographically close to it. Unfortunately, as noted in [74], this defense assumes that the delay measurements cannot be manipulated by an attacker. Also, the lack of accuracy of the measurement in large-scale networks like Internet makes it only effective for small networks.

Periodic Routing Tables

A completely different approach is the one followed in [100], already described in Section 2.1.1, that uses periodic resets of the nodes' routing tables and a rate limitation of routing updates to prevent an attacker to completely fake the routing table of a node. Unfortunately, as already noted, this technique introduces a high latency due to churn.

Cross Checking Tables

An extra measure that can be implemented in conjunction with any of the methods presented before is the one described in [60] that recommends to cross-check routing tables with other randomly chosen nodes in order to test its correctness.

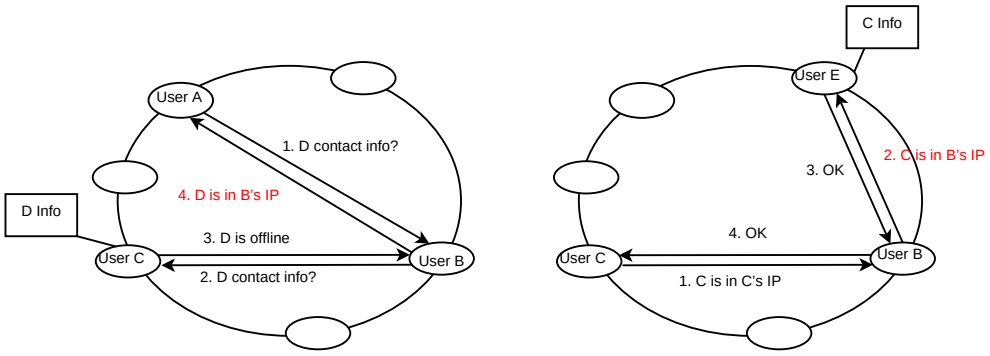
2.3.3 Man in the Middle Attack

The particular routing mechanism used by P2PSIP systems, where a user relies on other users to access the resources needed to establish multimedia sessions, gives the possibility to an intermediate user of monitoring, modifying or inserting fake replies in order to impersonate either the source or the destination of a communication. This is commonly known as a *Man in the Middle Attack*. Figure 2.9 presents two examples of the application of this attack: in Figure 2.9a a malicious intermediate user B modifies a reply from user C to a request from user A about the location of user D, to impersonate user D; while in Figure 2.9b a malicious intermediate user B modifies a contact info update request from C intended for user E, to impersonate C. In the first case, the attack only affects one user, but in the second all the users of the system are affected by the impersonation. Below, we present the most commonly used defense to prevent this attack.

Encryption & Digital Signature

The most widely accepted prevention of information tampering is the use of digital signatures [142]. Assuming that every user in the system holds a unique pair of public/private keys, each user signs messages using her private key before sending them. This way, the addressee can check the integrity of the message and the identity of the sender. Authentication of messages using digital signatures ensures than an attacker can not modify or insert a fake message in the network, but does not prevent her from accessing the message's content. In order to obfuscate the content of the message, the sender also has to encrypt [15] it using the destination node's public key. An approach that combines both signing and encryption are digital signatures schemes with message recovery [143, 144].

The special properties of routing in peer-to-peer networks suggest a clarification of how



(a) Malicious intermediate user B modifies C's reply about the location of D to impersonate her.

(b) Malicious intermediate user B modifies C's contact info update request to impersonate her.

Figure 2.9: *Man in the Middle Attack.*

this mechanism should be implemented here. In typical Internet routing, security is normally ensured end-to-end, i.e. the sender encrypts and signs the message and delivers it to the receipt using some specific protocol for this task as TLS [41], DTLS [90] or IPsec [42]. Unfortunately, this approach is not valid in peer-to-peer networks, because the intermediate hops need to access to some information of the message in order to route it properly. Therefore, the routing protocol needs to implement two features: it must separate the routing information (needed for the intermediate nodes to route the message) from the content of the message per se (that must be only accessed by the addressee). Also, it must permit to use both hop-by-hop and end-to-end security. As we can see in Figure 2.10, first the sender encrypts and signs the content of the message with the public key of the ultimately receiver and the sender's private key respectively (end-to-end security at the application layer), then the sender appends to it the routing information and encrypts and signs the whole message for the first hop using TLS, DTLS or IPsec (hop-by-hop security at the network/transport layer). This way, every hop can check and modify the routing information of the query in order to properly route it, but only the receiver can see its content. An example of a P2PSIP protocol implemented both features is RELOAD [57].

As a final recommendation against *Man-in-the-Middle Attacks*, note that both encryption and digital signature should be implemented together to achieve the desired security. As already presented, digital signature alone can not prevent an attacker from accessing the content of a message. Alike, when only encryption is used, an attacker could insert fake responses in the network. As a clarification of the second statement we present here the example described in [60]: Consider an iterative lookup process where the querier Q is referred by node E to node A. Node E knows that Q will next contact A, presumably with a follow-up to the query just

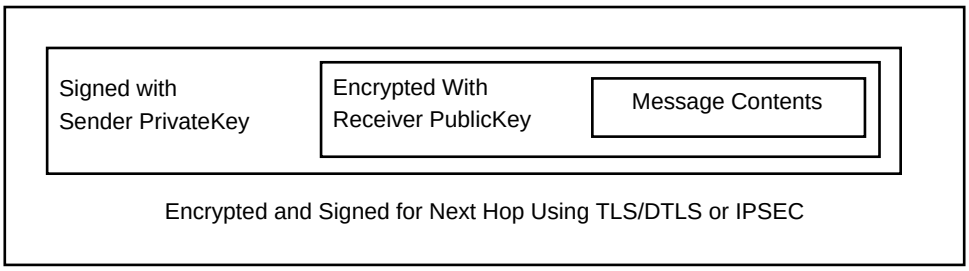


Figure 2.10: *Hop-by-Hop and End-to-End Security of Messages.*

processed by E. Thus, E can attempt to forge a message from A to Q with incorrect results.

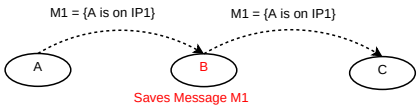
2.3.4 Replay Attack

A *Replay Attack* is a form of network attack in which a valid data transmission is maliciously or fraudulently repeated or delayed. Replay attacks have been widely discussed in the literature [145], mainly within the context of cryptographic protocols.

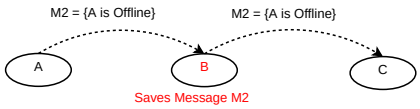
In the specific case of P2PSIP routing, the threat relies on a malicious node capturing an older message sent by other node and resending it to the overlay, replacing any newer data with the old information present in the replayed message [98]. The typical use of this attack is to denial the contact with a specific user of the network by replacing her actual contact information by an old (and not valid) one. To sign and to encrypt the message, in order to authenticate it and to grant its integrity and confidentiality, it is not enough to prevent this attack; as the following example illustrates.

Imagine a user A that joins the network from an IP address IP1 and, in order to be contactable, stores her contact information in the overlay. The store order message (signed by A and encrypted with its destination public key) traverses several nodes of the network (among which is the node B, that saves internally the forwarded message M1 despite the fact that B cannot read or modify it) until it reaches its destination C (Figure 2.11a). From that moment, every user could know that user A is contactable at IP1 by contacting C. After using the network for a while, user A logs off. The logout message (signed by A and encrypted with its destination public key) traverses several nodes of the network (among which is the node B, that saves internally the forwarded message M2 despite the fact that B cannot read or modify it) until it reaches its destination C (Figure 2.11b). Suppose now, that after sometime, user A connects again to the network; this time using a different IP address IP2; that again registers as her contact information with C (Figure 2.11b). If now node B resends the message M1 or the message M2 to C it would set the contact information of A back to IP1 or set A offline, making A uncontactable by the rest of users of the network.

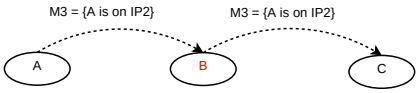
In our example, we assume that the second time A joins the network the node responsible



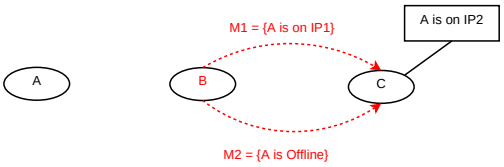
(a) A register herself in IP1, B saves A's message M1.



(b) A leaves the system, B saves A's message M2.



(c) A log in again, this time at IP2.



(d) B replays either M1 or M2 to make it appear that A is either in other location or offline.

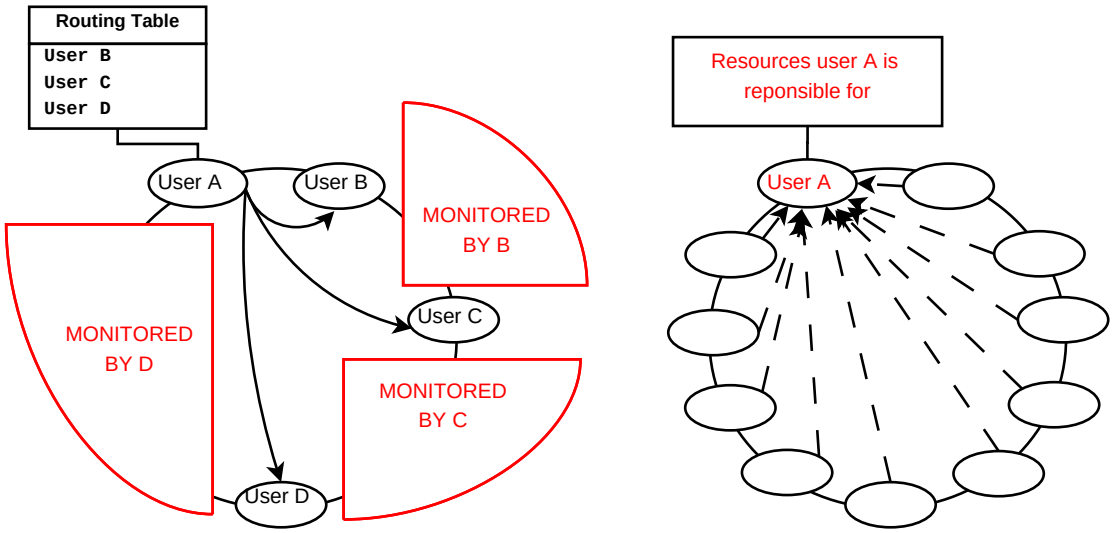
Figure 2.11: *Replay Attack.*

for A's resources would be C again. This may be not very probable, but serves to illustrate the worst case where a signed and asymmetrically encrypted message can be replayed. However, it is not very usual to send a store order message with contact information publicly accessible encrypted due to its inefficiency: you need two messages, the first one to discover the node responsible at a specific time of the resources (to be able to encrypt the message for it) and the second one with the information encrypted; and it does not make much sense to encrypt an information that will be publicly accessible. Said this, it is clear we need an extra protection to defend the network against this kind of threat.

Message's Identifiers

The most common defense against replay attacks is the inclusion of identifiers in the messages. Mainly freshness identifiers (nonces), as timestamps, counter values, or random values [146]; and identity identifiers, as the sender and the recipient of the message.

The Internet-draft of RELOAD [57] proposes the inclusion of a counter (local time of the sender since the Unix epoch plus a lifetime) in each sent message to prevent this attack. Any message from a node with a counter before the last message received from that node is discarded. In principle, this mechanism does not require synchronized clocks; the receiving peer uses the counter of a previous message for the comparison, not its own clock. Nevertheless, if a counter from a previous connection is not available, it forces a comparison with the local timer, that may not be accurate.



(a) Areas of the ID space where the users of A's routing table can monitor her activities.

(b) User A can monitor any access over the resources she controls.

Figure 2.12: *Communications Log Attack.*

2.3.5 Communications Log Attack

The special nature of routing in P2PSIP networks allows other users of the system to easily record the activities of one user within the network. Since all of the user A's communications are established through the fingers in her routing table, these fingers could monitor the list of the users contacted by A (Figure 2.12a). Also, since users' contact information are stored as resources in the P2P network, the responsible for storing the contact information of a specific user can make a profile of the users trying to contact her (Figure 2.12a). The ability of an attacker to record these activities is what we call *Communications Log Attack*.

The defenses against this attack are closely concerned with anonymous systems, specifically with sender anonymity, i.e. the untraceability of a user accessing a specific resource of the network, in contrast with recipient anonymity (to hide the user responsible for a specific resource of the network). Below, we describe the different approaches presented in the literature to combat this attack.

External Anonymous Systems

One of the first papers on anonymity was [147] by D. Chaum in 1981 that allows an electronic mail system to hide who a participant communicates with through the use of mixes (nodes hiding the correspondences between their input and output messages in a cryptographically strong way). From this schema several protocols and applications to protect anonymity have

emerged, among which stress GNUnet [148] (using the GAP protocol [149]), Tor [150] (using onion routing [151]), Freenet [28], Tarzan [152] or MorphMix [153]. For a specific analysis of the characteristics of each one of the systems, we refer the reader to their original papers[4]. An example of a P2PSIP anonymous system based on onion routing is presented in [154]. It improves the anonymity of the system but at a high performance cost.

Routing Variations

The study [155] analyzes the anonymity of the Chord protocol and concludes that the implemented recursive routing algorithm provides a high degree of sender anonymity against passive observers. It also shows that using data or location caching, and a larger successor list increases the anonymity.

The proposal described in [156] goes a bit further and compares the anonymity of different alternatives with the original recursive routing algorithm of Chord:

- *Random Recursive Routing:* In this variation, peers forward the message at random to whatever finger is closer to the destination, instead of routing messages to the closest finger to its destination. Random Recursive Routing improves the anonymity of Chord, but unfortunately it also increases the path length (the average case from $(\log_2 N)/2$ to $(\log_2 N)^2$, and the worst case from $O(\log_2 N)$ to $O(N)$).

- *Weighted Random Routing:* Instead of picking the next forwarding hop at random from the closer fingers, fingers are weighted and picked with different probabilities, as for example 1/2 for the closest, 1/4 for the second closest, and so on. In this case, the average path length is only $\log_2 N$ but the worst case is still $O(N)$. Its degree of anonymity is nearly as good as the random one, which suggest that it may be a suitable compromise between performance and anonymity.

- *Indirect Routing:* In this routing algorithm, when a peer wants to send a message, instead of routing it directly to its destination, it chooses at random an intermediary peer in the network to route the message in its behalf. Also, the query to the intermediary is secured using an m-of-n secret sharing scheme, i.e. the message is split in n shares sent using independent routes and at least m shares (of n sent) need to be captured in order to reconstruct the message. This way, it very difficult for an attacker to know the truly

[4]It is out of the scope of this thesis to analyze all these systems (it is beyond doubt the relevance these analysis would have in the study of anonymity in P2PSIP networks, but the dimension of such an analysis would require a specific research on this subject) and in this section we intend to discuss only the defenses against the *Communications Log Attack* that can be implemented internally within the P2PSIP routing protocol and without using an external application.

destination of the query. Indirect routing improves the anonymity of the sender but it also increases the number of messages needed to route a query and its latency. A similar routing alternative is used in the AP3 system [157].

Headers Obfuscation

Besides the routing algorithm used, another factors should be taken into account in order to prevent the *Communications Log Attack*. It is important to not use header fields that may reveal information about the route followed by a message:

- Avoid setting fixed default values for Time-To-Live (TTL) counters. Alternative methods as the implemented in Freenet [28] may be used.

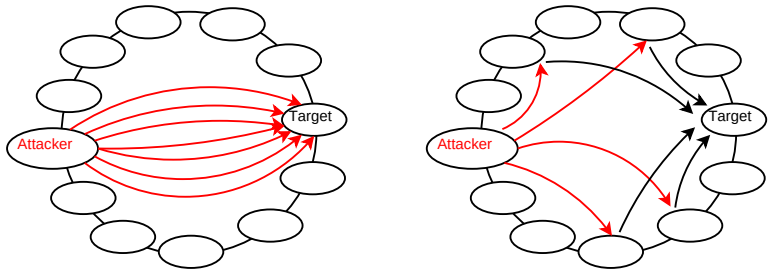
- Methods as the forwarding tables in AP3 [157] or the Truncated Via-Lists in RELOAD [57] should be used to obfuscate the information needed to route back the response of a query.

Finally, it is crucial the use of a secure ID assignment and an efficient access control (one of methods described in Section 2.1 should be used to achieve this) to prevent possible attackers to surround the user or the resource they want to monitor.

2.3.6 DoS Flooding Attack

One of the most famous, and difficult to defend, attack that can be launched against an information system is the DoS (Denial of Service) attack or its large scale distributed version DDoS (Distributed Denial of Service). The intention of a DoS attack is, as its name indicates, to prevent the victim or victims from accessing or providing services within the network. There are several papers analyzing DoS attacks, most of them related to network layer DoS attacks, as for example [163, 164, 165] and [166]. Nevertheless, in this section we will focus only on the application level attack that emerges due to the special routing mechanism used by the P2P overlay network of a P2PSIP communication system.

In a *DoS flooding attack*, an attacker, or a coalition of them, saturate the victim's resources by flooding her with queries. The attack can be launched sending directly the queries to the victim (Figure 2.13a) or using other innocent users to amplify the attack by, for example, routing the queries to the victim through them using recursive routing [95] or sending them queries with the victim node as source so that the replies from the innocent nodes flood the victim (Figure 2.13b). Following, we present some of the techniques described in the literature to prevent this attack.



(a) Direct DoS Flooding Attack.

(b) Indirect DoS Flooding Attack using recursive routing or fake requests with the victim as source.

Figure 2.13: *DoS Flooding Attack.*

Balancing techniques

In [158] different balancing techniques are studied to prevent DoS attacks based on query floods in Gnutella that, despite being a study over an unstructured P2P network, can be extrapolated to the context of a communication system based in a structured P2P network. The paper presents the idea of managing the query load by using a combination of three strategies:

- *IAS -Incoming Allocation Strategies-:* IAS determines how many queries a node should accept from each peer (node/client) per time unit. Two options are studied: *Weighted IAS* (the number of queries accepted from a particular incoming link is proportional to the total number of queries arriving on that link) and *Fractional IAS* (each node is given an equal fraction of query bandwidth).

- *DS -Drop Strategies-:* If the amount of queries received from a remote peer is bigger than its allocation, DS determines which queries are accepted and which ones are discarded. Four strategies are presented: *Proportional* (the probability of acceptation of a query is proportional to the number of times it is received), *Equal* (all the queries have the same probability of being accepted), *PreferHighTTL* (accept queries with the highest TTL), *PreferLowTTL* (accept queries with the lowest TTL).

- *Reservation Ratio:* States how much of the processing capacity of a node is reserved for local queries (communication of a node with its clients in a P2PSIP context), and how much for remote queries (normal communication among nodes in a P2PSIP context).

The study concludes that the combination of a Fractional IAS with an Equal DS or a PreferHighTTL DS presents the best results, minimizing the effects of a DoS attack.

Table 2.4: *Storage Summary of Attacks and Defenses.*

STORAGE			
Attacks	**Security Requirement at Risk**	**Defenses**	**References**
Unauth. Resource Access	Confidentiality	Local Control	[57] [167] [168]
		Cryptography	[15] [61] [109] [167] [168] [169]
		Dedicated Security Servers	[170]
Malicious Contact Publish.	Integrity	Digital Signature	[57] [171]
		Crypto SIP-URI	[97]
		Reputation Systems	[94] [97] [172]
Malicious Resource Mgmt.	Availability	Replication	[51] [52] [53] [54] [55] [173]
		Erasure Codes	[167] [174] [175] [176] [177] [178] [179] [180] [181] [182] [183]
		Hybrid Strategies	[178]

Pricing techniques

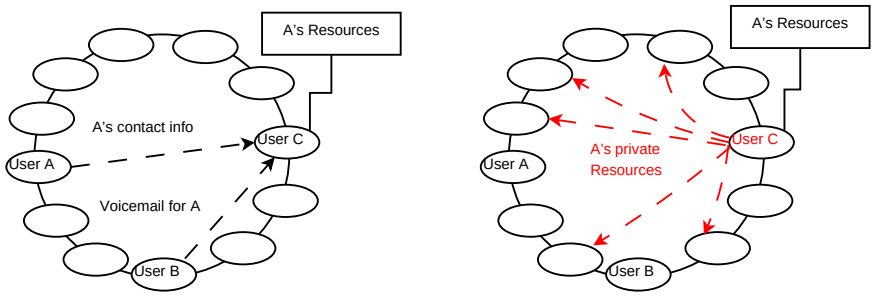
The pricing technique is used to limit the speed at which nodes send queries to other nodes of the network. When a node A sends a query to other node B in the network, B responds with a computational puzzle [101]. B will not process A's query until it receive a valid response to the puzzle. Several papers have shown that this technique achieves good results in preventing DoS attacks in several protocols, such as Tor [159] or TLS [160], and against network-layer DoS attacks [161] and [162].

User Authentication

One of the user authentication mechanisms described in Section 2.1 can be used to uniquely identify the sender of a query and discriminate malicious users, therefore, helping to defend against this attack.

2.4 Storage

A very important function of a P2PSIP network is to store the contact information of the users of the network in order to permit ones to communicate with each others. This storage scheme is also responsible for storing private and public users' resources such as voicemail messages, public certificates, etc. Following, we present the attacks and defenses described in the literature against the storage scheme. Table 2.4 summarizes the rest of this section.



(a) User A's resources (such as her contact info or a voicemail intended for her) are stored in a specific location of the network.

(b) The user responsible for A's resources leaks her private information.

Figure 2.14: *Unauthorized Resource Access Attack.*

2.4.1 Unauthorized Resource Access Attack

In DHT based SIP networks each peer is responsible for a specific part of the content in the network. These contents go from publicly accessible data, like a user's contact information, to private resources, such as a user's personal voicemail. Since the node responsible for storing these data is determined by the structure of the network and may not be trustworthy, a security mechanism should be implemented in order to prevent the storing peer or an attacker to access this data without authorization. This ability of an attacker to access the private data of a user stored within the network is what we call *Unauthorized Resource Access Attack*. Figure 2.14 presents an example of this attack where the responsible for user A's resources give access to them to all the users of the P2PSIP system regardless of whether they are public or private. Below, we describe the different approaches presented in the literature to combat this attack.

Local Control

In the RELOAD protocol [57] each resource identifier may contain multiple kinds of data identified by a kind-ID. The definition of each data kind specifies rules for determining which certificates can access each resource-ID/kind-ID pair, controlling this way the data access. However, if the node responsible for the resource is malicious, it can access the resource content or allow unauthorized users to access it. Another possibility, with the same drawbacks, is to use an ACL to determine the privileges of each user over an object like in OceanStore [167] or Fairsite [168].

Cryptography

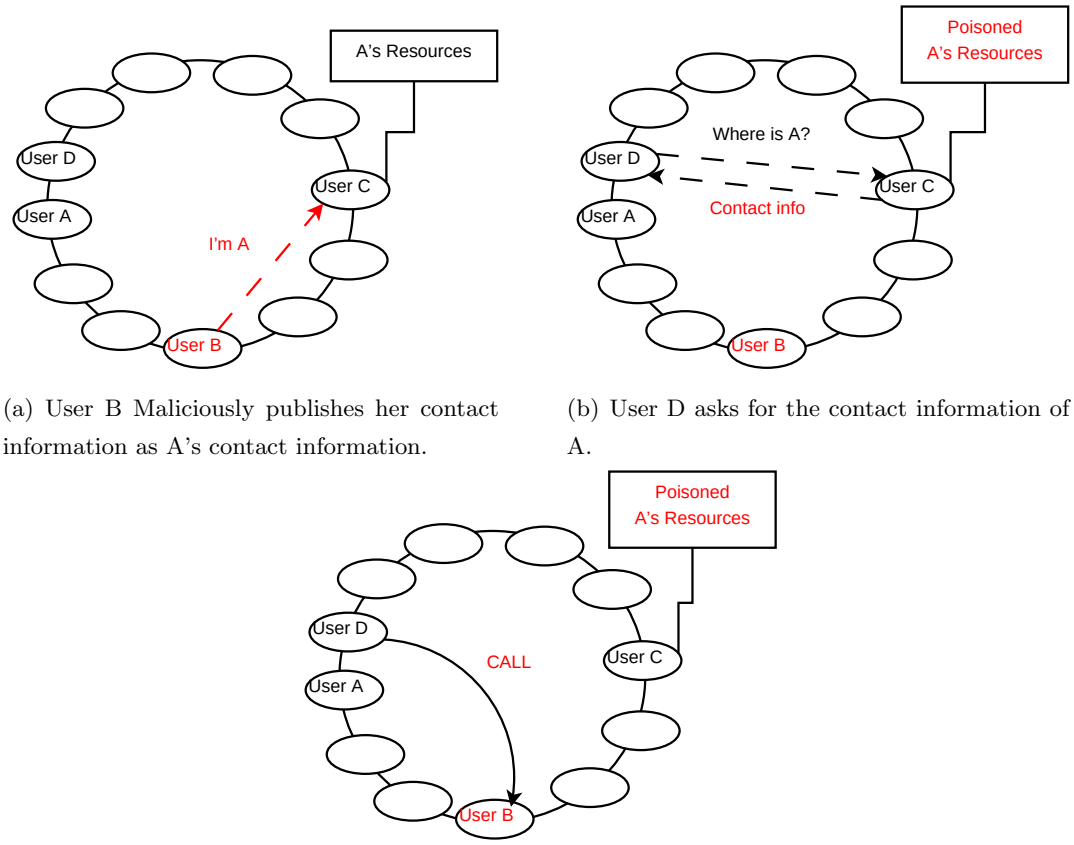
The more effective way to prevent malicious users from accessing the private data of other users within the network is to use cryptography. If a user wants to store a private resource for her

personal use, symmetric cryptography, such as the AES algorithm [15], could be used to encrypt the data before storing it in the network. On the other hand, if the private resource is intended to be accessed by other user, like a voice-mail, it may be encrypted using the public key of the recipient, as described in [169]. In case the publisher wants the resource to be accessible by a group of users, three possibilities arise: to extend the scenario of a single recipient by storing one copy of the resource for each recipient encrypted with her corresponding public key, to encrypt the symmetric key using the public key of all authorized readers and store the encrypted keys with the resource [168] or to store only one copy of the resource encrypted with a symmetric key and send a private message to each recipient with the location of the resource in the network and the key needed to access it [167]. In [61], authors present an alternative to the last presented option based on hiding the key used to encrypt the resource. Instead of sending the key to each recipient, the sender splits the key in r shares [5] (using the technique first described in [109]), store the shares at r random identifiers and distribute the random IDs to the nodes it wants to give access to the resource.

Dedicated Security Servers

Another solution is to add dedicated security servers, like VMS (Voice Mail Service) servers, to the architecture. The users rely on these servers for storing voice mail or other private resources. Unfortunately, as noted in [170], these servers reduce the advantage of the P2P architecture introducing an extra cost in its development, and issues such as load balancing and capacity problems. This paper also presents an alternative based in centralized security servers, but in this case the resources are stored in the overlay instead of in the servers. When a user A wants to leave an offline message, such as a voice-mail, to other user B of the system, first chooses a node in the network to store the resource using an heuristic, then generates an encryption key and encrypts the message, and finally stores it in the selected node. After that, A sends to the central server the name of the recipient B, the encryption key used and the node storing the message. Next time B logs on the network, she contacts the central server in order to check if there is any offline message for it, and if so, B retrieves from the server the location and the encryption key necessary to read it. This alternative reduces the cost of implementing the architecture and reduces the load and the required storage capacity of the servers. Nevertheless, it shares with the previous presented approach the problem of having in the central security server a single point of failure that must be contacted every time a user logs on the network.

[5] *The number of shares r is a configuration parameter.*



(a) User B Maliciously publishes her contact information as A's contact information.

(b) User D asks for the contact information of A.

(c) User D contacts user B thinking that she is A.

Figure 2.15: *Malicious Contact Information Publication.*

2.4.2 Malicious Contact Information Publication

The main task of the storage layer in a P2PSIP system is to keep the user's contact information (among other resources), being the authenticity of this information crucial to the smooth functioning of the system. If a user of the system could publish false contact information relative to other users (Figure 2.15a), she could establish her own device as the contact point of them (Figure 2.15b) and, therefore, impersonate them (Figure 2.15c), or publish erroneous info to prevent other users contacting them. In order to prevent this *Malicious Contact Information Publication Attack* a security mechanism should be implemented to authenticate the origin of a storage request received by a user of the network and that grants the authenticity of the information received in response of a retrieval request. Below, we present the defenses that can be used to prevent this attack.

Digital Signature

The most effective and suitable mechanism to authenticate the contact information (or any other resource) of a user is to use digital signatures. Assuming that all the participants in the system have a PKC assigned by a trusted authority (using one of the centralized mechanisms described in Section 2.1), when a user wants to publish her contact information (or any other resource) in the network signs it using her PrK before sending the storage request. This way, the node responsible for storing the resource can determine the authenticity of the storage message by checking its signature.

Similarly, when a user A retrieves a resource from the network, such as the contact information of other user B, A can be sure of the contact information authenticity by checking its digital signature. This is the mechanism implemented in the P2PSIP communication protocol RELOAD [57] and in older systems, like the storage peer-to-peer utility PAST [171], among others. Digital

Crypto SIP-URI

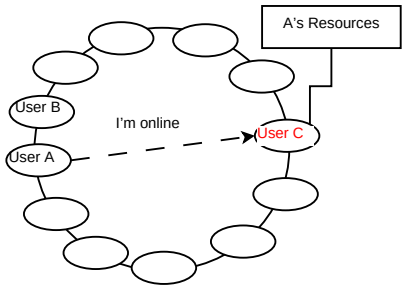
The paper [97] proposes a decentralized self certifying alternative to digital signature based on trusted certificates. This proposal uses cryptographically generated SIP-URIs as self certifying data for user registration and location lookup in the P2PSIP system. With such a solution, any user can generate a SIP-URI and sign binding updates. The authenticity of these binding updates can be verified by any entity in the network without relying on any kind of security infrastructure. The main advantage of this approach is its decentralized nature. Its main drawbacks are the readability of the SIP-URIs and that, as discussed in Section 2.1, self-signed certificates are not Sybil proof.

Reputation System

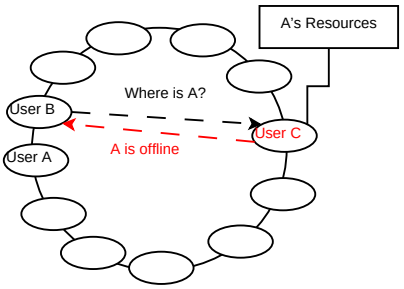
Reputation systems could also be used to assign different levels of trust to the information stored in the network. Nevertheless, actual reputation systems are focused on file-sharing applications [97], most of then relying on a central authority [94], and are not yet suitable for P2PSIP networks. For further information in reputation systems, we refer the reader to [172].

2.4.3 Malicious Resource Management Attack

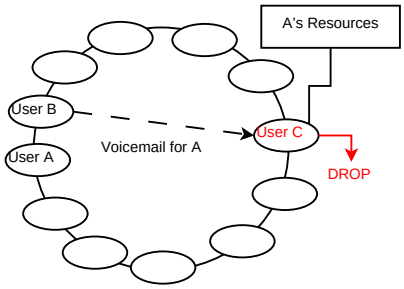
A node within a P2PSIP network may deny the existence of a resource of the system it stores and it is responsible for. Alike, it might claim to store data when asked, when in fact it does not and then refuses to serve it to the users of the network. This way, if no mechanism is implemented to avoid this *Malicious Resource Management Attack*, a node may, for example,



(a) User A stores her contact information in the network.



(b) User B wants to contact A but malicious user C negates A is online.



(c) User B wants to leave a voicemail for A, but malicious user C drops it.

Figure 2.16: *Malicious Resource Management Attack.*

prevent a user of the network from establishing a multimedia communication with other user (Figure 2.16b) that is logged in (Figure 2.16a) or receiving a voicemail (Figure 2.16c), even when the authenticity of the resources is granted using one of the methods described in the previous section. Below, we present the defenses that can be taken against the attack in question.

Replication

The most common countermeasure against this attack is replication, i.e. instead of storing a resource in a single location of the network, the resource is also stored in r different locations called replicas[6]. In this way, if the node responsible for a resource denies its existence, the requester can contact one or more replicas in order to access the resource to be sure that it really does not exist. Replication also prevents the network from losing data when a node leaves abruptly (due to an application or network error, a DoS attack, etc.) without passing the resources it is responsible for to the new responsible node. Several replication methods have been described in the literature:

- *Several Hashing Functions*: Instead of using one hashing function to determine the location

[6]The number of replicas r is a configuration parameter.

of the network where a resource should be located, several hashing functions are used. The number of hashing functions used determines the replication factor. This technique was first described in the context of CAN [53] and Tapestry [55].

- *Proximity Replication*: The r replicas are stored in the r nodes numerically closest to the location of the resource, as in Pastry [54] and Kademlia [52], or in the list of successors of the node responsible for the resource, as in Chord [51].

- *Multiple Realities*: Nodes belong to multiple replicated networks, called realities [53] but being located at a different point (and hence being responsible for different resources) in each one . This way, each resource is stored by a different node in each reality and the replication factor is determined by the number of them (realities).

- *Symmetric Replication*: The identifier space is partitioned into $\frac{N}{r}$ equivalence classes such that identifiers in an equivalence class are all associated to each other [173]. Each member of the class saves a replica of the resources stored by the other members of the group. Any such a partition will work, but for simplicity the original paper suggest to use the congruence classes modulo $\frac{N}{r}$. The number of members of each class, r, determines the replication degree.

Erasure Codes

Erasure codes allow to reconstruct an encoded object divided in n fragments using any m out of the n encoded fragments ($m < n$). The fraction of fragments required for decoding $r = \frac{m}{n}$ is defined as the rate of erasure [174]. Examples of architectures implementing erasure codes, like the Reed-Solomon codes [175] or the Tornado codes [176], are OceanStore [167] or TotalRecall [178].

The main advantage of erasure codes is that they can achieve the same level of availability as replication with less storage overhead [177]. Their main drawbacks are a bigger computational overhead of coding/decoding [179] and an increase of the complexity of the system maintenance [180]. Papers show that erasure codes are only practical for relatively large objects [181], high available nodes [182] and write intensive workload [183].

Hybrid Strategies

Hybrid strategies, combining both replication and erasure codes, are also possible. The paper [178] presents an example of their use based on a log file written in an append-only fashion. In this case, the head of the log is stored using replication to provide good performance. While the old entries are migrated into erasure coded representation to provide higher efficiency.

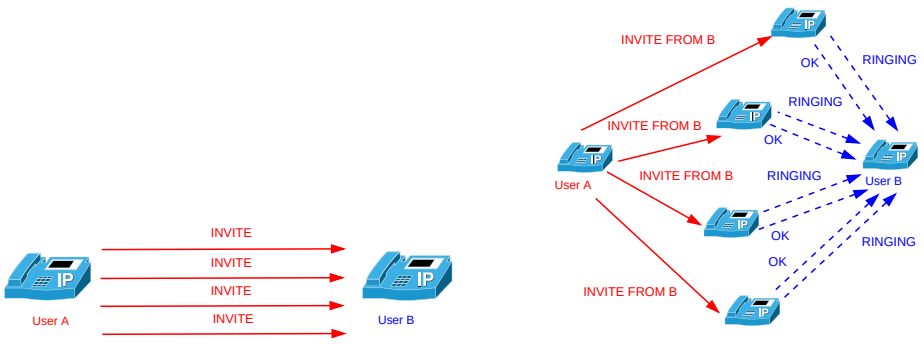
Table 2.5: *Communications Summary of Attacks and Defenses.*

COMMUNICATIONS			
Attacks	**Security Requirement at Risk**	**Defenses**	**References**
User Flooding Attack	Availability	Strong Authentication + Limitation	[85] [184] [185] and Section 2.1.2
		Stateless Barrier	[184] [186]
Parser Attack	Availability	Good Implementation + Size Limit	[11] [184] [187] [188]
		Extra security Modules	[189] [190]
Tampering with Message Bodies	Confidentiality Integrity Availability	Transport Security	[11] [191] [192] [193] [194] [195] and Section 2.1.2
SIP Session Modification	Confidentiality Integrity Availability	Signaling Authentication	[11] [191] [192] and Section 2.1.2
Media Session Alteration	Confidentiality Integrity Availability	Media Security	[187] [196] [197]
	Latency	Split Traffic	[191]
SIP Spam	Availability	Content Filtering	[44] [198] [199]
		Black Lists	[44] [198] and Section 2.1.2
		White Lists	[44] [81] and Section 2.1.2
		Gray Lists	[198] [200]
		Consent-based Communications	[44] [201] [202] and Section 2.1.2
		Reputation Systems	[44] [172] [203] [204] [205] and Section 2.1.2
		Address Obfuscation	[44]
		Limited-Use Addresses	[44]
		Turing Tests	[44] [103] [206] and Section 2.1.2
		Computational Puzzles	[101] [207] [208] and Section 2.1.2
		Payments at Risk	[44] [209] [210] [211]
		Legal Action	[212] [213]

Nevertheless, redundancy by itself cannot prevent this attack if a coalition of attackers can place themselves at the nodes of the network responsible of storing a specific resource and its replicas or erasure-coded blocks. In order to prevent this, one of the mechanisms of access control and secure ID assignment described in Section 2.1 should be implemented in conjunction with redundancy.

2.5 Communication scheme

The role of the communication service is the establishment of communications, mainly messaging, telephony and videoconferencing; using the SIP protocol in conjunction with the protocols involved in the media transmission (SDP - Session Description Protocol- [214], RTP -Real-time



(a) Direct INVITE Flood. (b) DDoS Fake INVITE Flood.

Figure 2.17: *User Flooding Attack.*

Transport Protocol- and RTCP -Real-Time Control Protocol- [215], SRTP -Secure RTP- [216] , etc.). Several documents [11, 187, 198, 217, 218] and [219] exist that analyze the security of these protocols. However, these analysis are related to the traditional architecture of SIP and some of the presented attacks are no longer valid or substantially change with the introduction of the new P2P architecture [85]. Examples are attacks related to the user registration (that it is now handled by the P2P overlay) like *Registration Hijacking* [11] or against the server architecture of the system (that it is no longer present) like *Impersonating a Server* [11]. Following, we present a summary [7] of these attacks and how they affect SIP when a P2P overlay is used. Table 2.5 summarizes the rest of this section.

2.5.1 User Flooding Attack

In order to start a communication, one of the participants of the conversation has to send an INVITE message to the other (others). Due to the fact that end-user devices have been designed mainly to respond under normal conditions, they are normally able to process few incoming messages simultaneously. This fact can be exploited by a malicious user to flood a user with INVITE requests (Figure 2.17a), without waiting for any respond message, trying to paralyze the victim [187]. Also, as a side effect, an attacker could launch a DDoS (Distributed Denial of Service) attack by sending INVITE messages to a huge number of nodes of the network with the spoofed IP address of the victim as source (Figure 2.17b), that would receive a huge number of responses to collapse [198]. A great overview of this kind of DoS attacks against traditional SIP networks is presented in [220]. Unfortunately, most of the analyzed defenses are centered on the protection of SIP proxy servers.

[7] A full description of the attacks and defenses against SIP itself would require its own research and it is out of the scope of this thesis.

Strong authentication + Limitation

The best defense against this attack is to use a strong authentication method and to limit the number of conversations accepted from a single source [184] using a threshold-based rate limitation method such as the per-host incoming rate limit presented in [185]. Since the P2P overlay network is used to set up the SIP direct connection among users [85], SIP sessions can also benefit from the authentication systems based on certificates, Section 2.1.2, to establish secure sessions with TLS/DTLS or IPsec. However, a large enough coalition of attackers could paralyze the victim despite this measure.

Stateless Barrier

The paper [184] also suggests using an stateless barrier, similar to the TCP SYN cookies [186]. Stateless applications did not store any information about sessions until they are fully established (cookies are used to relate messages) and, therefore, reduce the resource consumption of each malicious connection attempt.

2.5.2 Parser Attack

In principle, SIP was designed with simplicity in mind as opposed to other existent VoIP protocols like H.323. However, with time and due to its flexibility, several extensions have been appeared to add new features to the protocol. These extensions have added new functionalities to the protocol but also have increased its complexity. Exploiting the amount of headers and options that can be included in a SIP message, an attacker could create unnecessarily complex and long messages to launch a *Parser Attack* to try to render inoperable the user application [187]. Some techniques that can be used to protect the system from this attack are:

Good Implementation + Message Size Limit

In order to defend SIP against this kind of attacks, the paper [187] recommends to develop a good implementation of the protocol following the standard defined in [11] and subsequent documents, plus limiting the size of the messages [184]. A method to asses the robustness of SIP implementations is described in the PROTOS research project [188].

Extra security Modules

The papers [189] and [190] study the security of the infrastructure of SIP-based networks and propose to enhance their security with additional security modules that provide specialized SIP related security features, such as a signature-based malformed message detection module that

checks incoming messages. Despite being designed for SIP servers, similar modules could be used in the user's P2PSIP applications.

2.5.3 Tampering with Message Bodies

Once the originator of a request has the contact address of the user she wants to start a communication with, a SIP session is initiated between them in order to perform the dialog. This connection is performed across various hosts of Internet that may see or modify the exchanged information [11]. Examples of possible consequences of this attack are Call Hijacking (a user dials a SIP URI but establishes a session with a different and malicious user) or Security Bid-Down (calls to or from a user are forced to use a lower level of security by an attacker) [221]. To protect the system from this tampering, we can use the following mechanism:

Transport Security

The best way to prevent this attack is protecting the transport mechanism by using TLS or DTLS [191] in conjunction with the authentication systems described in Section 2.1.2 to authenticate, encrypt and sign the exchanged information. A framework presenting these characteristics is described in [192] that enhances end-to-end security based on a hybrid symmetrical-asymmetrical cryptography and X.509 user certificates. Another possible options are to use one of the other security protocols presented in [11]: IPsec, SIPS-URI (Session Initiation Protocol Secure Universal Resource Identifier) [194] or Secure Multipurpose Internet Mail Extensions (S/MIME) [193] and [195].

2.5.4 SIP Session Modification

Once a dialog has been established between two system's users, subsequent messages can be sent that modify the state of it. Examples [187] are BYE requests (Figure 2.18), that terminate an established session; CANCEL requests, that cancel a previous sent request; Re-INVITE requests, that modify the parameters of an established session; or UPDATE requests, that modify not yet established sessions. Also, fake redirect responses (3xx) can be used to force the user to communicate with the attacker herself or some malicious entity [198]. It is critical that principals in a session can be certain that such requests are not forged by attackers. Otherwise, an attacker could alter the session or tear it down [11]. Using signaling authentication we can prevent this attack:

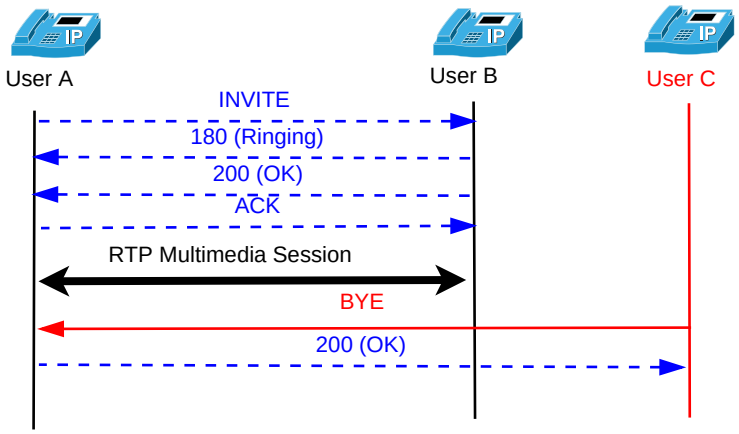


Figure 2.18: *SIP Session Modification.*

Signaling Authentication

Adding strong authentication to the signaling exchanged between the participants of the conversation [191] protects the session from unauthorized modifications. In [11] several security protocols are presented (TLS/DTLS, IPsec, SIPS-URI, S/MIME) that can be used in conjunction with the authentication systems described in Section 2.1.2 to authenticate the signaling and, therefore, prevent this attack. The already commented security framework presented in [192] should also work.

2.5.5 Media Session Alteration

SIP is the protocol used to initiate media communications between two or more entities. Once the session is initiated with SIP and described with SDP, the media transmission starts using the RTP/RTCP protocols. If no security mechanism is used to protect this media session, a malicious user could manipulate the RTP header packets to disturb a conversation. Furthermore, she could see or even modify the RTP data payload to listen or alter a conversation [191].

Media Security

The best way to secure the media transmission is to use SRTP instead of RTP to add confidentiality and integrity to the communication [187]. Another option is to use an IPsec tunnel. However, as it is designed specifically for streaming real-time data, Secure RTP is more efficient than IPsec in terms of bandwidth [196]. The paper [197] analyzes the most widely used media keying protocols (SDES -SDP Security DEScriptions for Media Streams- [222], ZRTP -Media Path Key Agreement for Unicast SRTP- [223] and DTLS-SRTP -DTLS Extension to Establish Keys for the SRTP- [224]) to derive the master key and other parameters in the cryptographic

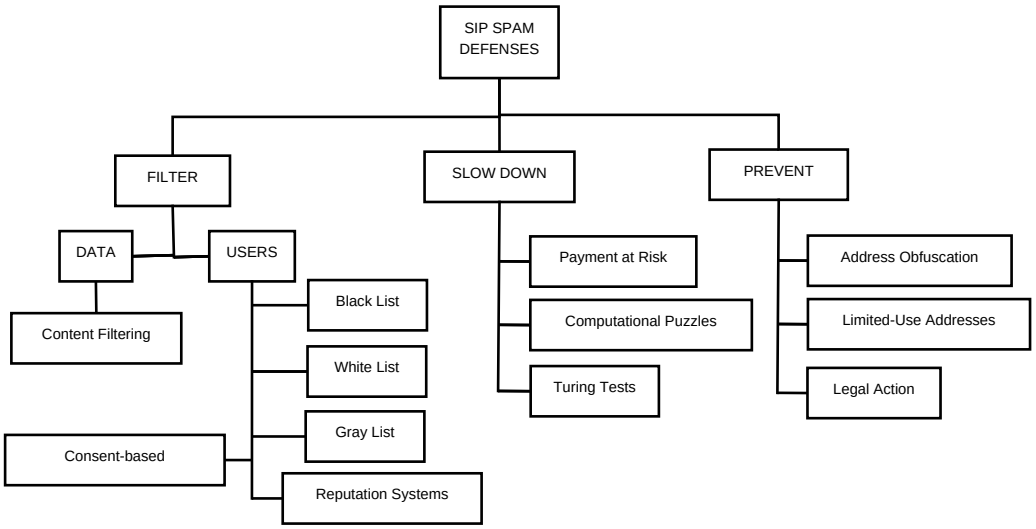


Figure 2.19: *Defenses against SIP spam.*

context needed for SRTP.

Split Traffic

An extra measure that helps to prevent this type of attacks is to maintain the VoIP traffic in a separate network from the non-VoIP traffic [191].

2.5.6 SIP Spam

Due to the different types of communications supported by SIP (video, voice and text) several types of spam exist that can flood this kind of systems, as defined in [44]:

- *SPam over Internet Telephony*: SPIT (also known as Call spam) is defined as an unsolicited set of session initiation attempts. This is the classic telemarketer spam applied to SIP.
- *SPam over Internet Messaging*: SPIM (also known as IM -Internet Messaging- spam) is defined as an unsolicited set of instant messages. This is very similar to email spam.
- *SPam over Presence Protocol*: SPPP is defined as an unsolicited set of presence requests (attempts to get on the user's buddy list).

Figure 2.19 groups the exiting defenses against this attack according to the their objectives: prevent the spam, slow it down and filter the communications either by their contents or by the users initiating them. Below, we summarize these defenses following the schema presented in [44].

Content Filtering

It is the most common form of spam protection used in e-mail. Messages are analyzed looking for clues that the e-mail is spam. As presented in [44], it is useless with Call spam, but works well with IM spam and spam over presence [198]. Also, like all methods working with patterns or statistical data, they suffer from the drawback of possibly generating false positives, resulting in legitimate communications being blocked [199].

Black Lists

Black lists are list of addresses (usernames or entire domains) that identify spammers. They are not very effective with email, because emails are easy to spoof and to obtain [44]. Nevertheless, if authentication mechanisms with a limited assignment of usernames like the ones described in Section 2.1.2 are used, black lists could be effective for SIP spam. In [198], several ways to create black list are described, such as spamtraps (email addresses that are not published anywhere).

White Lists

White lists are the opposite of black lists, a list of accepted senders [44]. IM buddy lists are examples of successful use of white lists. A similar approach might be effective in SIP in conjunction with a strong authentication mechanism like the ones described in Section 2.1.2. The main drawback of this approach is the "introduction problem" [44]: how to decide in the first time who should be placed in the white list. Examples of white lists are the buddy lists of communication systems like Skype [81].

Gray Lists

Gray lists [200] are complementary to black and white lists. When a communication that is neither in the black list nor in the white list is received for the first time, it is rejected temporarily. But its retransmission will be accepted. Gray listing is based on the assumption that spam software is simple and does not care about retransmissions. In this way, messages from legal users are never dropped unnecessarily and are always forwarded to the receivers, albeit slightly delayed [198].

Consent-based Communications

Consent-based measures are complementary to black and white list. When a user A wants to contact other user B that has not explicitly accepted (white list) or blocked (black list) A, B is informed that A wants to establish a communication with her. User B can then authorize or reject the communication [44]. These consent requests can be used as spam themselves,

but with a limited impact. The set of requirements for extensions and a framework to add consent-based communications to SIP are described in [201] and [202] respectively. Again, strong authentication mechanisms like the ones described in Section 2.1.2 would improve the efficiency of this measure.

Reputation Systems

Reputations systems [203] are used in conjunction with black or white lists and consent-based communications. This measure adds to the consent request the reputation of the user attempting to start a connection to help the other user to decide whether to accept or to reject her. Following this idea, [204] and [205] propose to introduce a trusted path-finder server in the P2PSIP infrastructure to help users decide if a received message is SPAM or not. Again, strong authentication mechanisms like the ones described in Section 2.1.2 are very important to limit the number of identities users can have to prevent them from cheating the reputation system [44]. For a deeper analysis of the existing reputation systems and a description of the attack and defense techniques for them, we refer the reader to [172].

Address Obfuscation

Another complementary measure is to obfuscate the SIP addresses in public sources of information, such as web pages or ENUM (telephonE NUMber mapping) [225] servers, to hide them from spam bots [44].

Limited-Use Addresses

With limited-use addresses users have a large number of SIP addresses of contact, each of which has constrains in its applicability [44]. A typical use is to give a different contact address to each correspondent and limit the communications arriving at a specif address to that correspondent. If spam arrives from one correspondent her limited address is canceled. The main drawback of this approach if the management and distribution of those addresses, whose difficulty increases in the case of P2PSIP.

Turing Tests

Turing tests or CAPTCHAs can be used to ensure that there is a human being behind every communication by generating and grading tests that humans can pass but current computer programs cannot [103]. In the case of SIP, voice-based Turing test should be used [44]. The SIP application interaction framework presented in [206] could be used for this purpose. Some of the drawbacks of using CAPTCHAs have been already commented in Section 2.1.2.

Computational Puzzles

Computational puzzles exploit the limitation of the user's available resources by making costly for an attacker to send spam [207]. When a user A wants to communicate with other user B, B responds with a computational puzzle [101]. B will not process A's request until B receive a valid response to the puzzle. As commented in Section 2.1.2, the main problem of cryptographic puzzles is the disparity of user's computational resources, increased by the ability of attackers of using zombies [208].

Payments at Risk

In order to communicate with a user B, a user A has to transfer a small amount of money to B. If B decides that the message is not spam and accepts the communication, user B refunds the money back to A. On the contrary, if B thinks that the communication is spam, she keeps the money and rejects the communication. The advantage of this approach is that sending spam becomes too expensive. Its drawbacks are the cost of the micro-payment commission and that it loses effectiveness when there are strong inequities in the value of currency between sender and recipient [44]. This approach was first proposed for email [209] and later migrated to SIP [210]. A discussion of the security requirements of this mobile and constrained payment scenarios is presented in [211].

Legal Action

Another possible measure is to pass laws that prohibit spam, like the "do not call" list in the United States [213] or the E-Privacy Directive of the European Parliament [212].

2.6 Miscellaneous

In the previous sections we describe the attacks that can be launched against the different mechanisms that form a P2PSIP system. Nevertheless, there are some attacks that do not affect only one specific part of it but the whole system. Following, we describe these attacks and the defenses presented in the literature to prevent them. Table 2.6 summarizes the rest of this section.

2.6.1 Eclipse Attacks

As stated in [74], the overlay's integrity depends on the ability of correct nodes to communicate with each other over a sequence of overlay's links. We have already described some techniques

Table 2.6: *Summary of Miscellaneous Attacks and Defenses.*

MISCELLANEOUS			
Attacks	**Security Requirement at Risk**	**Defenses**	**References**
Eclipse Attack	Confidentiality Integrity Availability	Sybil & ID Mapping A. Defenses	Section 2.1.1 and Section 2.1.2
		Fake Updates & Incorrect Routing Defenses	Section 2.3.2 and Section 2.3.1
		Man in the Middle A. Defenses	Section 2.3.3
		Degree Observation	[73] [74]
Free-Riding	Availability Latency	Monetary Payments	[50] [66] [226] [227] [228]
		Reciprocity	[172] [229] [230] [231] [232] [233]
		Neighborhood Monitoring	[234]
		Ejecting Misbehaving Nodes	[31] [60] [86] [172] [235] [236] and Section 2.3.1
Join and Leave Attack	Availability	Reactive Recovery	[237] [238]
		Periodic Recovery	[51] [53] [54] [239] [240] [241]
		Adaptative Recovery	[87] [174] [178] [242] [243] [244] [245]

that may disrupt this communication process by attacking specific parts of the P2PSIP system. What we present now is a more general attack that can be launched against the whole network using one or a combination of some of the attacks previously described, as shown in Figure 2.20. In an *Eclipse Attack*, an attacker (or a coalition of them) tries to mediate most overlay traffic in order to eclipse legit users from each others' view. In the extreme, it allows the attacker to control all the overlay traffic, enabling arbitrary denial of service or censorship attacks [74].

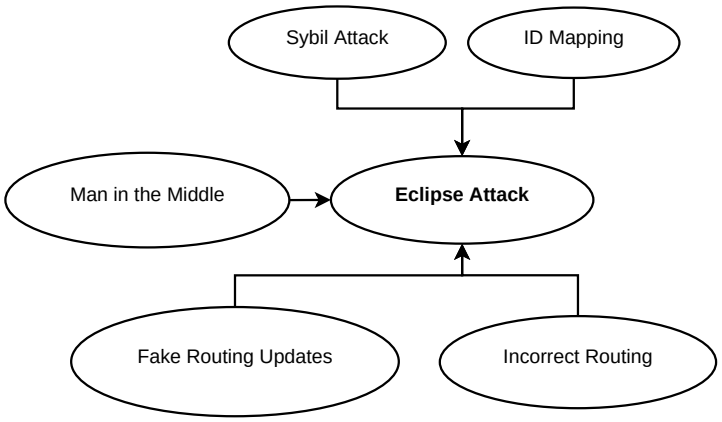


Figure 2.20: *Different ways to launch an Eclipse Attack.*

Sybil + ID Mapping

The simplest way to launch an *Eclipse Attack* is using a *Sybil Attack* in combination with an *ID Mapping Attack* to introduce nodes in specific parts of the overlay in order to control the whole traffic of the network. But also both methods can be used separately: an attacker may mediate the whole traffic of the network if she is able to insert enough nodes in the overlay, even if she cannot choose where; and a reasonable small number of attackers may control the network if they can place themselves in strategic parts of the overlay, even if each attacker has access to only one node. That is why the first measure against *Eclipse Attacks* should be to prevent *Sybil* and *ID Mapping Attacks*.

Fake Routing Updates + Incorrect Routing

An *Eclipse Attack* can also be launched by an attacker (or a coalition of them) in control of only a small number of nodes using the *Fake Routing Updates Attack* in conjunction with an Incorrect Routing Attack, first poisoning the routing tables of the good behaving nodes with inexistent entries or attackers' nodes, and afterwards discarding all the communication among them to eclipse legit nodes from each others. Hence, mechanisms to prevent these routing attacks should be also used to prevent *Eclipse Attacks*.

Man in the Middle

A more advanced *Eclipse Attack* can be launched using either methods described before (or a combination of both) in conjunction with a *Man in the Middle Attack*. In this way, an attacker not only prevents good nodes from accessing to the resources of the network but also make them believe they are accessing them when they are really accessing fake resources created by the attacker. Again, some of the methods described in Section 2.3 to prevent these routing attacks should be also used as a defense against *Eclipse Attacks*.

Degree Observation

Besides the mechanisms to prevent the attacks mentioned before and, hence, *Eclipse Attacks*, the papers [73] and [74] present a specific mechanism to stop *Eclipse Attacks* launched using a *Fake Routing Update Attack*. This defense is based in a simple observation: during an *Eclipse Attack* of this kind, the in-degree of attacker nodes must be much higher than the average in-degree of correct nodes in the overlay. Thus, correct nodes choose neighbors whose in-degree and out-degree (this is also bound to prevent malicious nodes to consume all the in-degrees of correct nodes) are below a certain threshold. Simulations with a small population of nodes (up to 10.000) show that that this mechanism reduces the number of malicious entries in the routing

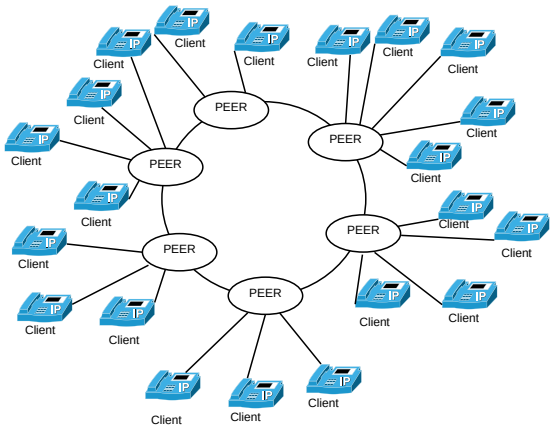


Figure 2.21: *Free-Riding of clients.*

table of good nodes considerably when a global *Eclipse Attack* is launched, but it cannot detect a *Fake Routing Updates Attack* against a specific node.

2.6.2 Free-Riding

For peer-to-peer systems to work properly, members of the system must collaborate, share resources. Unfortunately, some nodes of the network can try to consume as much resources of the network as they can while minimizing the amount of resources they provide to it. This kind of behavior, also known as *Rational Attack* [77], is what the literature presents as *Free-Riding*. Free-riders represent a serious problem to the performance of a P2PSIP system, existing the possibility of tearing down the whole network if the amount of free-riders is high enough. One example of a typical rational attack is a coalition of nodes acting as clients instead of peers in a P2PSIP system (using a protocol that allows its existence like RELOAD). Due to the fact that clients are nodes of the network that do not have neither routing nor storage responsibilities, if the number of clients became highly enough, they could overload the peers of the system and hence the whole network (Figure 2.21).

This phenomenon of *Free-Riding*, that is a consequence of the "Tragedy of the Commons" [246], was first analyzed for P2P systems on the paper [247], that found that approximately 70% of peers on the Gnutella network were free-riders. Subsequent studies, as [248] and [249], showed similar patterns of *Free-Riding* in P2P networks; confirming the importance of this problem. Below, we analyze the incentivation methods that can be used to prevent this attack, following the classification presented in [229]. Despite that most of the research on the prevention of this attack is related to file-sharing systems, it can show us an idea of the incentivation methods that can be used to overcome *Free-Riding* on P2PSIP systems.

Monetary Payments

With this incentive mechanism peers are paid to contribute to the system and pay to consume. With other words, one party offering some service to other is explicitly remunerated, either directly or indirectly [50]. An example is the distributed auditing mechanism presented in [66] to control the storage quota of the system used by each user. For that, each node publishes and digitally signs two logs containing the files it is storing and the files stored by other nodes on its behalf. This way, when a node B receives a storing request from a node A, B can test that A is paying for its quota of the network (comparing the amount of data it is storing with the amount of data other nodes are storing in its behalf) before accepting the request. Among others, examples of monetary payment systems are described in [226, 227] and [228].

Reciprocity

On this approach, users decide to take actions based on the past behavior of other users. In direct-reciprocity schemes a user A decides how to serve other user B based solely on the past behavior of B with A. On the other hand, in indirect-reciprocity schemes the decision of A also depends on the service provided by B to other users of the system.

Direct-reciprocity schemes are appropriate for applications with long session duration, as they provide ample opportunities for reciprocation between pairs of users [229]. Examples of direct-reciprocity are the tit-for-tat mechanism used in Bittorent to incentive the exchange of large digital files [230] and [250], the incentivation technique for tree-based multicast system presented in [231] or the taxation scheme for P2P streaming applications proposed in [232].

Indirect-reciprocity schemes, also known as reputation systems [203], are more scalable than direct-reciprocity schemes, especially for P2P systems with large population sizes, highly dynamic memberships, and infrequent repeated transactions [233], and therefore for P2PSIP systems. For a deeper analysis of the existing reputation systems and a description of the attack and defense techniques for them, we refer the reader to [172].

Neighborhood Monitoring

An alternative to reciprocity schemes is presented in [234], suggesting an approach against *Free-Riding* that requires every node to passively monitor its neighbors. Each node monitors and records the number of messages coming from and going towards its neighbors: based on five counters (queries routed by a node, queries routed towards a node, responses submitted by a node, responses routed by a node and responses received by a node) a node decides if its neighbors are free-riders. And, in case they are so, it takes actions against them, such as dropping their queries, to force them to collaborate. The main drawback of this approach is that its counters were chosen with file-sharing systems in mind. Also, the situation where colluding

peers mutually cover up for each other (not addressed by the authors) makes such monitoring protocols unusable [251].

Ejecting Misbehaving Nodes

Some of the routing techniques (hop testing, alternate routing path and parallel routing) described in Section 2.3.1 can help to identify malicious nodes either by noticing which nodes do not reply to the requests or by comparing responses coming from different paths. Also, periodic checks of the information stored in the network can be done: publishers can request their own data to verify that it is correctly stored and available [60], and comparatives between the information stored of one object in its different replicas can be done. Alike, reputation systems [172] can be used to identify malicious nodes. Once a malicious node has been identified, several actions can be performed. Locally, a node may label the misbehaving node as malicious to avoid sending any further message to it [31] and to block any message coming from it [235]. Globally, nodes can try to isolate the malicious nodes by informing the other nodes about them [235], by lowering their reputation (if reputation systems are used) or reporting their malicious activities to a central authority like a certification server. The paper [236] proposes to use a SOS (Secure Opinion Server) that stores the opinion, dynamically updated, for each user of the system, while the draft [86] presents a diagnostic mechanism intended to detect and localize failures or monitor performance in P2PSIP overlay networks that can be used to localize malicious nodes.

2.6.3 Join and Leave Attack

P2PSIP systems are continually changing due to node's join and leave (either intentional or due to a failure) operations. As in any self-organizing network, the participants have to perform maintenance tasks in order to adapt themselves to the changing nature of the network: the routing tables should be revised and the data should be relocated in accordance with the new topology. The cost of these maintenance operations (an excellent analysis of this cost in P2PSIP system is presented in [252], [253] and [254]), added to the expense for the network of the bootstrapping process of each new joining node, could be used by a malicious user, or a coalition of them, to quickly and repeatedly join and leave the network to try to destabilize the system. This, commonly known as *Join and Leave Attack*, could from increasing the delay to establish a multimedia communication session or update a user's status to completely collapse the system. The paper [255] presents three ways to launch this attack: controlling zombies to join and leave the system; controlling LANs (Local Area Networks) and disconnect several major parts to create concurrent failures; and controlling a set of nodes which are responsible for introducing newcomers according to the bootstrapping mechanism and spread forged messages to notify others of non-existing newcomers.

One of the first papers on analyzing the effects of churn in P2P networks is [256]. Like most of the papers in this area, it is related to the discussion of maintenance techniques under normal churn conditions and does not see it as a potential attack. Others, like [75], study the effects of join and leave operations with malicious purposes. However, it does not take into account the effects of the operations in the performance of the system and only pursues a balanced and honest network. Following, we present the maintenance techniques presented in the literature and its resilience against this attack.

Reactive Recovery

In reactive recovery nodes immediately react to changes in the network topology. When a node detects or is notified that a node member of its routing table or storing a replica of the data it is responsible for leaves the network, it automatically looks for a replacement. It reacts similarly to the inclusion of new nodes in the network. Examples of systems using reactive recovery were the first versions of OpenDHT [237] or OceanStore [238]. The paper [239] shows that reactive recovery works well for small overlays and moderate churn. However, it also shows that for bigger networks and high churn rates reactive recovery can lead to network collapse.

Periodic Recovery

In periodic recovery, nodes periodically actualize their routing tables independently of the changes in the network. Most of the actual DHT routing algorithms like Chord [51], CAN [53] or Pastry [54] use this technique. Studies such as [239] and [240] show that this strategy improves performance under churn conditions. Similar results, in this case focused on replication, are presented in [241], finding that periodic recovery outperforms reactive recovery under high churn.

Adaptative Recovery

Adaptative recovery takes into consideration the continuous evolution in network conditions. Each node collects statistical data about the network and dynamically adjusts its stabilization rate based on the analysis of this data [242]. Also, adaptative stabilization outperforms periodic stabilization in terms of both lookup failure and communication overhead. The importance of tunning the maintenance parameters due to the changing nature of the network is also highlighted in [243]. Focusing on replication, the paper [244] shows that biasing data placement towards highly available nodes reduce the number of objects that must be shipped for regeneration. However, this increases not equitably the load of the selected nodes. A similar method that manages availability in a dynamically changing network is used by Total Recall [178]. Results

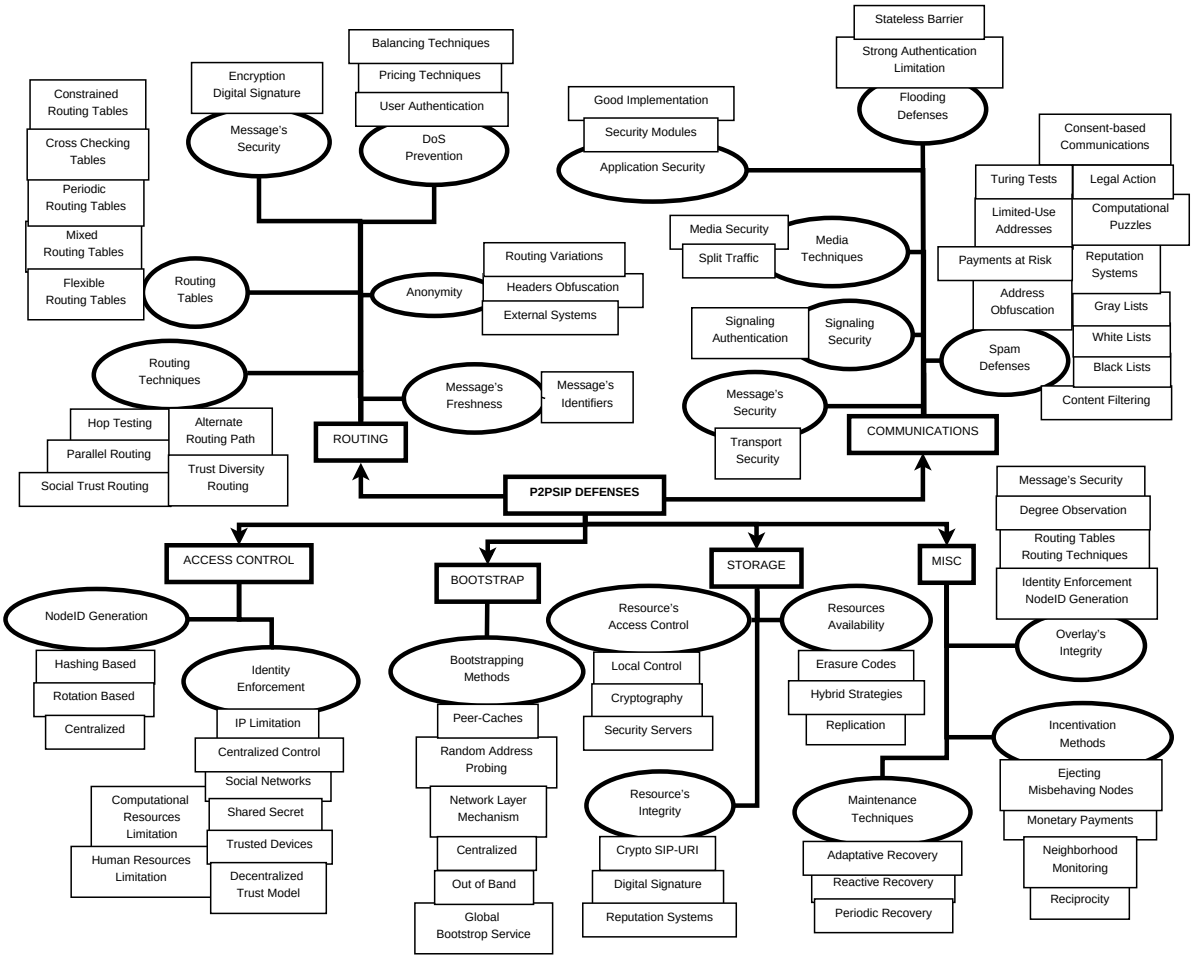


Figure 2.22: *Existing Defenses for P2PSIP Communications Systems.*

from the papers presented before are used in [87] to develop a self-tuning Chord algorithm for the RELOAD protocol.

Most of these analysis are based on churn rates of normal behaving file-sharing networks and do not take into account a really high churn rate due to a join and leave attack. It remains to be seen the effectiveness of these alternatives, mainly adaptative recovery, in such circumstances. A good start point for this analysis may be the performance vs. cost framework presented in [245] and the stochastic model presented in [174].

2.7 Security Discussion

Until now, we have analyzed the attacks that can be launched against P2PSIP systems and the existing defenses (summarized in Figure 2.22) that can be used to mitigate them. As we have seen, a secure access control and node-ID assignment are the key of the security of P2PSIP

systems since the effectiveness of most of the defenses against the presented attacks depends on the ability of the system to assign a PKC to each user and to limit the number of users that are malicious:

1. A secure node-ID is crucial at bootstrapping time to prevent malicious admitting nodes forging the node-IDs of trusted ones.

2. In relation to the routing scheme, the assignment of secure certificates to the users of the system provides an easy way to protect the system routing via TLS/DTLS or IPSec and to authenticate message's senders to discriminate malicious users. Also, a secure node-ID assignment prevent attackers from surrounding the possible users or resources they may want to monitor.

3. The storage service is also benefited by a secure access control since the access to the system's resources depends on a secure authentication and authorization of users. Besides, a secure node-ID assignment prevent malicious users to decide the resources of the system they control.

4. A secure access control also enhances the communication service allowing the authentication of SIP communications and improving the defenses against several attacks like DoS and spam.

5. Defenses against miscellaneous attacks, such as the *Eclipse attack*, are also improved with the use of a secure access control and node-ID assignment.

With this in mind, and since the development of specific countermeasures for all the services involved in a P2PSIP communication system would be a target too broad to be addressed by a single thesis, in the next three chapters of this thesis we focus on the design of new secure solutions to improve the access control service of P2PSIP communication systems.

2.8 Conclusions and Contributions

There are a plethora of attacks malicious users can launch against P2PSIP systems. Due to their impact on system operations, it is very unlikely that a P2PSIP system could run without implementing specific measures to prevent, detect and combat them.

Several defenses exist against each one of the presented attacks, however, the choice of the ones to be used and their implementation is a complex task. Most of the solutions presented so far to secure P2PSIP systems are adaptations of security mechanisms developed for P2P file-sharing systems or traditional SIP systems. And, despite its effectiveness on those environments, it is still early to affirm that they are the most appropriate for P2PSIP systems.

Also, each security measure has drawbacks: central servers limit the decentralized nature of the network, cryptographic protocols need extra computational resources, secure routing and maintenance mechanisms increase the load of the network, etc. These drawbacks limit the network capabilities, and, in some scenarios, it may not be possible to implement them.

Among the P2PSIP services analyzed during this chapter, access control stands out as the cornerstone of the security of P2PSIP systems and as the basis of the security of the rest of services. It is, therefore, of great importance to conduct a thorough analysis of the access control service to lay down a foundation for the secure development of P2PSIP communication systems.

From the results of the research presented in this chapter derives the paper *Survey of Attacks and Defenses on P2PSIP Communications* that has been accepted for publication in the journal *IEEE Communications Surveys and Tutorials*.

CHAPTER 3

NEW CERTIFICATION MODEL FOR P2PSIP AUTHENTICATION

> You wake up at Seatac, SFO, LAX. You wake up at O'Hare, Dallas-Fort Worth, BWI. Pacific, mountain, central. Lose an hour, gain an hour. This is your life, and it's ending one minute at a time. You wake up at Air Harbor International. If you wake up at a different time, in a different place, could you wake up as a different person?
>
> Chuck Palahniuk - Fight Club

As we have seen, and due to its decentralized architecture, the development of a secure access control service is one of the most challenging tasks of a PSPSIP system design. The access control of the IETF proposal RELOAD is based on the inclusion in the system of an offline CA that issues to each authorized user one or more PKCs [89] that allow them to access the network. These PKCs permanently link the username with the nodeID and the PK of a user in order to authenticate her and authorize her access to the system's resources. RELOAD's access control solves almost all the problems related to the access control in P2PSIP systems [94]: it provides a controlled assignment of nodeIDs, resistance to *Sybil Attacks* [70] and the authentication, integrity and confidentiality of the communications using the PK included in the certificates. However, we have found a flaw in its design that should be improved.

RELOAD's certificates permanently link a username with a nodeID. In this way, both a user and her device are identified by the same PKC and the same PrK is used to secure their communications. Nevertheless, devices and users are different entities that carry out different roles within the system and therefore the identity of a user (represented by her username) and

the devices she is using (represented by its nodeIDs) should be separated. Also, communications performed by a device acting as a node member of the network (like routing messages, retrieving or storing resources, etc.) should be separated from the communications performed by a user (making a call, updating her contact information in the network, etc.). It is unnecessary (and a security flaw) for a user contacting a node of the network just to route a message on her behalf to have the knowledge of the user using the contacted node. Likewise, there is no need for the contacted node to know which node the user performing the request is operating from. Following these observations, in this chapter we will present an alternative to the RELOAD's access control based on the separation of certificates for devices and users. Our approach improves RELOAD by raising the security of the communications with a two-layer security, providing an improved anonymity to users and allowing the establishment of a more secure network by using trusted devices with hard-coded certificates. Also, extra features are added, such as letting several users to be connected to the same device, allowing a user holding a single PKC to be connected to several devices, or having a greater interoperability with traditional SIP networks.

.

3.1 RELOAD's Authentication Security Discussion

At first glance, it seems that the RELOAD's model solves almost all the problems related to the access control in a P2PSIP network. However, the inclusion of the identity of the user and the device in the same certificate has several drawbacks:

- Users and devices are different entities performing different roles within a P2PSIP system. Devices are nodes of the P2P overlay network (represented by a nodeID) that offer services (to route messages, to store data, ...) to the system, while users (represented by an username) utilize these services, usually to establish media communications using SIP. Alike, the communications performed by a device acting as a node member of the network (like routing messages, retrieving or storing of resources, etc.) are independent from the communications performed by a user (making a call, updating his contact information in the network, etc.). Also, it is unnecessary (and a security flaw in terms of anonymity) for a user contacting a node of the network just to route a message on her behalf to have the knowledge of the user using the contacted node. Likewise, there is no need for the contacted node to know which node the user performing the request is operating from. Nevertheless, in RELOAD, users and devices are included in the same certificate and their communications are secured using the same PrK/PK pair like if they were a single and inseparable entity when they are not. A new certification scheme that clearly represents the different entities involved in the system, their roles and privileges is needed.

- One of the most interesting features of the modern communication systems (like SIP [11] or Skype [82]) is that a user A can be logged on several devices at the same time using the same account. When another user B wants to contact A, the call is automatically sent to all the devices user A is logged on. Once A answers from one of them, this conversation starts and the other calls are canceled. This is very useful in mobility scenarios, like a user that might be at home, at the office or on the road and has several fixed phones and a mobile phone; freeing the user from having to log off and log on from one device to another every time she moves. Also, it is secure since a PIN-code or a password protection over the certificate prevent making other administrative operations over her account apart from the authorized ones (receiving calls in this case). Nevertheless, in RELOAD, if a user wants to be connected to the network using several devices at the same time she needs to have several PKC with her username and different nodeIDs, one for each device. This is inefficient: user identity is the same regardless of the device she is using, and therefore should be represented by a single PKC. Also, having two or more different public-key certificates with the same public-key is against the nature of the PKCs. The other option, a user represented by several PKCs with different PKs is even more problematic.

- As well as mobile phones are private and intended for personal use, fixed phones are usually shared among several users (in a house, an office, etc.) that make calls and want to be reachable through them. In order to do so in a SIP based communication protocol, several users should be logged on the same device. Nevertheless, in RELOAD this is not possible. Due the fact that nodeIDs are directly linked to usernames, only one user can be logged on a device at a time. To overcome this limitation, a new scheme based on the independence of nodeIDs from usernames is needed to allow that multiple users could be logged on the same device at the same time, and, therefore, be reachable through it.

- Most of the nodes connected to a P2PSIP network are devices linked to an online user. However, there may be also nodes in the network playing a special role that do not need any user associated. Examples of such a nodes are PSTN gateways, security application servers [170] or public pay phones[1]. Also, despite being designed to support a P2PSIP network, RELOAD may also be used by other applications that may not require usernames, like, for example; file sharing. More flexibility, therefore, is needed in the certification scheme of RELOAD.

- It is common the use of hard-coded trusted devices in P2P networks when extra security

[1] As the mobile phone market was growing, pay-phones were disappearing from our streets. However, currently new VoIP pay-phones with extra functionalities (browse the Internet, view and send emails, etc.) are appearing on the streets [257]. It is not unreasonable, therefore, to think of the emergence of similar P2PSIP phones in the near future.

is needed in a system [258]. Hard-coded trusted devices are provided with extra security hardware measures at production time, like the inclusion of an anti-tampering certificate that uniquely identify them. In a P2PSIP specific context, apart from being useful in the development of a system with high security requirements, these devices could be used to add extra security to crucial nodes of the system such as a gateway or the auto-configured home routers a TELCO (TELecommunications Operator) sends to their users when they contract its services. Also, they can be used to limit the systems a device can connect to, like TELCOs do with the mobile phones they provide to their users. Nevertheless, RELOAD it is not suitable to be used with hard-coded trusted devices. The fact that the username also has to be included in the hard-coded certificate makes the devices user specific and not reusable. A new certification scheme where devices were independent from users is needed to permit an efficient development of environments where hard-coded trusted devices are desirable.

- P2PSIP communication systems emerge as an alternative to SIP in environments where the original client-server SIP's architecture may fail due to technical, financial, security, or social reasons. The idea is, therefore, that both kind of networks coexist and that a service provider (government, telecommunication company, university, etc.) develops one or the other depending on the specific scenario. In same cases, a network would consist of different interconnected SIP and P2PSIP subnets and a user would be connected to one type or the other depending on the place she is at a specific time. With this in mind, it is reasonable to think of the convenience of giving to the users a single identifier they could indistinctly use to log on both kind of networks. Unfortunately, the inclusion of P2P specific information, like user's device nodeID, in the current RELOAD's certification model presents some problems. On the one hand, SIP certificates would not be valid for P2PSIP since they do not include a nodeID, and, on the other hand, P2PSIP certificates disclose private but not necessary information in SIP networks, like the user's device nodeID. A new proposal, that permits using the same user certificate in both kind of network is needed.

Following these observations, in the next section of this article we propose an improvement over the RELOAD access control system based on the assignment of different certificates to devices and users.

3.2 Identity Segregation Scheme for P2PSIP

Devices and users are different entities that carry out different roles within a P2PSIP network. Each device represents a node participating in the P2P overlay network. Nodes offer services (to route messages, to store data, ...) to the network while users utilize these services, usually

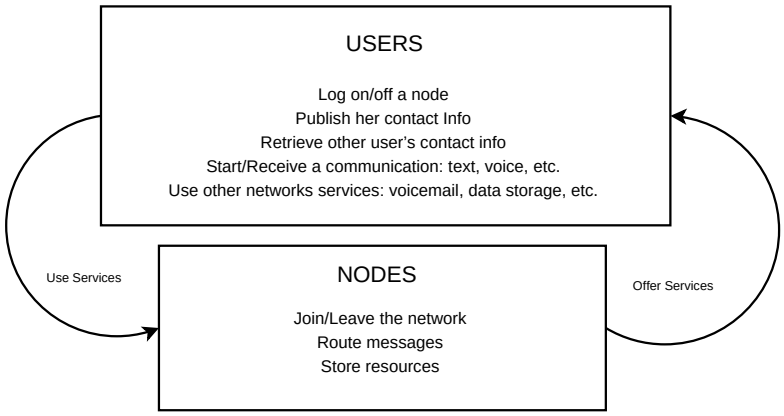


Figure 3.1: *Roles of users and nodes.*

to establish media communications using SIP. We can see it as a two layer stack: in the bottom layer nodes form a P2P overlay, offering services to the application in the upper layer utilized by users, as shown in Figure 3.1 .

Based on this differentiation, we propose the use of different certificates for devices and users. Each device in the network is represented by a certificate. This certificate, signed by the CA of the system, includes a nodeID that uniquely represents the device in the network and the device's public key (apart from other attributes that may be included for informational purposes). The device's certificate serves multiple purposes:

- Identifies the device by its nodeID and determines the location of the network where it is placed.
- The NodeID also authenticates the device against users that want to access the resources it stores.
- Specifies the location of the network where the device's resources, if needed, are placed, $\{Resource\text{-}ID = Hash(nodeID)\}$.
- Allows the device to join the system and to become a node of the network.
- Permits the device to establish TLS/DTLS tunnels with other devices of the network, adding hop-by-hop authentication, integrity and confidentiality to the network's routing.

For its part, each user also holds a certificate that includes the user's username that uniquely identifies the user within the network and the user's public key. The user's certificate serves multiple purposes:

- Identifies the user in the system by her username.

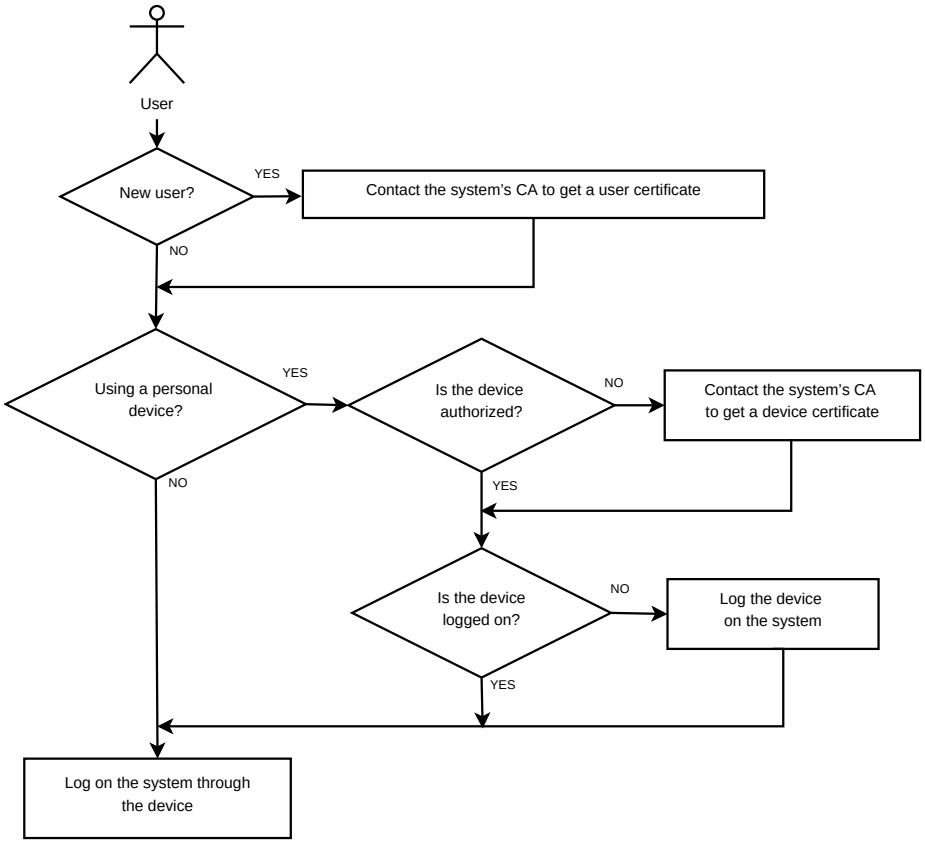


Figure 3.2: *Flow of the scheme.*

- Allows the user to join the system and to use its resources.
- Specifies the location of the system where the user can place her resources, {*Resource-ID = Hash(username)*}.
- Adds end-to-end authentication and integrity to the user's operations in the system[2].

Figure 3.2 presents the flow of the proposed scheme. Before accessing the system for the first time, a user has to contact the offline CA in order to get a PKC that authorizes her access to the system by linking her public key with her identity (username). The credentials required by the CA to issue this PKC (username/password, credit card payment, etc.) may vary depending on the system access policy.

Once the user is in possession of an authorized identity certificate, she needs one (or more) devices to access the services of the system. Two possibilities arise here: to access the system using a personal device or through a shared node of the system, such as a fixed phone in an office or a public P2PSIP phone. In the first case, if the user does not have any authorized device,

[2]End-to-end confidentiality is also possible using a previous query to obtain the Public Key of the destination.

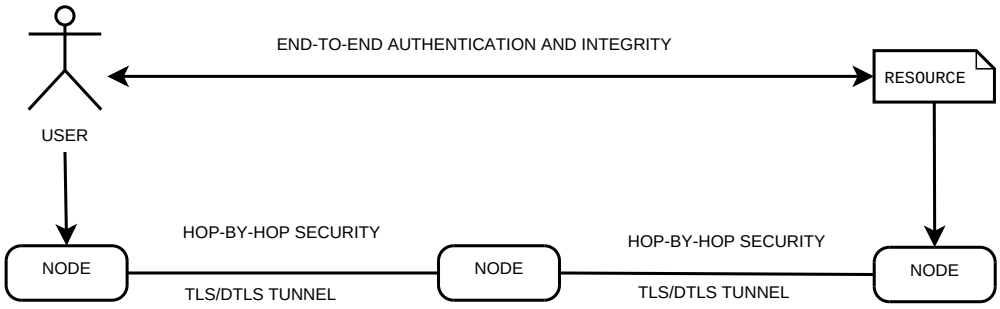


Figure 3.3: *Two layer communication security.*

she has to request the offline CA a nodeID certificate for each device she has (if the user is a new user she could merge this request with the previous one related to her identity certificate). These nodeID certificates link the device public-key with a random generated nodeID. Again, the credentials required by the CA to issue these certificates depend on the system access policy. Once a device have a nodeID certificate, it can join the network by presenting its certificate and can become a node (with the nodeID assigned in the certificate) of the overlay. In the second case, the shared device should be already on the system since the user do not have privileges over it (if she had, then it would be a personal device).

Once the device is online, the user (or several authorized users) can log on the system and can access its resources through the device. Each device uses its nodeID certificate to authenticate itself with the other nodes of the network and to establish TLS/DTLS tunnels with its neighbors. This way, all the communications performed by the device acting as a node member of the network (like routing messages, storing resources, etc.) are done using the nodeID certificate. On the other hand, the actions performed by users (making a call, updating her contact information in the network, etc.) are done using the user's certificate. The two different layers of communication are presented in Figure 3.3.

With our proposal we clearly split the different roles that devices and users represent within the network. Devices are independent from users. They form a secure P2PSIP network using their certificates to authenticate ones to each others and become nodes of the system. They by themselves maintain the network: stabilize the system when new nodes join or leave, establish secure channels of communications between them, route messages, control the access to the resources, etc. On the other hand, users are independent from devices. They use the services offered by the P2PSIP network but their identity and their privileges do not depend on the device they are using at a specific time. A representation of our proposal is presented in Figure 3.4. In it, we can see the relationship between users and devices and how each user and each device has its own certificate, with its own pair of private/public keys, that define each one as an entity represented either by its nodeID (devices) or her username (users).

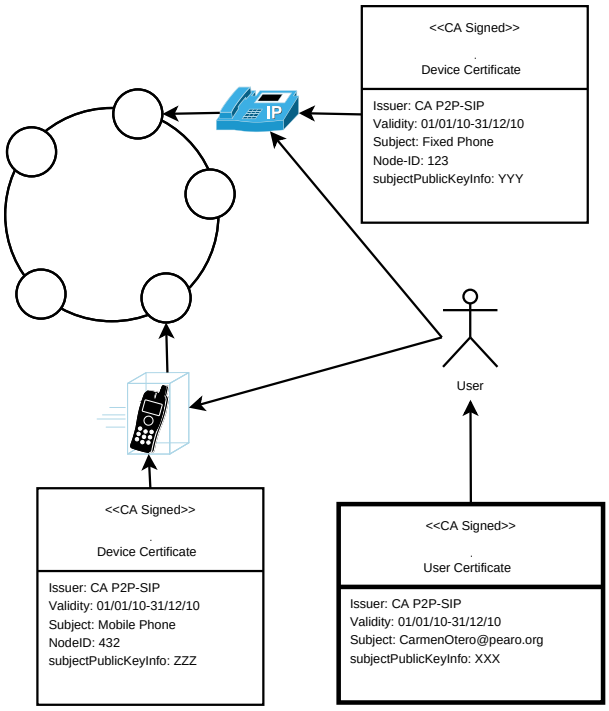


Figure 3.4: *Different certificates for devices and users.*

This proposal follows the model used by other communications systems, such as GSM (Global System for Mobile communications) where devices and users are separately represented by the IMEI (International Mobile Equipment Identity) stored in the phones and the IMSI (International Mobile Subscriber Identity) stored in the user SIM (Subscriber Identity Module), respectively. This split certification does not require any extra infrastructure with respect to RELOAD and does not exempt a user or a company from the responsibilities on their devices. In the next section we evaluate our proposal and its advantages over RELOAD.

3.3 Evaluation of Certification Models

Once our certification scheme has been described, we will present an evaluation that compares it with the RELOAD protocol. First, we pay attention to its flexibility through the analysis of various scenarios and how they can be easily solved with our proposal in contrast to RELOAD. Then, we present an analysis of the performance of both models by studying their operational cost in terms of bandwidth, computational resources and storage use. This analysis shows that the operational cost of both proposals is similar or even less in our model. In a third point, we see that the infrastructure needed for the development of both proposals is the same. Finally, an study about the security of both schemes shows that our proposal not only maintains the security of RELOAD but offers extra security functionalities.

3.3.1 Flexibility

In order to show the improvements in flexibility of our proposal we present four common scenarios and how they are solved with RELOAD and with our proposal:

- *Scenario 1*: A company wants to implement a secure P2PSIP system using trusted devices. These devices have to be user-independent and reusable.
- *Scenario 2*: A user wants to be connected to different devices (such as a fixed and a mobile phone) while at the same time holding a single PKC.
- *Scenario 3*: A laboratory has a single fixed P2PSIP telephone. All the people in the laboratory want to have full functionalities from that phone.
- *Scenario 4*: A company offers communication services to their clients through several interconnected SIP networks. In order to access the system, each user holds a PKC issued by the company. Now, the company wants to increase their network coverage adding new P2PSIP networks to the system. Also, they want their PKC access control to be valid and usable in the whole system.

RELOAD:

- *Scenario 1*: With RELOAD a device might be trusted and hard-coded with a certificate. Nevertheless, due to the fact that the identity of the user and the identity of the device are included in the same certificate, the device can only be used for that specific user that should be defined at production time.
- *Scenario 2*: In RELOAD if a user wants to be connected to different devices at the same time using the same identity she needs one PKC having the same username and a different nodeID for each device.
- *Scenario 3*: Several users can be reachable at the same phone if all of them put the IP address of that phone as their contact address. However, only one user can be logged on the phone at a time and therefore be able to use all the services offered by the P2PSIP system, such us making calls or hearing the voicemail.
- *Scenario 4*: Old client's SIP certificates would not be valid for P2PSIP since they do not include a nodeID, therefore, the company would have to issue a new certificate to each client to grant them access from every part of the system. Besides, RELOAD certificates would disclose private but not necessary information in SIP networks, like the user's device nodeID.

Our proposal:

- *Scenario 1*: The proposed scheme permits producing trusted devices easily. Each device can have a hard-coded certificate including a nodeID to join the network. These devices are trusted, regardless of users and reusable.
- *Scenario 2*: Due to the Independence of devices (having its own certificate) from users, a user can be logged on several devices using the same PKC without any problem.
- *Scenario 3*: The independence of the nodeIDs from the usernames permit different users to be logged on a single phone and use it with full functionalities.
- *Scenario 4*: With our proposal, old client's SIP certificates would still be valid and identical of those issued for the new users. The company only would have to issue a new certificate to each device intended to access from a P2PSIP part of the system.

As we have seen, the use of different certificates for devices and users gives more flexibility to P2PSIP systems:

- Users and devices are identified by different PKCs that permit them to represent different roles within the network.
- Devices can be connected to the network without the necessity of having an online user associated, for example nodes performing special services like a PSTN gateway.
- A user can be connected to several devices at the same time holding a unique PKC.
- Multiple users could be logged on the same device and have full functionalities over it.
- Greater interoperability with traditional SIP networks.
- Allows using reusable user-independent trusted devices.

3.3.2 Performance

In this section, we present a theoretical analysis of the cost of the main operations of the system in both proposals. Our analysis shows that to split the identity of users and devices in different certificates does not carry any extra performance cost in the system. The identity and the communications of these different entities are secured using different certificates but the cost of the operations is the same. Furthermore, it also shows that the fact that nodes can be connected to the network sharing its resources with no user associated improves the performance of the system under common circumstances.

	Initiator	Receptor
One side Auth	$2 \times RSA_{verify}$	RSA_{sign}
Mutual Auth	$2 \times RSA_{verify} + RSA_{sign}$	$2 \times RSA_{verify} + RSA_{sign}$

Table 3.1: *TLS tunnel operational cost.*

Before starting the analysis, we present some notation and the system configuration used during the test:

- M represents the total number of users of the system, while N the number of online nodes.
- The P2P overlay network used is Chord, the default for RELOAD.
- Each node has $\log N$ fingers in its routing table [57]. Nodes establish TLS tunnels with their fingers.
- The number of messages required for a lookup operation is $\bigcirc(\log N)$ while a join/leave operation takes $\bigcirc(\log^2 N)$ [51].
- Messages size and storage size are measured in terms of x.509 certificates, with a RSA key of 4096 bits, of 2 Kb of size.
- Users credentials are permanently stored in the network. This way users can receive offline messages.
- The operational cost of a TLS tunnel is measured in terms of cryptographic RSA operations, following [259]. Table 3.1 shows the cost of the establishment of a TLS tunnel taking into account that the cost of a RSA_{verify} is equal to the cost of a $RSA_{encrypt}$ and the cost of a RSA_{sign} is equal to the cost of a $RSA_{decrypt}$.

Obtain/renew credentials This action is carried out the first time a user wants to access the system and periodically to renew her credentials. The objective is to obtain one or more PKCs that grant her access to the system.

- *Messages:* It takes a communication with the offline certification server the first time and one extra for each renewal in both proposals.
- *Operational Cost:* Is the same in both proposals, the establishment of a TLS connection with the server. In this case only the server is authenticated, so $2 \times RSA_{verify}$ on the user side and RSA_{sign} on the server side.

- *Exchanged data:* In RELOAD, the server sends its certificate to the user (in order to authenticate itself) and one certificate for each device the user has, so the amount of exchanged data is: *(user's devices + 1)* $\times$ *2KB* . In our proposal, a user needs one certificate for each device she has and one extra for her credentials, so the amount of data exchanged is: *(user's devices + 2)* $\times$ *2Kb* .

- *Storage Resources:* In RELOAD, each user stores her device certificates: *number of devices* $\times$ *2KB* . In our proposal, each user stores her device certificates plus her user certificate: *(number of devices + 1)* $\times$ *2 KB* .

Join This action is carried out when a node joins the system. The objective is to insert users and nodes in the system.

- *Messages:* In both proposals a join operation costs $\bigcirc(\log^2 N)$ messages to the P2P network. Also, in order to register herself within the system the user has to perform one store operation [3].

- *Operational Cost:* To establish a TLS tunnel with its fingers the joining node has to perform $\log N \times (2 \times RSA_{verify} + RSA_{sign})$ operations while each finger has perform $2 \times RSA_{verify} + RSA_{sign}$ operations. The cost is the same in both proposals.

- *Exchanged data:* For the mutual authentication of the tunnels, nodes have to exchange their certificates, so: $\log N \times 2 \times 2KB$. Finally, the accepting node has to transfer the data the new node is responsible for to it, in average $M/N \times 2KB$. Same cost in both proposals.

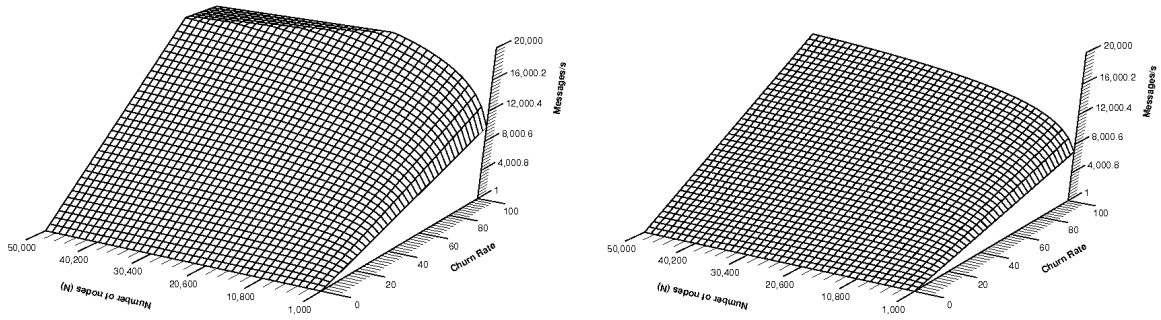
- *Storage Resources:* In RELOAD, the system has to store the contact information of all the users of the system: M $\times$ *2 KB* . In average each node stores $M/N \times 2KB$. In our proposal the cost is the same; since devices are nodes and not users of the system, device's certificates do not have to be stored in the network and only user's certificates are stored.

Leave This action is carried out when a node leaves the system. The objective is to remove users and nodes from the system.

- *Messages:* In both proposals, a leave operation costs $\bigcirc(\log^2 N)$ messages to a P2P network. Also, in order to set herself offline the user has to perform one store operation.

- *Operational Cost:* In RELOAD, each one of the $\log N$ fingers of the leaving node has to establish a new tunnel to substitute the one closed by the leaving node. Therefore, the

[3] Store operations are analyzed later in this section



(a) RELOAD. (b) Our Proposal.

Figure 3.5: *Messages exchanged in the network under churn.*

leaving node's fingers have to perform $\log N \times (2 \times RSA_{verify} + RSA_{sign})$ operations while each new finger has perform $2 \times RSA_{verify} + RSA_{sign}$ operations. The cost is the same in both proposals.

- *Exchanged data:* In both proposals, for the mutual authentication of the tunnels, nodes have to exchange their certificates, so: $\log N \times 2 \times 2KB$. Finally, the leaving node has to transfer the data it was responsible for to a new node, in average $M/N \times 2KB$.

- *Storage Resources:* None

Fetch/Store These actions are carried out by a user to access the resources of the network (usually to store her contact information or retrieve other user's ones).

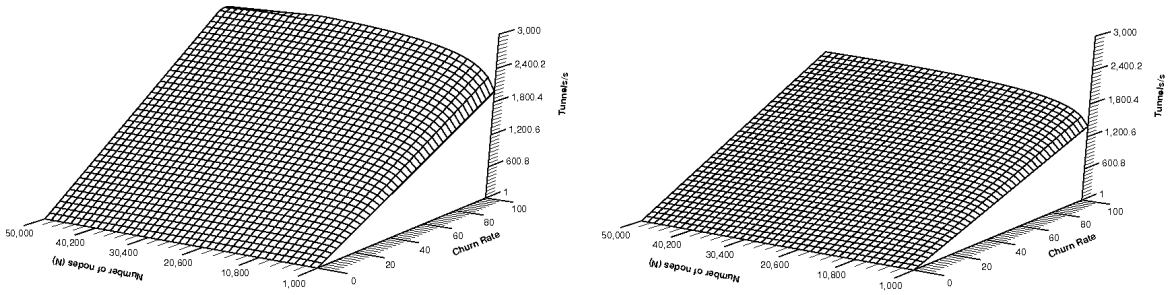
- *Messages:* In both proposals, these operations cost $\bigcirc(\log N)$ messages to a P2P network.

- *Operational Cost:* User's requests are authenticated by means of digital signature. Therefore, the requesting user has to perform one RSA_{sign} while the node responsible of the resource has to perform $2 \times RSA_{verify}$, one to verify the user's certificate and the other to verify the signature. Same cost in both proposals.

- *Exchanged data:* In both proposals, the only data exchanged is the user's certificate of $2KB$.

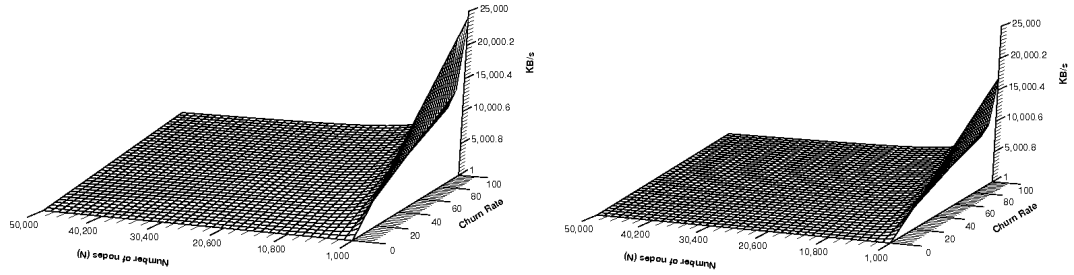
- *Storage Resources:* None

As we have seen, the cost of the system's main operations is the same in both proposals (the only difference is the exchange and storage of one extra certificate of 2 KB during the enrollment process with our proposal). However, there is another factor we should take into account. Two



(a) RELOAD. (b) Our Proposal.

Figure 3.6: *Operational cost of the network under churn.*



(a) RELOAD. (b) Our Proposal.

Figure 3.7: *Bandwidth requirements of the network under churn.*

of the major advantages of the P2P model over the classical client/server model, apart from its decentralization, are its self-organization and self-maintenance. Unfortunately, these advantages do not come at free cost. P2PSIP systems are continually changing due to churn (node's join and leave operations). As in any self-organizing network, the participants have to perform maintenance tasks in order to adapt themselves to the changing nature of the network: the routing tables should be revised and the data should be relocated in accordance with the new topology of the network. In RELOAD, due the fact that usernames and nodeIDs are included in the same certificate, every user join/leave operation causes a node join/leave operation with the cost these operations require in terms of network maintenance (as we have seen before). Nevertheless, in our proposal, since devices and users are different entities represented by different certificates, a user join/leave operation does not necessary implies a node join/leave operation. [4]

[4]It is possible, in both proposal, that a user enters in an "invisible mode" by removing her contact information from the network. In this case, the user seems to be offline for the other users of the network (neither they can have the knowledge that she is online or access to her contact information) but she is actually online since she can access to all the resources of the network, like her voicemail or the contact information of other users to initiate media calls.

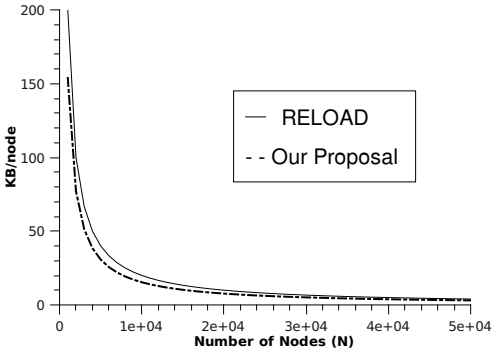


Figure 3.8: *Storage load per node.*

Based on this observation and on the previous description of the system's operations, we present below an analysis of the system performance with both proposals under churn. In the represented scenario, the number of online users in the system is constant (the number of concurrent joins and leaves is the same). Also, we presuppose that 30% of the users take advantage of the split certification model and do not log on or log off a device in the network before joining or leaving the system[5].

The analysis measures the maintenance cost of the system under churn in two different ways. First, we compare the behavior of the network as a whole in both schemes. The measured parameters are the number of messages exchanged (Figure 3.5), their operational cost (Figure 3.6) and their bandwidth requirements (Figure 3.7). The three graphics show that our proposal reduces RELOAD's maintenance cost under churn in proportion to the number of devices that remain online despite the user's joins/leaves. Finally, we analyze two specific parameters of the nodes: the storage load per node (Figure 3.8) and the lookup cost (Figure 3.9). The largest number of online devices in our proposal reduces the first one while increases the second. Nevertheless, the growth in lookups size is minimal ($\bigcirc(\log Z)$, being Z the number of devices that remain online despite the user's joins/leaves.)

3.3.3 Infrastructure

The RELOAD Certification mechanism does not require the implementation of a complete PKI. For its implementation, the establishment of one or more (depending on the number of users of the network) offline CAs is the only requirement. As with RELOAD, the only infrastructure needed to develop our proposal is an offline CA that issues certificates for users and devices.

[5]This may happen because they are connected to a shared device that usually does not go offline, such as a home P2PSIP router, etc. We use the 30% in our test because it is approximately the market share of fixed devices [260].

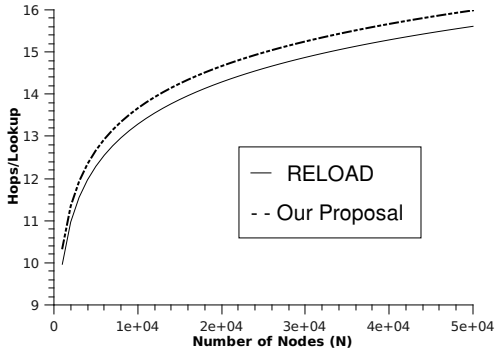


Figure 3.9: *Lookup cost.*

3.3.4 Security

In the first place our certification model maintains the security of the RELOAD model. A restricted assignment of PKCs signed by a CA grants its resistance against *Sybil Attacks* [70]. Also, the authentication, integrity and confidentiality of the communications are certified using the PK included in the certificates.

Besides, the separation of certificates for users and devices offers extra security functionalities:

- Users and devices are represented by different PKCs as they are different entities performing different roles within the network. Communications performed by the device acting as a node member of the network (like routing messages, retrieving or storing resources, etc.) are separated from the communications performed by a user (making a call, updating his contact information in the network, etc.). This provides a two-layer security.

- Since nodeIDs are not linked to users in the certificates, overlay maintenance and routing communications are performed between nodes without the unnecessary knowledge of which users are connected to them. Alike, user operations are not linked to nodeIDs, so users perform actions in the network without having to explicitly announce the node they are operating from. This way, a user can access the contact information of other users of the system without announcing her identity (a request can be secured hop-by-hop by the nodes of the system without including any information of the user operating the request). This lays a foundation to achieve better anonymity and reduces the chances of an attacker to track the media communications started by a user of the system. Nevertheless, this anonymity does not exempt a user or a company from the responsibilities on their nodes.

- Easy use of hard-coded trusted devices. Our proposal permits the establishment of secure P2PSIP networks where only trusted devices are allowed to became nodes of the network.

This reduces the possibility of an attacker to cheat the system's access control providing an extra protection against *Sybil Attacks*.

- When trusted devices are used to access the network, the separation between the communications of users and devices facilitates the identification of *Misbehaving Users*.
- The improvements in the performance of the system (presented in section 3.3.2) due the split certification reduces the impact of *Denial of Service Attacks*.

3.4 Conclusions and Contributions

In this chapter we have presented a new access control scheme for P2PSIP systems based on a clear differentiation between the identity of users and devices. This differentiation is built on the assignment of different certificates for both devices and users.

Our proposal splits the roles of each entity in the network in a more intuitive way, raises the security of the communications with a two-layer security, lays a foundation to achieve a secure P2P network where user's anonymity is improved and allows the establishment of a more secure network by using trusted devices with hard-coded certificates. Also, it adds extra features to the system, such as letting several users be connected to the same device, allowing a user holding a single PKC to be connected to several devices, or having a greater interoperability with traditional SIP networks. The evaluation conducted of both proposals shows that our scheme is more flexible and secure than the RELOAD certification scheme while improving its efficiency and preserving its simple infrastructure.

From the results of the research presented in this chapter derives the paper *Advantages of identity certificate segregation in P2PSIP systems* that has been published in the journal *IET Communications* (15 April 2011 – Volume 5, Issue 6, p.879–889).

CHAPTER 4

SECURE ACCESS CONTROL FOR ON-THE-FLY P2PSIP SYSTEMS

United we stand, divided we fall.

Aesop - The Four Oxen and the Tiger

Most of the actual P2P applications' access control is based on the inclusion of a logically centralized authority in the system, such as the offline Certification Authority of RELOAD [57] or the login server of Skype [82], because this is the unique method able to prevent that an attacker can control an unlimited number of nodes of the network (*Sybil Attack*) [70]. Unfortunately, the deployment of this infrastructure is not always possible, as in the case of on-the-fly P2P systems.

On-the-fly P2P systems are created with a limited duration to meet an immediate demand and a specific goal. Due to its on-the-fly creation, they cannot relay in any external infrastructure and all the functions must be performed by the entities forming the system. Examples of this kind of systems are: multimedia conferences among the participants of a congress, multi-player games started by the passengers traveling in a train or file-sharing applications established by the attendees of a meeting.

Within the scope of this thesis, P2PSIP communication systems, several alternative schemes have been already presented in Chapter 2 that try to solve the access control problem when the possibility of including a logically centralized authority (either online or offline) in the system is not possible: IP restriction [19], cryptographic puzzles [101], CAPTCHAs [103], web of trust [104], threshold cryptography [107], social networks [115], shared secret [57], identifier rotation [100] or identity cryptography [114]. Unfortunately, an analysis (Section 4.1) of the characteristics of all these alternatives shows that most of them are not suitable for on-the-fly P2PSIP systems. Also, the most common schemes for on-the-fly P2PSIP systems (IP based, shared secret and threshold cryptography) have some drawbacks: shared secret do not scale

well and it is insecure (secret disclosure), IP based does not work well with NAT and is also insecure (IP access restriction is not enough to be considered secure), and threshold cryptography have serious performance and scalability problems. From these observations, in this chapter we present a new access control scheme for on-the-fly P2PSIP systems, based on the recently published Internet Attribute Certificate Profile for Authorization [261], that tries to achieve an equilibrium between the simpleness and performance of IP based and shared secret schemes and the security of threshold cryptography based access control.

Our proposal parts from the assumption that an on-the-fly P2PSIP system is typically established by one or several trusted users to meet an immediate demand and a specific goal. In this scenario, the creator of the system initially certifies the new users of the network. Also, as the size of the system grows (and to reduce the overhead and a single point of failure on the creator) the creator issues ACs (Attribute Certificates) to other users giving them administrative rights. In possession of these ACs, they can do administrative tasks like accepting and certifying new users. This way, from a initial creating user; the access control of the system is distributed among several trusted users. Also, we present a simple protocol to maintain the freshness of the information shared by all the administrative users.

The evaluation conducted at the end of this chapter shows that our proposal greatly improves the security of IP based and shared secret schemes with no infrastructure cost and a minimal performance charge. Also, it achieves a similar level of security than threshold cryptography in non very hostile scenarios [1] while highly reducing its computational and communicational cost. All this factors position our proposal as an alternative to access control for on-the-fly P2PSIP systems in non very hostile environments where performance is a factor key.

4.1 Analysis of Existing Access Control Schemes for on-the-fly P2PSIP Systems

We have already presented in Chapter 2 a plethora of researches related to access control in P2PSIP systems, most of them focused on the prevention of the *Sybil* and *ID Assignment* attacks. However, in the particular case of on-the-fly P2PSIP systems, and due to its requirements, the establishment of a secure access control is an even bigger issue:

- Its on-the-fly creation imposes a total lack of external infrastructure (centralized servers, offline or online CA, etc.).
- It must be scalable since the number of possible users of the system is unknown.
- Computational and communicational cost should be minimum due to the diversity of the

[1] We understand as *non very hostile* the scenarios where administrators cannot be compromised, as discussed in Section 4.3.1.

devices of the system and the possible presence of devices with low bandwidth access rate, and low computational and battery resources.

- Secure assignment of IDs.
- Resistance against Sybil attacks.
- Robustness against malicious users.

In Table 4.1, we analyze the characteristics of the presented schemes (if the certification and the admission are centralized or not, its resistance to the *Sybil Attack*, the security of its ID generation, its scalability and its performance) based on the requirements for on-the-fly P2P systems' access control described before. As we can see, seven of the presented schemes (IP and Computational Based, CAPTCHAs, Trust Model, Social Nets, Shared Secret and ID Rotation) are fully decentralized; and therefore, prima facie, the more suitable for on-the-fly P2PSIP systems. Nevertheless, most of them do not present security guarantees and the others are impractical:

- IP-based access control is simple and efficient but do not present enough security guarantees, mainly when NAT networks and the IPv6 protocol are present in the system.
- Computational based access control only works well under the assumption that all entities operate under nearly identical resource constraints. Also, using the PK as source for IDs calculation it is not secure.
- CAPTCHAs are impractical for fully decentralized systems.
- A decentralized trust model access do not prevent an entity from having several identities and it is ID generation is not secure.
- The use of social networks is actually not an access control alternative but a measure to reduce the impact of *Sybil attacks*.
- Shared secret is a simple alternative but it is not really secure against neither *Sybil attacks* nor *ID mapping attacks*.
- ID-rotation schemes introduce an extreme cost in the network maintenance to reduce the impact of *ID mapping attacks*. Also, do not present practical measures against *Sybil attacks* and are very susceptible to *DoS attacks*.

In a second level of decentralization two alternatives appear (threshold cryptography and identity cryptosystems) that despite using decentralized schemes of admission, usually need a

Table 4.1: *Comparative of Access Control Schemes for P2PSIP Systems.*

ACCESS CONTROL	Cert.	Admission	Sybil Res.	ID Sec.	Scalability	Perf.
IP Based	**Dec**	**Dec**	Low	*Med*	**High**	**High**
Compt. Based	**Dec**	**Dec**	Low	Low	**High**	*Low*
CAPTCHA	**Dec**	**Dec**	Low	Low	*Med*	*Med*
Trust Model	**Dec**	**Dec**	Low	Low	*Med*	**High**
Threshold	Cen	**Dec**	*Med*	**High**	Low	Low
Social Networks	**Dec**	**Dec**	Low	Low	*Med*	*Med*
Shared Secret	**Dec**	**Dec**	Low	Low	**High**	**High**
ID Rotation	**Dec**	**Dec**	Low	*Med*	**High**	Low
Centralized	Cen	Cen	**High**	**High**	*Med*	**High**
Trusted Devices	Cen	Cen	**High**	**High**	Low	**High**
Identity Crypto	Cen	**Dec**	Low	*Med*	*Med*	**High**

TTP that previously certificate the users of the network. Also, the fully decentralized versions of threshold cryptography have some security problems [112], and high scalability and performance problems while identity cryptography does not really solve the major problems of IP based access control (not working well with NAT and slight Sybil defenses) but increases the complexity of the scheme. Finally, the fully centralized alternatives (including trusted devices) are secure but not suitable for on-the-fly P2PSIP systems.

With these inconveniences in mind, in the next section we present a new access control scheme for on-the-fly P2PSIP systems, based on the recently published Internet Attribute Certificate Profile for Authorization [261]), that tries to achieve an equilibrium between the simpleness and performance of IP based and shared secret schemes and the security of threshold cryptography based access control.

4.2 New Access Control Scheme for on-the-fly P2PSIP Systems

The schemes analyzed before present some deficiencies to manage the access control of on-the-fly P2PSIP systems. From these deficiencies, in this section we present a new access control scheme for on-the-fly P2PSIP systems which is based on the recently published Internet Attribute Certificate Profile for Authorization [261]. Our proposal parts from the assumption that an on-the-fly P2PSIP system is typically established by one or several trusted users with a specific objective and a limited duration. In this scenario, the creator of the system initially certifies the new users of the network. Also, as the size of the system grows (and to reduce the overhead and a single point of failure on the creator) the creator issues ACs to other users giving them

administrative rights. In possession of these ACs they can do administrative tasks like accepting and certifying new users. This way, from a initial creating user, the access control of the system is distributed among several trusted users. Also, we develop a simple protocol to maintain the freshness of the information shared by all the administrative users. Following, we present our proposal by describing the different phases that compound it.

4.2.1 System Creation

One user decides to create an on-the-fly P2PSIP system. In order to do so, she calculates an ID and a public key, and generates a self-signed (PKC) with them [2]. This certificate servers as root certificate of the system. The fingerprint of this certificate is included (in conjunction with the name of the network, etc.) in the header of every packet of the system to help the users identifying the root certificate and prevent impersonation attacks over it. Also, it is stored in a well-know location of the network (for example *resource-ID = 0*) to be easily accessible for all the users of the system. Due to the system limited duration and to maintain its simpleness, all the certificates used are short-lived and no revocation mechanism is implemented.

After creating her root certificate, the creator starts the on-the-fly P2PSIP application and waits for incoming connections. When the creator accepts a join request (see section 4.2.3 Bootstrapping) from a new user A, she calculates a random ID for A and uses the root certificate to sign the ID certificate[3] that grant A's access to the system.

4.2.2 System Scalability

It is not a good idea to have a single point of failure on the creating peer of the network. Also, the network may grow large enough to be too much for a single peer to manage it. Attribute Certificates are used to solve this inconvenient. After the creator user starts the system, she can issue ACs to other users giving them administrative rights. In possession of these ACs they can do administrative tasks like accepting and certifying new users. Also, the creator user gives to the other administrative users a symmetric password to access the administrative resources[4].

The structure of the ACs used to grant admin rights, Figure 4.1, is in concordance with the standard profile for authorization presented in [261] and has the following fields:

- *acinfo.version*: Represents the version of the AC used. It should be v2 (1).

[2] Any other kind of certificate (as for example a PKC signed by a TTP) can be used as root certificate. We use a self-signed one to illustrate the most typical and decentralized scenario.

[3] For simplicity reasons, we present our model using PKC certificates as ID certificates. However, ID certificates issued by the creator (and the administrative nodes) could also be AC linking the privilege of accessing the network with a previously obtained (out of the system) or a user's self-signed PKC.

[4] In case administrators want to share private information by leaving it encrypted in a well known location of the system.

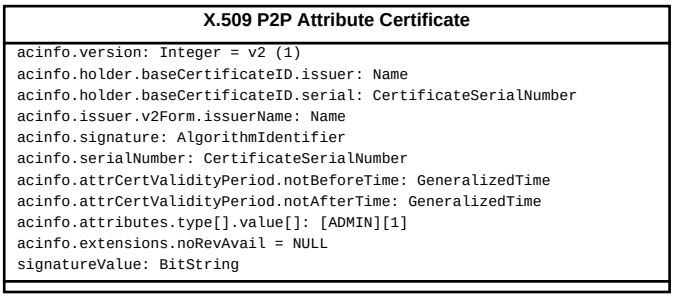


Figure 4.1: *Attribute Certificate structure for on-the-fly P2PSIP networks.*

- *acinfo.holder.baseCertificateID.issuer* and *acinfo.holder.baseCertificateID.serial*: Points to the PKC certificate to which this AC applies, i.e. the PKC certificate of the user with admin rights. The *issuer* field represents the issuer of the holder's PKC (the system creator) while the *serial* field represents its serial number. Both must be equal to the fields in the PKC of the holder (user with admin rights).

- *acinfo.issuer.v2Form.issuerName*: Name (in its PKC) of the issuer, i.e. the system's creator.

- *acinfo.signature*: Algorithm identifier used to validate the AC. It can be any of the defined in the standard [89].

- *acinfo.serialNumber*: Serial number of the AC. The pair issuer/serialNumber must be unique.

- *acinfo.attrCertValidityPeriod.*{*notBeforeTime,notAfterTime*}: Period of validity of the certificate.

- *acinfo.attributes.type[].value[]*: Set of privileges the AC gives to the holder. The defined type for admin rights is $ADMIN$ and the value 1.

- *acinfo.extensions.noRevAvail*: This field indicates that the revocation of this certificate is not possible. It includes no data.

- *signatureValue*: Signature of the issuer (the system's creator) over the certificate.

Due to the relative small size on-the-fly P2PSIP systems use to have, the number of users with administrative rights should be small. It is, nevertheless, crucial how this users are selected for the security of the network. In principle, we suppose a previous relationship among these users (as for example, the chairs of a congress that start a conference where all the attendees will participate). However, it is also possible the non existence of a previous relationship among

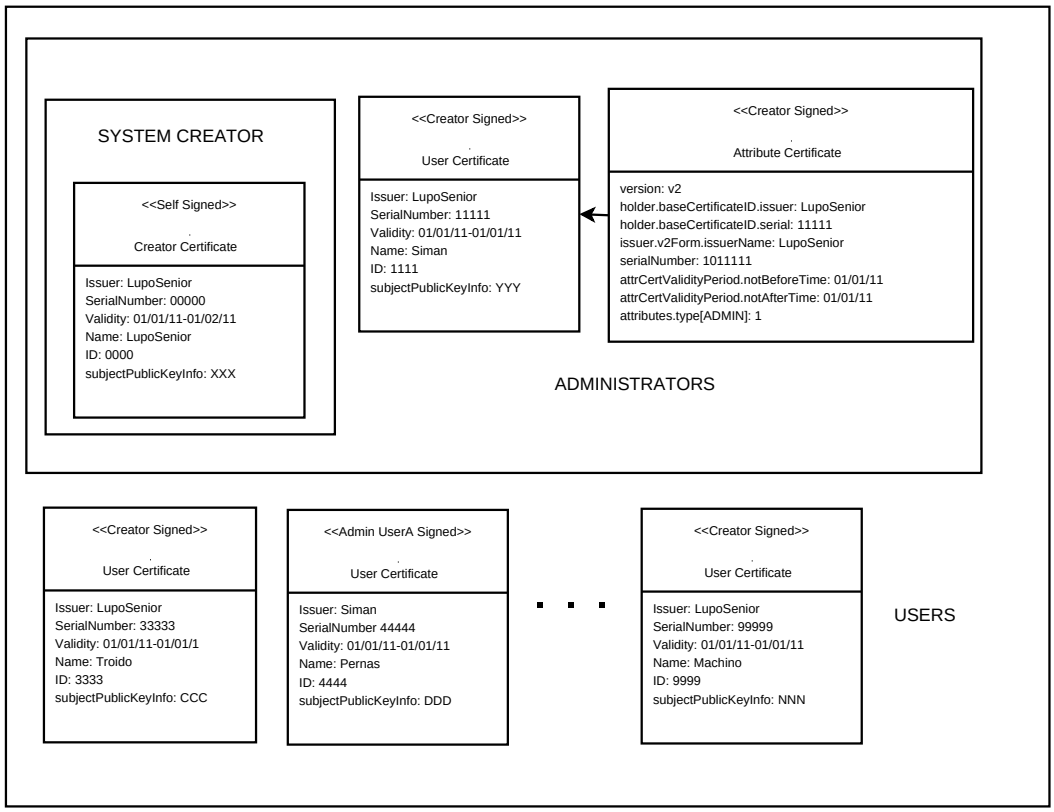


Figure 4.2: *On-the-fly P2PSIP access control scheme hierarchy.*

them and to use other methods, such as social networks or reputation systems, to select the administrative users.

All the administrative users' certificates are stored in conjunction with its ACs under the same *resource-ID* that the root certificate to be easily accessible to all the users of the system. Also, when the creating user certifies a new administrative user it informs the others (administrators) that a user has become an administrator. Figure 4.2 presents the hierarchy of the scheme and the contents of all the fields of the ACs used[5].

4.2.3 Bootstrapping

The first thing a user has to do in order to access the system is finding a peer already member of it. Several decentralized bootstrapping mechanism such as peer-caches and random address probing [122] or multicast groups [123] can be used for this purpose. Once the joining peer has contacted a peer member of the network (bootstrapping peer), two possibilities arise:

1. The joining peer has a certified ID already. In this case, the two peers authenticate one to

[5]In the case of the presented PKCs, and for simplicity reasons, only the most representative fields for our proposal are described; being valid any PKC compliant with the standard profile described in [262].

each other, the joining peer initializes its state and becomes part of the network. Also, if the new peer is an administrator it has to inform the other administrators of its presence in the network and update its DDACL[6].

2. The joining peer has not a certified ID. In this case, if the access control policy states that the bootstrapping peer has to perform a security check (like checking that the joining peer is in possession of a shared password), this must be done. If it is successful or no security check had to be done, the bootstrapping peer localizes one of the administrative peers of the network and suggest the joining peer to contact it in order to get its credentials (it is also possible to route the request throw the system but this alternative is more susceptible to suffer DoS attacks). Otherwise, if the security check fails, the bootstrapping peer closes the connection with the joining peer.

4.2.4 Access Control

The access control of the system is based on, what we call, a DDACL (Dynamic Distributed Access Control List) used for administrative purposes, containing several fields. This DDACL is based on the typical ACLs (Access Control Lists) but in a distributed way, using modification and update messages to maintain the freshness of the information stored by all the administrators. Following, and with illustrative purposes, we present an example of possible entry's structure with seven fields for row (apart from a global field of its last modification time) for a typical username/password access control policy:

- *User*: Authorized username.
- *Pass*: Password required to certificate the authorized user.
- *Timestamp1*: Local since the Unix era when the user was authorized.
- *Signature1*: Signature over the username, password and timestamp1 that grants its authenticity and who inserted it.
- *ID*: Assigned ID to the authorized user.
- *Timestamp2*: Local since the Unix era when the user was certified.
- *Signature2*: Signature over the ID and timestamp2 that grants its authenticity and who certified that user.

At the system creation, the root peer creates the DDACL saving a local copy of it and storing another one in a well-know location of the network encrypted with the administrative

[6]DDACLs are described in section 4.2.4.

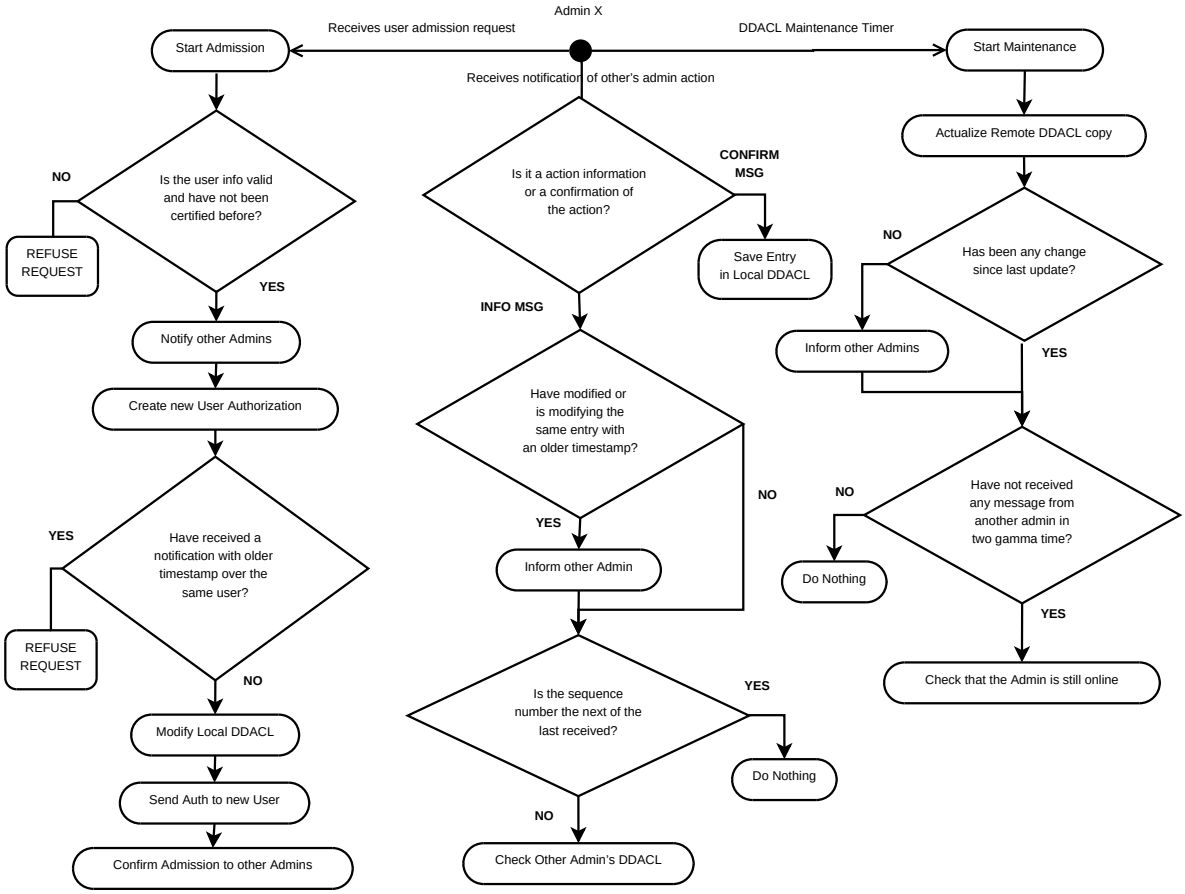


Figure 4.3: *DDACL Maintenance workflow.*

symmetric key. When the creator authorizes a new administrator user, she sends the DDACL to him. The new administrator user also stores a local copy of its DDACL and another one in a well-know location of the network (for backup purposes) encrypted with the administrative symmetric key. All the administrative users keep their own DDACL, but synchronize them. Each time an administrator users changes an entry of the DDACL it informs the other members of the administrative group of her intention of performing a change sending them the intended new or modified entry. If another administrator is modifying the same entry (for example, two admin peers trying to certify the same user, or a ID collision) the one with the older timestamp is assumed as the correct while the other(s) have to cancel the change. Finally, the admin that modifies the entry informs the others about the definitive modification. Each actualization is labeled with the owner's user and a sequence number, allowing other admins to realize if they have missed any actualization from a specific admin when they receive an update from it. If this is the case, the no updated admin checks the other admin DDACL (other admins' local copies are requested first, since they are more up to date, while remote copies are only requested as backup when the local copy requests fail) for the missed actualizations. Also, each certain period

of time γ, the administrators actualize the copy of their DDACL stored in the network and if they have not done any actualization since the last update inform the other admins of that. In the same way, if an admin (A) has not receive any update from one of the administrators (B) in $2 \times \gamma$ time, A contact B to see if she is still online. Figure 4.3 represents the workflow of this maintenance process. Finally, it is important to note that all these operations (but the creator's selection of other administrative users[7]) are performed automatically by the application and without any intervention from the users.

4.2.5 Certification Checking

As already commented, the root certificate of the system is saved in a well known location of the network and its fingerprint is included in the header of every packet of the system to prevent impersonation attacks over it. This way, it is very easy to authenticate a user certified by the creator of the system. All that another user has to do is checking that the signature of her certificate was done with the root certificate, stored locally and in the well known location of the system, which fingerprint should be equal to the one contained in every packet of the system. In case the certificate was created by one of the other administrators (admin A in this example), the process is as follows: first the user checks that the certificate of Admin A and A's AC (granting her as administrator) were signed by the root certificate which fingerprint should be equal to the one contained in every packet of the system. Finally she checks that the signature of the other user certificate was in fact done by admin A. This implies that, in principle, the cost of verifying a certificate signed by an admin other than creator is the triple. Nevertheless, admin-caches are used to so solve this inconvenience. The first time a user verifies a certificate signed by an unknown admin it costs the triple (verification of admin's certificate and AC, and verification of the other user certificate). However, the next time this user see a certificate signed by this admin only have to perform one verify operation since she has in her cache that the certificate and the AC of this admin are valid. Due the small number of administrators of the system, this mechanism very efficient and lightweight; as presented in section 4.3.3.

4.3 Evaluation of on-the-fly Access Control Schemes

Once our access control scheme has been described, we will conduct an evaluation that compares it with the more relevant alternative schemes presented in the literature for access control in on-the-fly P2PSIP systems: IP, shared secret and threshold cryptography based access control schemes. In the case of threshold cryptography, two different proposals are going to be evaluated:

[7]It could be also possible to automatize this process based on the number of users of the system and choosing the candidates from a web of trust model or a social network. However, that possibility is out of the initial scope of this paper.

a modification of the typical threshold cryptography scheme where the creator of the network acts as trusted dealer (to eliminate the requirement of a TTP in the system) and a fully distributed threshold access control scheme. The rest of this section compares all the schemes in terms of security, infrastructure requirements and performance; showing that our proposal improves the security of IP based and shared secret schemes with no infrastructure cost and a minimal performance charge. Also it achieves a similar security level than threshold cryptography in non very hostile scenarios while highly reducing its computational and communicational cost. Before starting the analysis, we present some common notation used in the rest of this section:

- M represents the total number of peers of the network, n represents the total number of administrative peers (including the creator), while t represents the threshold of the system.
- The DSA threshold scheme analyzed is the presented in [263]. The creator-centralized initialization is based on the original Shamir's secret sharing protocol [109], while the decentralized one is based on the distributed key generation presented in [264]. Their performance analysis is based on the results presented on [263, 111, 112].

4.3.1 Security

From the security point of view, three could be the main concerns of our proposal: the authenticity and confidentiality of the communications, the accuracy of our DDACL and the resilience of our proposal against malicious behavior. In relation to the first concern, despite it is not explicitly stated (for simplicity) during the explanation of our proposal; it is straight forward to see how its communications and data can be secured used standardized protocols, such us TLS/DTLS or IPSec, due the possession of each user of a PKC. Also, administrative data is protected by a secret symmetric key.

In relation to the second concern, our DDACL notification systems opens a slight possibility for a malicious user to be certified by more than one admin at the same time if during the certification process the malicious user can block the modification messages exchanged among the administrators. Possible solutions to this issue could be to use ACKs (ACKnowledges) for the notification messages (not a real improvement, because if the malicious user can block the notifications she could also block the ACKs) or do not definitively certificate the user until surely checked that no other administrator is also certifying her (by establishing a more complex and synchronous communication among admins). However, the low probability of this attack (due the difficulty of blocking the communications) together with the small number of certificates a malicious user could get (one for each admin as maximum) and the extra cost these alternatives would carry to the system, suggest us to keep using our alternative to maintain the simpleness and efficiency of our approach.

In relation to the third, we will compare the security of our proposal with the security of IP based access control, shared secret, and threshold cryptography. To do so, we present several scenarios and analyze the security of the discussed proposals against them.

Scenario 1, Non Hostile - Good Users

This is the trivial scenario where all the users are good: they present one single entity to the network and do not try to disturb or control the operations of other peers within the network. Under such a circumstances it is clear that all the alternatives are secure.

Scenario 2, Hostile - Several Misbehaving Nodes

In this scenario, a fraction of normal (non administrative) users of the network are malicious and try disturb the working of the system. However, they do not have capabilities of denying or controlling the operations of other users.

- *IP based:* Malicious users can include as many identities in the system as IP addresses they have available[8]. Since most of the on-the-fly P2PSIP systems are deployed over networks that usually assign IPs dynamically, like wireless or Ethernet networks, the number of IPs is high enough to let a reduced number of malicious user compromise a large part of the network. Furthermore, in case NAT networks are involved (inclusion of the port in the ID calculation) the defenses against Sybil and ID mapping attacks are inexistent.

- *Shared Secret:* Once a malicious user knows the shared secret, nothing stops her to distribute it among others malicious users to allow them to join the network. Also, each user can present as many identities to the network as she wishes and decide their location.

- *Threshold Cryptography & Our Proposal:* Grant a secure assignment of IDs and present total resistance against *Sybil Attacks* due to the virtually centralized authority formed by the administrators of the network.

Scenario 3, Hostile - Several Misbehaving Nodes with DoS Capabilities

In this scenario, a fraction of normal (non administrative) users of the network are malicious and try disturb the working of the system. They have capabilities of denying the operations of other users (admins included), but not of controlling them. IP based and shared secret access control schemes are do not described here again because they have been proved to be insecure in less hostile scenarios.

[8]Even with a single device available, like a laptop, a user may use virtualization to simulate several devices and include them in the network.

- *Threshold Cryptography:* Under this scenario the scheme is still secure. Also, the access control mechanism works without problems while at least t of the admin users are available. However, if the malicious users can deny the operations of at least $n-t+1$ admins peers, they completely deny the access to the system.

- *Our Proposal:* Our proposal is secure under this scenario. Furthermore, in order to deny the access to the system the malicious user has to deny the operations of all the n admin users.

Scenario 4, Very Hostile - Several Misbehaving Nodes with DoS and Control Capabilities

In this scenario a fraction of normal (non administrative) users of the network are malicious and try disturb the working of the system. They have capabilities of denying or controlling the operations of other users (admins included). IP based and shared secret access control schemes are do not described here again because they have been proved to be insecure in less hostile scenarios.

- *Threshold Cryptography:* The distribute version of this scheme is secure while less than t admin peers were corrupted by the malicious users. But the dealer based version can be also compromised by compromising the dealer (creator user).

- *Our Proposal:* Our scheme fails if the malicious users can take control of one of the admin peers.

As we can see, our proposal clearly surpasses the security of IP based and share secret schemes while achieving a similar level of security than threshold cryptography in non very hostile scenarios. In very hostile scenarios our scheme has better resilience against DoS attacks while threshold cryptography resists better to a possible compromise of admin peers.

4.3.2 Infrastructure

Our proposal is fully decentralized and do not need any external infrastructure (servers) to work. Also, the fact that the creator can adapt the number of administrators to the size of the network makes it very scalable.

4.3.3 Performance

In this section, we present a theoretical analysis of the cost of the main operations[9] of our scheme in comparison with the more representative proposals for on-the-fly P2PSIP systems.

[9]The costs required for protecting each protocol message are not taken into account because they vary with the specific secure protocol used.

We also present a particular analysis (Figures 4.4, 4.5, 4.6, 4.7 and 4.8) of a scenario with $n = 5$ administrators and threshold (in threshold-crypto schemes) $t = 2$. We have chosen this particular scenario because $n = 5$ administrators represent a reasonable number for a small/medium size on-the-fly P2P system and $t = 2$ can give us a good idea of the minimum extra performance cost induced by threshold security. Following we present the setup used in the performance analysis:

- RSA key size is $k = 1024$ bits.
- DSA parameters are $p = k$ and $q = 160$ bits.
- The operational cost is measured in terms of bit operations. Table 4.2 shows the complexity and the performance[10] of the main RSA and DSA operations having in mind that:
    1. The cost of $RSA_{encrypt}$ is equal to the cost of RSA_{verify} and the cost of $RSA_{decrypt}$ is equal to the cost of RSA_{sign}
    2. RSA_{verify} operation takes exactly 17 modular multiplications using a fixed short exponent $E = 2^{16} + 1$ [265].
    3. RSA_{sign} can be computed as two $\frac{k}{2}$ modular exponentiations plus a recombination [265].
    4. We assume that each user has already a PK and, therefore, we do not include its calculation cost in the performance analysis.
    5. DSA_{verify} takes one p-modular exponentiation to an exponent no more than q, plus no more than q multiplications modulo q [142].
    6. DSA_{sign} main operations are two p-modular exponentiations to exponents no more than q [142].
    7. We assume the system uses precomputed values (p, q, g) for the $DSA_{generation}$ of the Master Key (MK) since they can be public and common to a group of users [142]. Therefore, its generation only involves a p-modular exponentiation to an exponent no more than q.
    8. A modular exponentiation involves $\bigcirc(k)$ multiplications and a modular multiplication has a complexity of $\bigcirc(k^2)$ bit operations[266].

[10] Using the OpenSSL (version 0.9.8g) speed test in an Ubuntu 10.04 (lucid) 64-bits with kernel Linux 2.6.32-25 running over an Intel(R) Core(TM)2 Quad CPU Q8200 @ 2.33GHz with 4GB of RAM.

Table 4.2: *Operational equivalences.*

	Bit Operations	Performance for $k = 1024$ bits
RSA_{verify}	$17 \times \bigcirc(k^2) = \bigcirc(k^2)$	110,000 verify/s
RSA_{sign}	$2 \times \bigcirc(\frac{k}{2}) \times \bigcirc((\frac{k}{2})^2) = \bigcirc((\frac{k}{2})^3)$	6,000 sign/s
DSA_{verify}	$\bigcirc(q \times k^2)$	10,000 verify/s
DSA_{sign}	$\bigcirc(q \times k^2)$	11,000 sign/s
$DSA_{generation}$	$\bigcirc(q \times k^2)$	———

System Creation

This phase includes all the operations needed to set up the system.

Our Proposal: The creator of the network self-signs its certificate with its PK. After that, starts the on-the-fly P2PSIP application and waits for incoming connections.

- *Exchanged Messages:* Zero.
- *Operational Cost:* One RSA_{sign} to sign its certificate in the creator. Total: $\bigcirc((\frac{k}{2})^3)$ bit operations.

Dealer Threshold Crypto: The creator acts as a trusted dealer creating the Master Key (MK) of the system and sending the shares to the other $n-1$ administrative peers. After that, the distributed admission protocol starts for the administrators.

- *Exchanged Messages:* Each administrative peer exchange one message with the dealer to receive the MK share ($n-1$ messages). Then, $\bigcirc(t^2)$ messages should be exchanged to sign each one of the certificates of them [111], so in total: $n-1+\bigcirc(nt^2) = \bigcirc(nt^2+n)$ messages.
- *Operational Cost:* The creator generates the MK and the shares for the users (One $DSA_{generation}$). Also the signature of each administrator's certificate costs $\bigcirc(t^2+t)$ exponentiations to the system [111]. In total: $\bigcirc(q \times k^2) + n \times \bigcirc(t^2+t) \times k^3 = \bigcirc(qk^2 + (nt^2+nt)k^3)$ bit operations.

Distributed Threshold Crypto: In this approach the Master key is created in a distributed way, following the distributed key generation protocol presented in [264].

- *Exchanged Messages:* Each administrative peer sends $\bigcirc(n+t)$ messages [112] for the distributed generation of the shares. Also, $\bigcirc(t^2)$ messages should be exchanged to sign each one of the certificates of them [111], so in total: $\bigcirc(n^2+nt)+\bigcirc(nt^2) = \bigcirc(n^2+nt^2+nt)$ messages.

- *Operational Cost:* Each administrative peer performs $\bigcirc(nt)$ exponentiations due to the distributed protocol [112]. Also the signature of each certificate costs $\bigcirc(t^2 + t)$ exponentiations to the system [111]. So, in total: $n \times \bigcirc(nt) \times k^3 + n \times \bigcirc(t^2 + t) \times k^3 = \bigcirc((n^2t + nt^2 + nt)k^3)$ bit operations.

IP Based: The creator of the network self-signs its certificate with its PK. After that, starts the on-the-fly P2PSIP application and waits for incoming connections. Her ID is the hash of her IP address.

- *Exchanged Messages:* Zero.

- *Operational Cost:* One RSA_{sign} to sign its certificate, in the creator. In total: $\bigcirc((\frac{k}{2})^3)$ bit operations.

Shared Secret: The creator of the network self-signs its certificate with its PK. After that, starts the on-the-fly P2PSIP application and waits for incoming connections. Her ID is the hash of her PK address.

- *Exchanged Messages:* Zero.

- *Operational Cost:* One RSA_{sign} to sign its certificate, in the creator. In total: $\bigcirc((\frac{k}{2})^3)$ bit operations.

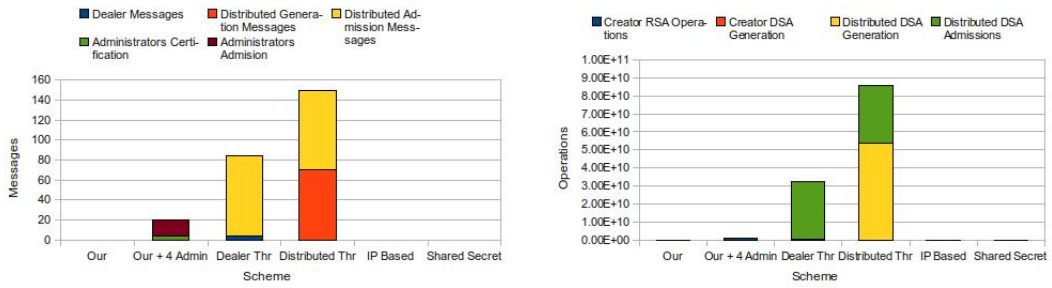
Figure 4.4 graphically represents the particular cost of network creation for the scenario $(n = 5, t = 2)$. Apart from the five commented proposals, we include the cost of setting up our proposal with four admin admissions for a better comparison with the threshold schemes that include the admins' admission in the setting up of the system. Focusing in the results, as we can see, the network creation cost of our proposal is much lower than threshold schemes and has a low overhead in comparison to IP and shared secret based schemes.

System Scalability

This phase includes all the operations needed to adapt the system to the increasing number of peers.

Our Proposal: The creator of the network generates an AC for each new administrative peer and sends it to all the other administrative peers (including the new member) to inform them that there is a new administrative peer.

- *Exchanged Messages:* The creator sends one broadcast message with the AC. Cost: 1 message.



(a) Messages.

(b) Operations.

Figure 4.4: *System Creation Cost.*

- *Operational Cost:* One RSA_{sign} to sign the AC, in the creator. Cost: $\bigcirc((\frac{k}{2})^3)$ bit operations.

Dealer Threshold Crypto: The creator creates new shares of the MK and redistribute them among the new $n-1$ administrative peers.

- *Exchanged Messages:* Each administrative peer exchange one message with the dealer to receive the MK share. In total: $n-1$ messages.
- *Operational Cost:* Cost of creation of the new shares, one $DSA_{generation}$. In total: $\bigcirc(q \times k^2)$ operations.

Distributed Threshold Crypto: The only way to adapt a decentralized threshold cryptography scheme is to redistribute the shares.

- *Exchanged Messages:* Each administrative peer sends $\bigcirc(n+t)$ messages for the shares' redistribution [112], so: $\bigcirc(n^2+nt)$ messages in total.
- *Operational Cost:* Each administrative peer has perform $\bigcirc(nt)$ exponentiations for the shares' redistribution [112], so: $\bigcirc(n^2tk^3)$ bit operations.

IP Based: No needed.

Shared Secret: No needed.

Figure 4.5 graphically represents the particular cost of network adaptation for the scenario $(n=5, t=2)$. As we can see, the network adaptation cost of our proposal and the dealer-based threshold cryptography is very low, while in the distributed threshold proposal it is really high.

Admission

This phase includes all the operations needed to admit a new peer in the system.

(a) Messages.

(b) Operations.

Figure 4.5: *System Adaptation Cost.*

Our Proposal: The new peer contacts one of the administrative peers that, if the new peer satisfies the access control policy, certifies its to access the network. Also, the administrator informs the other administrators of this admission.

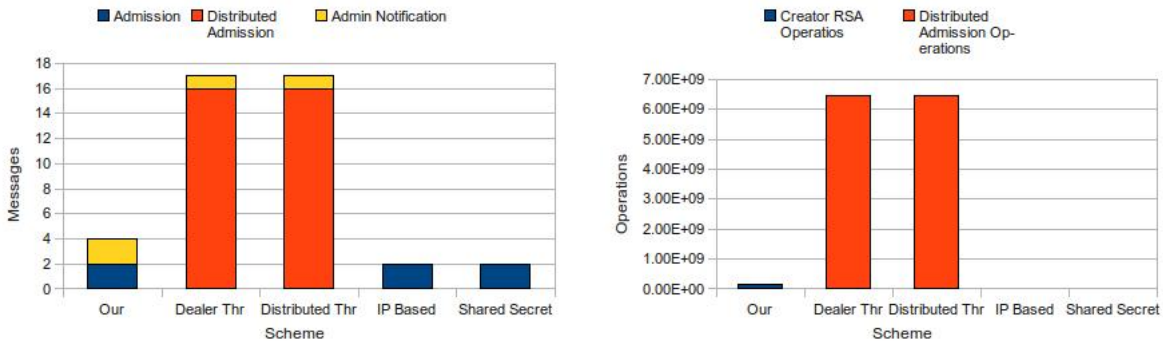
- *Exchanged Messages:* The new peer sends a message to one of the administrative peers that replies with the certificate or a negative answer (*2 messages*). Also, the administrator first informs and then confirms the admission to the other administrators (2 broadcast message). So, in total : 4 messages.
- *Operational Cost:* One RSA_{sign} to sign the certificate. So: $\bigcirc((\frac{k}{2})^3)$ bit operations.

Threshold Crypto: The new peer broadcast her joining request to all the administrative peers. After that, the threshold admission protocol [263] starts. Finally, if the admission was correct, an extra broadcast message should be send to inform the administrative peers that did not participate in the admission about it.

- *Exchanged Messages:* The admission of a new peer cost $\bigcirc(t^2)$ messages to the network [111].
- *Operational Cost:* The cost of an admission is $\bigcirc(t^2+t)$ exponentiations [111], so: $\bigcirc(t^2k^3)$ bit operations in total.

IP Based: The new peer contacts one of the peers of the network to start the bootstrapping process.

- *Exchanged Messages:* The new peer sends a message to one of the peers of the network to start the bootstrapping process that replies with an acknowledge before starting the bootstrap process. In total: 2 messages.



(a) Messages.

(b) Operations.

Figure 4.6: *System Admission Cost.*

- *Operational Cost:* Zero.

Shared Secret: The new peer contacts one of the peers of the network that, if the new peer proves the possession of the shared secret, starts its bootstrapping process.

- *Exchanged Messages:* The new peer sends a message to one of the peers of the network that replies with an affirmative or negative answer before starting the bootstrap process. In total: 2 message.

- *Operational Cost:* Zero.

Figure 4.6 graphically represents the particular cost of the system admission for the scenario $(n = 5, t = 2)$. As we can see, the cost of our proposal is much lower than threshold schemes and has a very low overhead in comparison to IP and shared secret based schemes.

Maintenance

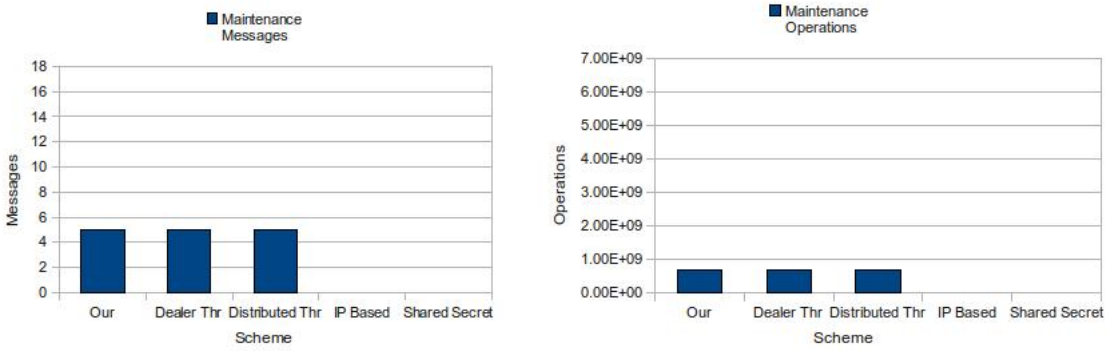
This phase includes all the operations needed for the maintenance of the access control scheme used.

Our Proposal: Each period γ of time administrators actualize their remote DDACLs.

- *Exchanged Messages:* One message per administrative peer to update their DDACL, in total: n messages.

- *Operational Cost:* One RSA_{sign} per administrator to sign and actualize its copy of the DDACL in the network, so: $\bigcirc(n(\frac{k}{2})^3)$ bit operations.

Threshold Crypto: No mechanism has been described in the literature to maintain the common information among the peers forming a threshold certification scheme. Our mechanism could be adapted for that purposes with the same cost.



(a) Messages.

(b) Operations.

Figure 4.7: *Network Maintenance Cost.*

IP Based: No needed.

Shared Secret: No needed.

Figure 4.7 graphically represents the particular cost for the scenario $(n = 5, t = 2)$. As we can see, the cost of our maintenance protocol is very low both in computational and communicational terms.

Certification Checking

This phase includes all the operations needed to check a certificate depending on access control scheme used.

Our Proposal:

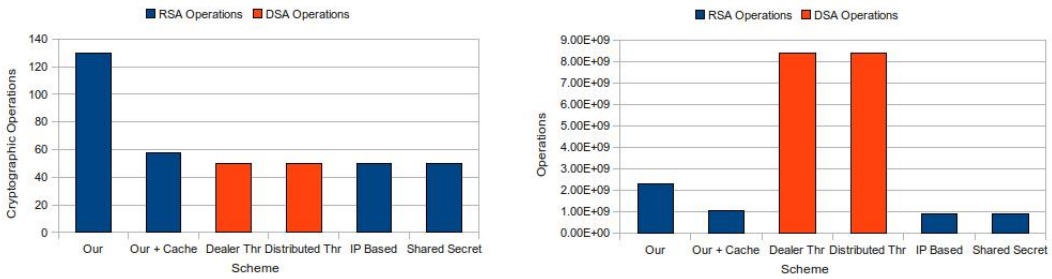
- *Exchanged Messages:* None.
- *Operational Cost:* One RSA_{verify} or three RSA_{verify} depending if the signer is the creator or another administrator, so: $\bigcirc(k^2)$ operations.

Threshold Crypto:

- *Exchanged Messages:* None.
- *Operational Cost:* One DSA_{verify}, so: $\bigcirc(q \times k^2)$ operations.

IP Based:

- *Exchanged Messages:* None.
- *Operational Cost:* One RSA_{verify}, so: $\bigcirc(k^2)$ operations.



(a) Cryptographic Operations.

(b) Operations.

Figure 4.8: *User Checking Cost of 50 Certificates.*

Shared Secret:

- *Exchanged Messages:* None.
- *Operational Cost:* One RSA_{verify}, so: $\bigcirc(k^2)$ operations.

Figure 4.8 graphically represents the cost of a user to check 50 certificates in the scenario $(n = 5, t = 2)$. For a better illustration of the efficiency of the improvement achieved using an admin cache, in Figure 4.8a we represent the cost in main cryptographic operations while Figure 4.8b represents the absolute cost in bit operations. Again, we can see how the cost of our proposal is much lower than threshold schemes and has a very low overhead (specially when admin caches are used) in comparison to IP and shared secret based schemes.

As we can see, both in the theoretical analysis and in the graphical representation of the scenario $(n = 5, t = 2)$, the cost induced for the extra security (that only makes sense in very hostile scenarios) of threshold cryptography is very high. Also, it is clear that the extra security added by our proposal in comparison to IP based and shared secret schemes comes at a very low cost. Finally, the efficiency of our DDACL maintenance protocol is displayed.

4.4 Conclusions and Contributions

Several alternative schemes have been presented in the literature to try to solve the access control problem in P2PSIP systems when the possibility of including a logically centralized authority (either online or offline) in the system is not possible. However, the initial analysis conducted of the characteristics of all these alternatives shows that most of them are not suitable for on-the-fly P2PSIP systems. Also, the most used schemes for on-the-fly P2PSIP systems have some drawbacks: shared secret and IP based systems are insecure and threshold cryptography consumes too much resources.

From the deficiencies of the existing schemes, in this chapter we have presented a new access control proposal for on-the-fly P2PSIP systems which is based on the recently published Internet Attribute Certificate Profile for Authorization. In our proposal, the creator of the network initially acts as a CA issuing certificates for each new user of the network and, as the size of the systems grows, uses ACs to distribute the access control of the system among several trusted users.

The evaluation conducted shows that our proposal greatly improves the security of IP based and shared secret schemes with no infrastructure cost and a minimal performance charge. Also, it achieves a similar level of security than threshold cryptography in non very hostile scenarios while highly reducing it computational and communicational cost. All this facts position our proposal as an alternative to access control for on-the-fly P2PSIP systems in non very hostile environments where performance is a factor key.

From the results of the research presented in this chapter derives the paper *Secure Access Control for on-the-fly P2P Systems* that is under review in the journal *Computer Networks*.

CHAPTER 5

NEW AUTHORIZATION FRAMEWORK FOR P2PSIP SYSTEMS

> You cannot pass! I am a servant of the Secret Fire, wielder of the Flame of Anor. The dark fire will not avail you, Flame of Udun! Go back to the shadow. You shall not pass!
>
> J.R. Tolkien - The Lord of the Rings

Since the beginning of the computer era, access control has been one of the major concerns in the development of secure computer systems. At first, the access control models were intended for homogeneous stand-alone computer systems. The more representative examples of these traditional approaches are: DAC (Discretionary Access Control) [267], MAC (Mandatory Access Control) [268] and the more recent and flexible RBAC (Role-Based Access Control) [269]. Later, with the appearance of networked and distributed systems new alternatives appeared for this heterogeneous systems, among which stress the Kerberos authentication system [270], Microsoft .NET passport [271] and the Public Key Infrastructure [262].

In the particular case of Peer-to-Peer systems, due to its decentralized architecture and the difficulty of having the necessary infrastructure to implement any of the previous cited models for distributed systems, access control is an even more challenging task. Concerning this, several researches have been carried out to try to achieve a secure access control in this kind of networks (as we have already analyzed in this thesis) such as cryptographic puzzles [101], CAPTCHAs [102], web of trust [105], shared secret [57] or offline certification [65].

Regardless of the specific used access control model, one property is common to all of them: PKCs (either self-signed or by a Trusted Third Party) are used for the users' identification. These PKCs represent two roles: user's authentication (who the user is, as for example Pilar

Touceda) and user's authorization (privileges of the user in the network: usernames allowing her to join the network and to have a location in the ID space to store her resources, node-IDs establishing her location in the network and the resources she is responsible for, storage quota limiting the amount of data she can store in the network, etc.). However, the fact that PKCs are both used for authentication and authorization of users is not a good idea [261]. Including the identity and the privileges of a user (username, node-ID, services contracted, etc.) in the same certificate determines that both the identity and any privileges should have the same lifetime and should be issued by the same authority. Also, every time a new privilege is added, removed or changed the certificate should be revoked and a new one should be created. This authorization approach is inefficient and does not consider scenarios where the identity of the users is granted by an external trusted certification authority.

In the same way, within these approaches, the access to the network resources is controlled by the definition of access control lists ACLs. ACLs perform well in operating systems or client-server architectures but not in P2P networks. In order to be usable, ACL's content has to be made public (to let the reader verify that the resource she is accessing has been created by an authorized user) revealing all the user's privileges over a resource and, therefore, attacking user's privacy. Also, the fact that all the resource's replicas should be contacted in order to modify the resource's ACL for granting a new user privileges over it, make this approach inefficient.

Finally, despite the fact that most of P2P applications use short-lived PKCs, the different nature of the privileges that can be assigned in a P2P system and the existence of applications with special security requirements, would make revocation of privileges desirable in same cases. Unfortunately, existing alternatives based on centralized servers (like the Certificate Revocation List -CRL- servers [262]) or trusted intermediate authorities (such as the Online Certificate Status Protocol -OCSP- responders [272, 273]), that should be contacted each time a certificate has to be checked, are not an option for P2P networks.

With this in mind, and since the centralized authorization approaches like Kerberos or Microsoft .NET are not suitable for P2P networks, we consider the use of the new Attribute Certificate (AC) profile for authorization [89][261] as an alternative for the authorization in P2P networks. A few works, like [274] or [275], have already presented the advantages of using ACs (or similar digitally signed authentication tokens like in [276]) for authorization over traditional approaches in distributed environments; however, no previous research exist that discusses the AC model in P2P systems and presents a framework for authorization based on ACs in the special conditions distributed P2P networks present.

In this chapter, after presenting the deficiencies of traditional identity-based authorization models in P2PSIP networks, we present a complete framework for authorization on P2PSIP networks (extensible to other P2P systems) based on ACs that link the privileges of a user within

the system with her identity represented by a PKC. Also, we present a distributed revocation system that can be established within the P2PSIP network and does not need the intervention of any external server or trusted intermediate authority. As concluded in the evaluation section, this separation between authentication and authorization outlines a more flexible and secure authorization scheme while improving the efficiency of the assignment of privileges.

5.1 Authorization Discussion

The fact that early P2P systems were intended for file-sharing purposes determined the guidelines followed by the access control solutions in this kind of networks. Their open-nature and the free availability of the shared resources motivated that these proposals were more concerned about restricting the number of malicious nodes in the network than implementing an authentication and authorization mechanism per se. In early P2P systems this control was based on the generation of node-IDs by hashing a 'unique' property of each node, like its IP address [51][19] or its public key [60]. The use of cryptographic puzzles, first described in [101], was also proposed in the literature to control the access of nodes to the network [65]. Besides, it is worth mentioning other decentralized access control systems based on the use of CAPTCHAs [103], the web of trust [104], social networks [115] or a shared secret [57].

However, the paper titled *The Sybil attack* [70] shows that, without a logically centralized authority, it is impossible to limit the number of identities a user can obtain to access the network except under extreme and unrealistic assumptions of resource parity and coordination among entities. Taking into account this research, [65] and [96] propose the introduction of an offline centralized CA in the system that assigns to each user a X.509 PKC [89] binding a node-ID, chosen randomly by the server, to a public key generated by the client and her username. These proposals are the basis of the access control schemes followed by actual P2PSIP systems, like RELOAD [57].

Within these schemes, PKCs represent two roles: user's authentication (who the user is, as for example Pilar Touceda) and user's authorization (privileges of the user in the network: usernames allowing her to join the network and to have a location in the ID space to store her resources, node-IDs establishing her location in the network and the resources she is responsible for, storage quota limiting the amount of data she can store in the network, etc.). Also, access control over the P2P system's resources is build around these privileges included in the user's PKC. P2P systems, like OceanStore [167] or Fairsite [168], use local access control lists to determine the privileges of each user over an object. Each resource has an ACL associated that contains the PKs of the users authorized to access it. User's requests are digitally signed so that the responsible for the resource can check that the user's access is authorized. A similar approach is used in other systems, like RELOAD [57], that also can include a usage for shared

resources by using delegation ACLs [277]. Unfortunately, due the fact that in some cases the node responsible for a resource may be malicious and give free read access to all the users of the network to a private resource it is responsible for, an extra mechanism should be used in combination with ACLs: cryptography. If a user wants to store a private resource for her personal use, symmetric cryptography, such as the AES algorithm [15], could be used to encrypt the data before storing it in the network. On the other hand, if the private resource is intended for another user, like a voice-mail, it may be encrypted using the public key of the recipient, as described in [169]. In case the publisher wants the resource to be accessible by a group of users, three possibilities arise: to extend the scenario of a single recipient by storing one copy of the resource for each recipient encrypted with her corresponding public key, to store only one copy of the resource encrypted with a symmetric key and send a private message to each recipient with the location of the resource in the network and the key needed to access it [167] or to encrypt the symmetric key using the public key of all authorized readers and store the encrypted keys with the resource [168].

As we have seen, actual P2PSIP systems' access control is very concerned about the authentication of users. The inclusion in the system of a logically centralized authority grants the users' identity while the use of PKCs permits to easily secure the communications established among them. Unfortunately, user's authorization in P2PSIP networks has not been as widely discussed and the traditional identity-based authorization mechanisms present several drawbacks associated to the use of the same certificate (PKC) for both the authentication and authorization of users:

- The fact that the user's privileges are included within the certificate that also grants her access to the network determines that all the privileges will have the same duration, and the same as the certificate of identity of the user. This is not acceptable because we may allow a user to contract different services of the network (a voicemail, more storage, new identifiers, etc.) during different periods of time.

- Any change in any of the privileges already incorporated in the PKC forces the creation of a new PKC to update all the privileges.

- The inclusion of new privileges for a user (due to the inclusion of new resources, to the acquisition of new privileges, ...) also forces the creation of a new PKC.

- The use of short-lived certificates is recommend for P2PSIP systems. However, the fact that is enough a change in any of the privileges included in the user's PKC to make it invalid could force the use a very short duration that increases the load of the offline CA. Besides, the lack of a specific revocation method for P2P system makes the alternative

of using revocation unpractical due the cost of including the necessary infrastructure to implement traditional revocation systems.

- From the security point of view, we found necessary to separate the network infrastructure from the applications or services running over it. We need a schema where different providers could offer different services or applications to the users of the network, even if the network is ran by another provider company, using simple and standardized mechanisms.
- Since users' PKCs must be available to all the other users of the system to allow their authentication, including all the user's privileges in the same PKC attacks the user's privacy (need to know) by disclosing all her privileges.

In addition, the existing resource's access control mechanisms do not fulfill all the security and flexibility requirements by themselves. The usual combination of local control and encryption present several issues:

- This mechanism works well with simple access control policies like publicly accessible or private. However, the lack of a standardized format for ACLs and the fact that they were not designed with distribution systems in mind make the definition of more complicated access control policies in P2PSIP systems a difficult and application specific task.
- Due the fact that P2PSIP systems replicate contents in several locations of the network, a change in the ACL of one resource forces a change in all the replicas even if the content has not been modified. It is not efficient to have to contact all the responsible nodes for a resource-ID in a decentralized network each time we want to modify its access control policy and it should be avoided when it be possible.
- Delegation ACLs need to be public to permit users requesting a shared resource to check its integrity. This is a serious privacy issue.

Following these observations, in the next sections of this chapter we propose a new authorization framework for P2PSIP systems based on the use of attribute certificates to manage the privileges of the users over the resources of the network.

5.2 Authorization framework

Once we have addressed the deficiencies of existing authorization mechanisms for P2PSIP systems, we present a new structure of authorization for this kind of systems (Fig. 5.1), which is based on the recently published Internet Attribute Certificate Profile for Authorization [261] that outlines a clear differentiation between the concepts of authentication and authorization

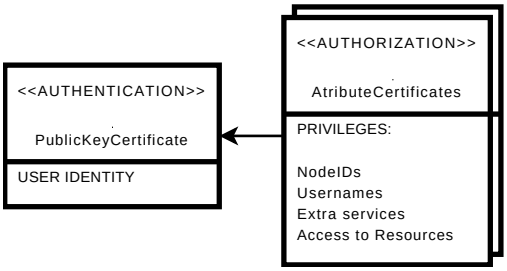


Figure 5.1: *Authentication and Authorization scheme.*

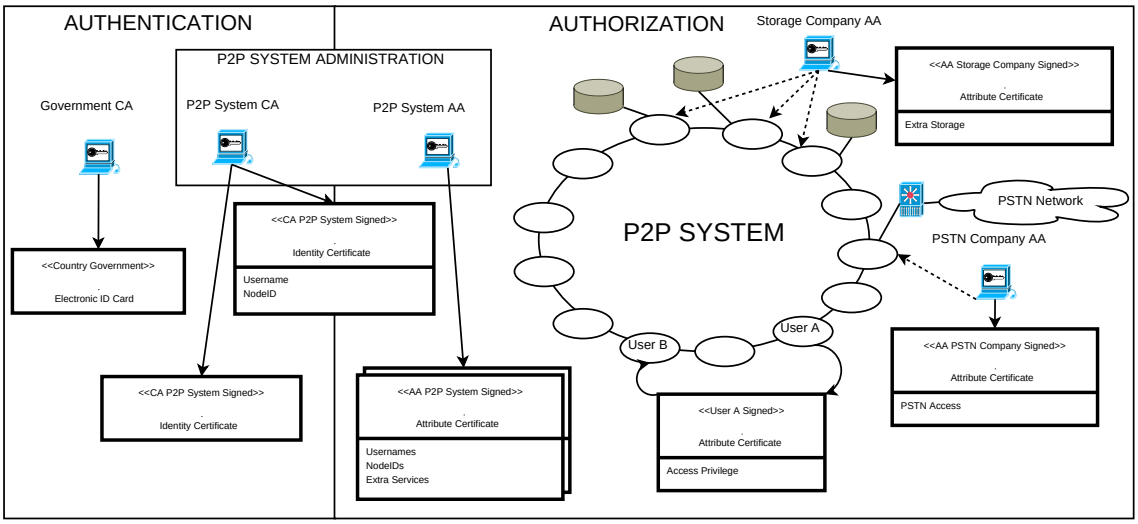


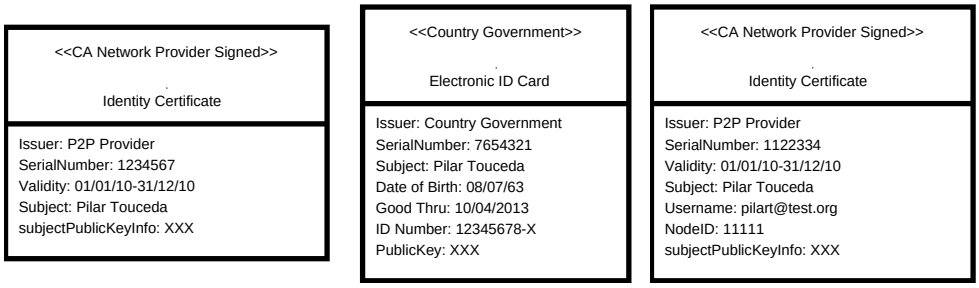
Figure 5.2: *User Authentication and Authorization architecture.*

of users. There is a difference between the concepts of authentication and authorization in the access control because the authentication of the users on the network must be solved first and the second step, and more complex, is to establish their privileges. The authorization is based on the authentication, but describes what a user is allowed to make, and therefore, defines the rights and privileges that a user has to perform a task. Figure 5.2 represents the architecture we describe during the rest of this section.

5.2.1 User Authentication

We assume that all the possible users of the system have a X.509 PKC compliant with the standard described in the RFC5280 [262] that grant their identity. This certificate could be issued using any of the existing certification models presented in the literature: by the system certification authority (either online or offline), by any external certification authority like the government of a country that issues an electronic ID card to all its citizens, etc.

It is desirable that these PKCs are only a proof of the user's identity and do not authorize any access to the network or its resources. This approach has two great advantages respect to



(a) Pure. (b) Hybrid.

Figure 5.3: *Examples of existing sources of authentication.*

the identity-based authorization models: first, the user identity certificate does not necessarily have to be issued by the CA of the system and can be created by an external organization (company, government, etc.) that the system's administrators believe it is reliable enough to ensure user's identity. Second, and as a consequence of the first advantage, a user can have a single identity certificate (as it should be because the identity of a user is unique) and use it as source of authentication with any system, instead of having to hold one different identity certificate for any of the systems she has access to. However, we are aware of the existence of already developed identity-based authorization models, such as the proposed standard for P2PSIP communications [57], where PKCs (issued by the system's CA) include, apart from the user's identity, a few privileges (specifically her username and node-ID). In such cases, our authorization model serves to complement the privileges already defined in the users' PKC.

Figure 5.3 shows examples of possible sources of authentication. Two pure models (PKCs only include information related to the users' authentication) are presented in Figure 5.3a: a PKC issued by the network provider and a PKC issued by a TTP such as a country government (electronic identity card). In turn, Figure 5.3b presents an hybrid model (PKCs also include users privileges apart from their identity). It is important to note that we do not expect here to describe in detail the content of these PKCs[1], but to present their more representative and relevant fields (*issuer*, *serialNumber* and *subjectPublicKeyInfo* in both proposals; and the extra privileges included in the hybrid proposal) for our authorization model.

5.2.2 User Authorization

Our authorization framework is based on the recently published Internet Attribute Certificate Profile for Authorization [261]. In it, authorization is granted using X.509 ACs [89], that associate privileges with the identity of the user defined in her PKC. ACs allow the privileges to

[1]The detailed description of the PKCs used in conjunction with our authorization scheme (whose structure vary depending on the specific P2PSIP system and its security requirements) is out of the scope of this paper.

have a different policy of certification, lifetime, etc. than the user's PKC. Also, they can be issued by several entities (Authorization Authorities, users that delegate a privilege to another user, etc.) different from the issuer of the user's PKC. In the rest of this section we introduce the privileges a user can have in a P2PSIP communication system, the data structures used in our proposal to do it and the process that must be follow to assign, revoke and use them.

Privileges

User authorization in P2P networks is based on privileges. These privileges can be classified in five major types:

- *Access to the System*: This privilege allows users to access the system and use its resources. It is normally granted by the assignment of a node-ID and, usually, a username[2] as we describe later in this paper. These credentials usually also determine the resource-IDs assigned to the user for her private use.

- *Access to Services offered by the system*: Apart from the access to the network itself, the system may offer extra services to its users like additional usernames or node-IDs, premium services like extra storage quota, etc. These privileges usually extend the number of resource-IDs a user has privileges over.

- *Access to Services offered by others within the system*: Other users or companies may offer extra services in the system, as for example including nodes in the system with large storage capacity to offer storage services to the system's users or connecting a node to a PSTN gateway to allow the users of a P2PSIP communication system to establish calls with the traditional telephony system.

- *Write Access Privileges*: In P2PSIP systems users are assigned some resource-IDs (based on the privileges assigned by the system, usually the hash of their usernames and node-IDs) where they can store, modify and share their resources. Also, users can delegate the privileges over these resource-IDs to other users of the system.

- *Read Access Privileges*: Users of the system must be able of specifying which users can read the resources they publish.

Data Structures Used for the Assignment of Privileges

Before describing how the different privileges presented below are assigned within our proposal, we introduce the data structures used in our system to do so.

[2]Excluding some special applications and entities, like gateways, that might only require a valid node-ID.

Attribute Certificates One of the most popular solution for the management of privileges is the use of attribute certificates. These certificates are supported by an Attribute Authority (AA). This kind of entities complement the functionalities of the CAs. But, instead of establishing certification of the identities associated with a particular public key, the AA allows to associate privileges to a PKC issued by another entity, with different policy of certification, lifetime, etc.

The concept of AC is thoroughly discussed in the ITU X.509 standard [89], which establishes its definition and structure, and in the recently published RFC5755 [261]. This idea arises from the problems associated with the certification of identity and privileges into the same certificate. PKCs allow the inclusion of privileges into the certificate through the use of the extension 'Subject directory attributes'. However, the problem with the PKCs is that they are designed for relatively long periods of time, specially when compared with the frequency of change of rights or privileges. If a PKC is also used for this purpose, it is necessary to make a new one containing such privileges, and then revoke it whenever the privileges augment, change or disappear.

Below we present the structure and fields of the of the ACs (following the profile standardized in [261]) used for our system:

- *acinfo.version*: Represents the version of the AC used. It should be $v2$ (1).
- *acinfo.holder.baseCertificateID.issuer* and *acinfo.holder.baseCertificateID.serial*: Points to the PKC to which this AC applies (the user who receives the privileges). The *issuer* field represents the issuer of the holder's PKC (creator of the user PKC) while the *serial* field represents its serial number. Both must be equal to the fields in the PKC of the holder (user who receives the privileges).
- *acinfo.issuer.v2Form.issuerName*: Name (in its PKC) of the issuer (entity assigning the privileges).
- *acinfo.signature*: Algorithm identifier used to validate the AC. It can be any of the defined in the standard [89].
- *serialNumber*: Serial number of the AC. The pair issuer/serialNumber must be unique.
- *acinfo.attrCertValidityPeriod.*{*notBeforeTime,notAfterTime*}: Period of validity of the certificate that specify the lifetime of the privileges included in it.
- *acinfo.attributes.type[].value[]*: Set of privileges the AC gives to the holder (user) [3].

[3]The structure of these privileges is not described here since they are application specific.

- *acinfo.extensions.authorityKeyIdentitifer.keyIdentifier*: resource-ID where the PKC of the AC's issuer is stored. This field and the next two allow to easily find the PKC of the AC's issuer in order to check the validity of the AC.
- *acinfo.extensions.authorityKeyIdentitifer.authorityCertIssuer*: Issuer of the PKC of the AC's issuer.
- *acinfo.extensions.authorityKeyIdentitifer.authorityCertSerialNumber*: Points to the serial Number of the PKC of the AC's issuer.
- *acinfo.extensions.crlDistributionPoints*: This field must only be included if revocation of this certificate is possible and must point to the resource-ID where the revocation information can be found.
- *acinfo.extensions.noRevAvail*: This field must only be included if revocation of this certificate is not possible. It includes no data.
- *signatureValue*: Signature of the issuer over the certificate.

Figure 5.4 represents an example where a user (Urbano Suárez) assigns a privilege to another user (Pilar Touceda) by issuing her an AC and describe the relation of the fields of the AC presented above with the PKC of the users.

The structure of the CRL (described below) used to revoke this privilege is also presented.

Certificate Revocation List Below we present the structure and fields of the of the CRLs (following the profile standardized in [262]) used in our system for the revocation of privileges (Figure 5.4):

- *tbsCertList.version*: Represents the version of the CRL used. It should be $v2$ (1).
- *tbsCertList.signature*: Algorithm identifier used to validate the CRL. It can be any of the defined in the standard [89].
- *tbsCertList.issuer*: Name (in its PKC) of the issuer of the CRL (entity revoking privileges).
- *tbsCertList.thisUpdate*: Date of the CRL.
- *tbsCertList.nextUpdate*: Date of the next CRL update. It value should be the same than the *notAfterTime* field of the AC to which this CRL is related. This allows to issue a CRL only if the AC is revoked and not periodically like in traditional client-server systems.
- *tbsCertList.revokedCertificates[].{userCertificate,revocationDate}*: This field is a list (that could be empty) of the revoked certificates. Each object of list contains two fields: the serial number of the revoked certificate and its date of revocation.

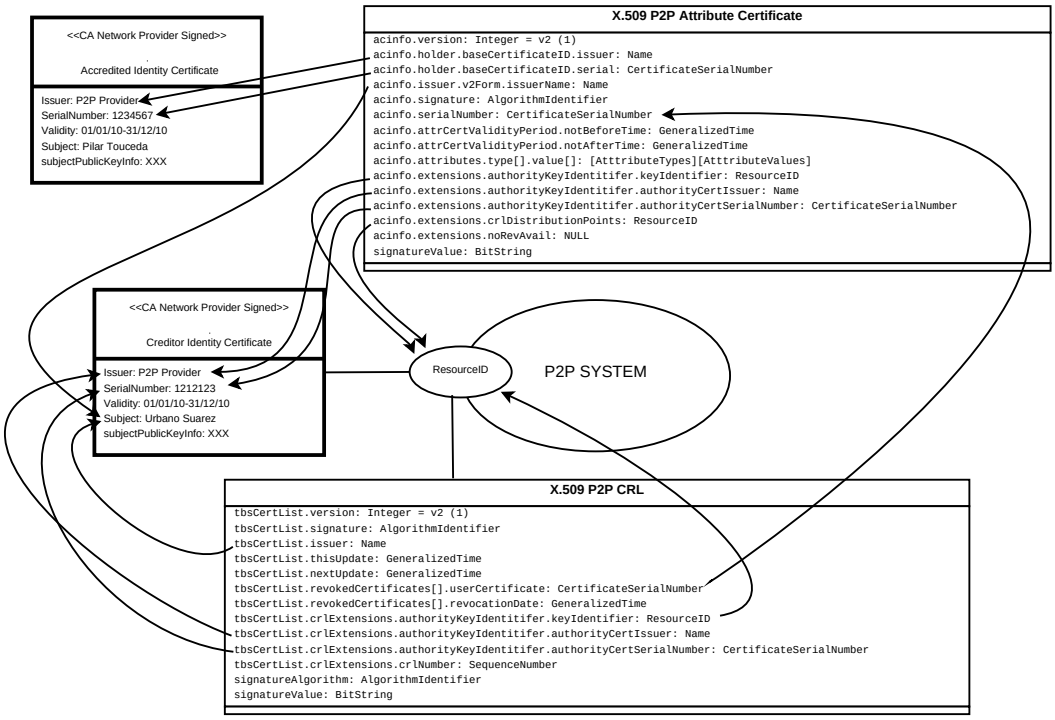


Figure 5.4: *Attribute Certificate and CRL structure for P2PSIP networks.*

- *tbsCertList.crlExtensions.AuthorityKeyIdentifier.keyIdentifier*: Points to the resource-ID where the PKC of the CRL's issuer is stored. This field and the next two allow to easily find the PKC of the CRL's issuer in order to check the validity of the CRL.

- *tbsCertList.crlExtensions.AuthorityKeyIdentifier.authorityCertIssuer*: Issuer of the PKC of the CRL's issuer.

- *tbsCertList.crlExtensions.AuthorityKeyIdentifier.authorityCertSerialNumber*: This field points to the serial number of the PKC of the CRL's issuer.

- *tbsCertList.crlExtensions.crlNumber*: Sequence number of the CRL.

- *signatureAlgorithm*: Algorithm identifier used to validate the CRL. It can be any of the defined in the standard [89].

- *signatureValue*: Signature of the issuer over the CRL.

In traditional client-server systems, CRLs are related to several certificates (contain revocation information of all the users of the system). However, the specific nature of P2PSIP systems recommends to use them in a different way. In our approach, a CRL is issued independently for each privilege and includes its revocation information.

Similarly, in client-server approaches CRLs are issued periodically, something unsuitable for P2PSIP systems. In our approach, a CRL is only issued when any of the privilege it references has been revoked to prevent the issuer of the privilege from having to periodically contact the node responsible for the privilege. This is principally critical when the issuer of the privilege is an external centralized entity, such as an AA issuing the user's usernames. To this end, *nextUpdate* field in CRLs has to be the same that *notAfterTime* field of the AC to which the CRL is related. This way, if the AC is not revoked, its issuer does not have to contact the responsible node for the CRL to update it. Besides, the fact that the CRL is stored in several locations, due to replication, that can be checked by a user in order to ensure that the received CRL is the last issued, prevents a malicious node responsible for the CRL to provide a user with an old and invalid one or deny its existence.

Resource-ID Structure In P2PSIP systems users are assigned some locations of the network (resource-IDs) where they can store, modify and share their resources. These resource-IDs are derivation of some of the user's privileges, such as her node-ID (resource-ID = Hash(node-ID)) or her username (resource-ID = Hash(username)). Besides, despite its name seems to represent a single entity, each resource-ID may content several resources of different types. Despite it was designed with P2PSIP in mind, we want our framework to be application independent, and therefore we are not going to define neither the specific resource-IDs each user has privileges over nor the kind or the amount of data they can contain, but only assume that each user, based on the application's specifications used, has privileges over certain resource-IDs. The only requirement of our proposal is that for each resource (not resource-ID) at least four different policies can by defined at creation (or modification) time by its owner: read-public (resource publicly readable), read-private (resource only readable by the owner or delegated users), write-public (resource publicly writable) and write-private (resource only writable by the owner or delegated users). However, this do not prevent our proposal to be used with systems having more access control policies while they have the four required.

Assignment of Privileges

Following we describe how the presented privileges are assigned in our proposal using the described ACs. Table 5.1 summarizes the content presented in this section.

- *Access to the System*: If not already included in the user's PKC (using an hybrid authentication model), each user should have at least one AC containing a node-ID, and usually a username, that grants her access to the network. The AC should be signed by the system AA linking the node-ID (and the username) of the user with her identity PKC. It is very important to note that, as opposed to other privileges not offered by the system itself, this

AC should have been issued by the system AA or by an entity authorized by the system administrators in order to be valid. In order to get the privileges to access the system, a user creates a request including the credentials stipulated in the system's access control policy (like a username and a password) along with her PK or PKC (in case she already has got it from an authorized source). Then, she signs the request with her PrK and sends it to the system AA (or CA). The administrative entity checks the user's credentials and the message's signature to confirm that the user is in possession of the PrK relative to the presented PK (it should also check that the user's PKC is valid in case the user already provides one). If everything is correct, it creates a PKC (if the users has not already presented one issued by an authorized entity) that certificates the identity of the user. If an hybrid proposal is used, this PKC also include the necessary privileges (usually a username and a node-ID) to access the system, if not they are included in a separate AC. In case the user's credentials are not valid, it creates and signs a negative answer. Finally, the user has to check the authenticity of the answer by verifying the signature over the PKC (and AC) or the negative answer. It is important to note that the user already has and trusts the PKC of the administrative entities of the system before accessing the system and therefore does not have to check or retrieve them.

- *Services offered by the system*: Access to other services offered by the system are specified using independent ACs. Again, these ACs must be signed by the system AA or by an AA authorized by the system administrators in order to be valid. The process to obtain these privileges is very similar to the presented in the previous point. The user sends in a signed request her PKC along with the necessary credentials to get the new service to the system's AA. The administrative entity checks the credentials and the signature over the message (the user's PKC does not have to be checked again since its integrity was already checked during the assignment of privileges to access the system). If everything is correct, it issues an AC and sends it to the user, while if it is not, it sends a negative signed answer. Finally, the user has to check the authenticity of the answer by verifying the signature over the AC (or the negative answer).

- *Services offered by others within the system*: Access to services offered by other users or companies within the system are also specified using independent ACs. However, in this case, the ACs should not be signed by the system's AA but by the offerers of these services. The process that a user should follow is the same presented before with the difference that the contacted entity should be the offerer of the service not the system's AA. Also, the PKCs of the both involved entities should be checked since it cannot be ensured a previous trust between them.

- *Write Access Privileges*: As we have noted before, we assume the existence of at least two different write policies. When a resource is write-public all the users can write it. However, when it is write-private only the owner have write privileges over it. In the latter case, users can delegate their write privileges to others users of the system by issuing them ACs. When a user A wants to grant write access over a resource she controls to another user B, user A creates an AC_2 pointing to B's PKC and sends it in conjunction with her credentials (PKC or PKC + AC_1) to user B. In order to check that user A has in fact the privileges A wants to delegate, user B checks that A's PKC has been signed by an authorized source and AC_1 has been signed by the system AA and points to A's PKC. Also, in case revocation of user's credentials to access the system (AC_1) is available, user B should also check A's CRL. Finally, by checking that the signature over the issued AC_2 corresponds to A's PrK, user B can be sure that the delegation is valid. It is important to note that due the fact that several different data resources could be stored in the same resource-ID, AC_2 should not only specify to which resource-ID it affects but also to which specific data within this resource-ID. Extra specifications can be included in the AC like the amount of data the user can write, if she can also delegate this privilege to another user, etc. If extra policies exist in the system, the same method can be used to delegate them. For example, delegation to a threshold group can be done by issuing an AC pointing to the PKC related to the group key.

- *Read Access Privileges*: Again, for read access we assume the existence of at least two read policies. When a resource is read-public all the users have access to it. However, when it is read-private, the owner can delegate her read privileges to others users of the system by issuing them ACs. When a user A wants to grant read access over a resource she controls to another user B, user A creates an AC_2 pointing to B's PKC and send it in conjunction with her credentials (PKC or PKC + AC_1) to user B. In order to check that user A has in fact the privileges A wants to delegate, user B checks that A's PKC has been signed by an authorized source and AC_1 has been signed by the system AA and points to A's PKC. Also, in case revocation of user's credentials to access the system (AC_1) is available, user B should also check A's CRL. Finally, by checking that the signature over the issued AC_2 corresponds to A's PrK, user B can be sure that the delegation is valid. Unfortunately, this approach is not always secure. If all the nodes where trusted, using only ACs to specify the read privileges over the resources would be a good idea. However, in some cases a node responsible for a resource may not be trusted and give free read access to all the users of the network to a private resource it is responsible for. In such cases, read permission's ACs should be complemented using encryption: user's private resources should be symmetrically encrypted, private resources intended for a specific user

Table 5.1: *Assignment of privileges.*

ASSIGNMENT OF PRIVILEGES	
Privilege	**Credential**
Access to the system	Public Key Certificate or Attribute Certificate
Services offered by the system	Attribute Certificate
Services offered by others	Attribute Certificate
Write access	Attribute Certificate
Read access	Attribute Certificate + Encryption

should be asymmetrically encrypted with the user's PK, and private resources intended for a group of users should be symmetrically encrypted with a key shared by all the members of the group. In the latter case, extra information (the symmetric key, encrypted with the delegate's PK, needed to read the resource) should be made accessible to the delegate (either storing it with the resource or sending it to him). More specific access control techniques, like when a consensus among several users is needed to read a private resource, can be specified by encrypting it using a threshold secret sharing scheme and issuing an AC pointing to the PKC related to the group key. Finally, it is important to note that due the fact that several different data resources could be stored in the same resource-ID, AC_2 should not only specify to which resource-ID it affects but also to which specific data within the resource-ID.

Revocation of Privileges

In this section we present a revocation scheme adapted to our proposal and describe how the different privileges can be revoked. A summary of our revocation scheme is presented in Table 5.2.

- *Access to the System*: It seems reasonable for us to use short-lived certificates for granting the user's privilege to access the system, usually related to some kind of subscription: daily, weekly or monthly. Nevertheless, it also may exist the possibility of systems giving this privilege a longer duration, like a year or more; mainly when an hybrid proposal is used and the user's privilege to access the system is included in the user's PKC. In such case, to have a revocation alternative is reasonable. Whatever is the proposal (pure or hybrid) used, this privilege should be provided by the system AA (or by an entity authorized by the system administrators) and, therefore, also should be the revocation information. To

take advantage of the P2P network facilities and to avoid the inclusion of extra entities in the system, we propose that the administrative entity store the revocation information relative to a user in the resource-IDs reserved for that specific user. The resource-ID where this information is stored is specified with the extension field crlDistributionPoints in the certificate but it can also be calculated in function of the specific network properties, like for example resource-ID = Hash(user's username) or resource-ID = Hash(user's node-ID), for cases like an hybrid proposal where the user's PKC do not have a crlDistributionPoints extension. We propose to use as revocation information a standardized Certificate Revocation List (CRL) [262] whose structure has been described before in this paper. In order to revoke a privilege of this kind, the administrative entity has to create a CRL and send it to the correspondent node (the administrative credentials do not have to be sent since the node should already have and trust them). The responsible node checks that the signature of the CRL is correct and replies with a message that confirms the operation. Finally, the administrative entity verifies the signature of the reply to check its authenticity (the administrative entity should already have the credentials of the user so she can avoid to send them in her reply). Replication of this information should be also performed, either by the administrative entity revoking the privilege or by the network topology plugin itself.

- *Services offered by the system*: Since this kind of privileges are also granted by the system administration, the same method described above to revoke the user's privilege to access the system should be used to revoke them.

- *Services offered by others within the system*: Users or companies offering extra services within the system do not have the privileges to store the revocation information related to the privileges they assign in the user's private resource-IDs, like system's AA has. Therefore, the revocation method presented before cannot be used for these privileges. However, a simple and similar method can be used. Since, they have control over the nodes offering these extra services they can directly communicate the revocation of privileges to them. In order to revoke a privilege of this kind, the user or company has to create a CRL and send it to the correspondent node (the company credentials do not have to be sent since the node should already have them). The node checks that the signature of the CRL is correct and replies with a message that confirms the operation. Finally, the user or company verifies the signature of the reply to check its authenticity (the offerer should already have the credentials of the nodes offering their services so they can avoid to send them in their reply).

- *Write Access Privileges*: User's write access privileges over her private resource-IDs are related to other user's privileges like a node-ID, a username or an extra storage quota

granted by an AC. Therefore, write access over these resource-IDs is revoked as soon as the privileges granting it are invalidated. Also, it is important to note that if the privileges of a user are invalidated this automatically revokes all the privileges she has delegated. In case a user wants to revoke the privileges other users have over her resources, two cases arise. For write-public resources a user can either change the resource to write-private or delete it. For write-private resources, the scenario is different. In order to revoke the privileges a user A has delegated to another user B, A stores a CRL in the resource-ID to which the delegation is related to inform the node responsible for it that the delegation is no longer valid. Therefore, in both cases, the owner of the resource has to send a message to the responsible for the resource including the action to be performed, change of policy or a CRL (user's credentials have not to be included in the message since they were included at the resource creation). The responsible node checks that the signature of the action is correct and replies with a message along with her credentials (PKC or PKC + AC) that confirms the operation (this time the credentials of the node responsible of the resource are needed since the responsible node might have changed). Also, in case revocation of user's credentials to access the system is available, the responsible node should check the user's CRL. Finally, the owner of the resource verifies the signature of the reply to check its authenticity. Replication of this information should be also performed, either by the user revoking the privilege or by the network topology plugin itself.

- *Read Access Privileges*: The main problem of read access privileges is that it is impossible to force a user to forget the contents she has read in the past. The most that can be done is to prevent her to have access to future versions of the resource. In the same way as write access, read access is related to other user's privileges and, therefore, revoked as soon as the privileges granting it are invalidated. In case a user wants to revoke the privileges other users have over her resources, two cases arise. For read-public resources a user can either change the resource to read-private or delete it. For read-private resources the scenario is different. In order to revoke the privileges a user A has delegated to another user B, A stores a CRL in the resource-ID to which the delegation is related to inform the node responsible for it that the delegation is no longer valid. Also, when symmetric encryption is used, the old key should be changed, the new key has to be made accessible to the remaining accredited readers (either storing it with the resource or sending it to them), the old content deleted and the new content encrypted with the new key. Therefore, in both cases the owner of the resource has to send a message to the responsible for the resource including the action to be performed, change of policy or a CRL (user's credentials have not to be included in the message since they were included at the resource creation). Besides, the new encrypted version of the resource should be sent, when necessary. The

Table 5.2: *Revocation of privileges.*

REVOCATION OF PRIVILEGES	
Privilege	**Location of Revocation Information**
Access to the system	resource-ID = Hash(Username)
	resource-ID = Hash(node-ID)
	resource-ID = acinfo.extensions.crlDistributionPoints
Services offered by the system	resource-ID = acinfo.extensions.crlDistributionPoints
Services offered by others	Locally stored by nodes offering services
Write access	Attached to resource-ID referenced by privilege
Read access	Attached to resource-ID referenced by privilege + Re-keying of data

responsible node checks that the credentials of the user are correct (including checking its CRL if revocation of credentials to access the system are available) and also is the signature over the action. Then it replies with a message that confirms the operation (this time the credentials of the node responsible for the resource are needed since the responsible node might have changed). Finally, the owner of the resource verifies the signature of the answer to check its authenticity. When encryption is used, the new key has to be made accessible to the remaining authorized users and used to create a new encryption of the resource. Replication of this information should be also performed, either by the user revoking the privilege or by the network topology plugin itself.

Use/Check of Privileges

Finally, we end the description of our authorization proposal by describing how users can make use of the privileges they are assigned.

- *Access to the System*: In order to join the system, a user has to send a join request to a node already member of it signed with her PrK and including her credentials (PKC or PKC + AC). The admitting node should check that the request is authorized (PKC, in the hybrid proposal, or PKC + AC have been signed by the system AA or by an entity authorized by the system administrators). Also, in case revocation of certificates is available, it should make sure that the user's credentials have not been revoked by checking her CRL. The admitting node should send the answer in a reply including it credentials (PKC or PKC + AC) signed with its PrK. Finally, the joining user should check this

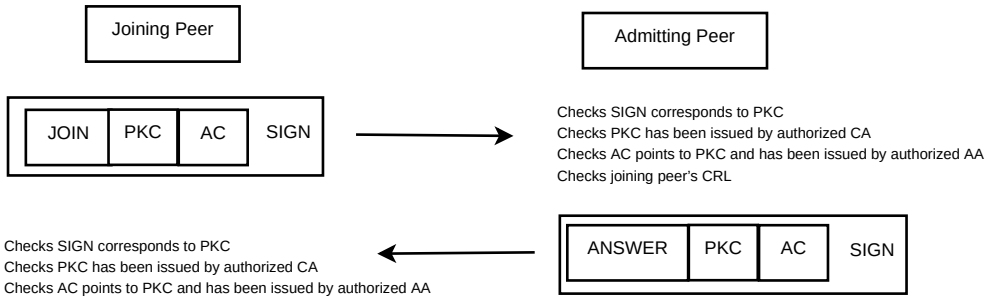


Figure 5.5: *Access to the System.*

answer (PKC, in the hybrid proposal, or PKC + AC have been signed by the system AA or by an entity authorized by the system administrators) to be sure that the node she has contact is really an authorized node of the system. Since the joining user does not have access to the network resources yet, she cannot check the revocation information related to the admitting node that could connect her to a fake network. However, as discussed later in the security point of the evaluation section, this could also be done if the node credentials are valid and, therefore, do not deteriorate the security of the system. Figure 5.5 illustrate the process to access the system.

- *Services offered by the system*: To make use of one of these services, a user has to send a request signed with her PrK. She should also include her credentials (PKC + AC) in the message to prevent the node having to do an extra communication to retrieve them from the network. The responsible node should check that the request is authorized (PKC + AC have been signed by the system AA or by an entity authorized by the system administrators). Also, in case revocation of certificates is available, it should make sure that the certificate has not been revoked by checking the user's CRL. The responsible node should send the answer in a reply including its credentials (PKC or PKC + AC) signed with its PrK. Finally, the requesting user should check this answer (PKC or PKC + AC have been signed by the system AA or by an entity authorized by the system administrators) to be sure that the node she has contacted is really the responsible for the requested service. Also, in case revocation of certificates is available, she should make sure that the node's credentials have not been revoked by checking the node's CRL.

- *Services offered by others within the system*: Again, to make use of one of these services, a user has to send a request signed with her PrK to the node responsible for it including her credentials (PKC + AC). Nodes offering extra services within the system should check that users requesting access to these services have an authorized AC (issued by themselves or the company ruling the node). Also, they should check in its local CRL database that the AC granting the privilege has not been revoked. The responsible node should send

the answer in a reply including it credentials (PKC + AC) signed with its PrK. Finally, the requesting user should check this answer (PKC have been signed by the system AA or by an entity authorized by the system administrators and AA has been signed by the offerer of the service) to be sure that the node she has contact is really the responsible for the requested service. Also, in case revocation of certificates is available, she should make sure that the node's credentials have not been revoked by checking the node's CRL.

- *Write Access Privileges*: In order to write a resource, a user has to send a write request to the node responsible for it. If the resource is write-public the request may neither be signed nor include the user's credentials. Nevertheless, if the resource is write-private the request should be signed with her PrK and include her credentials (PKC or PKC+AC). Also, in case extra security is needed, the content should be encrypted (using for example the secret key of a shared resource). In the write-private case the responsible node should check that the request is authorized (PKC or PKC + AC have been signed by the system AA or by an entity authorized by the system administrators). Also, in case revocation of certificates is available, it should make sure that the certificate has not been revoked by checking the user's CRL. The responsible node should send the answer in a reply including it credentials (PKC or PKC + AC) signed with its PrK. Finally, the requesting user should check this answer (PKC or PKC + AC have been signed by the system AA or by an entity authorized by the system administrators) to be sure that the node she has contact is really the responsible for the requested service. Also, in case revocation of certificates is available, she should make sure that the node's credentials have not been revoked by checking the node's CRL. For the case of delegated write privileges, the work-flow is almost the same with the exception of the credentials presented by the requester user that are her PKC and the AC created by the owner of the resource to grant her privileges over it. In this case, the responsible for the resource has to check that the requester user's PKC has been signed by an administrative entity and that the AC has been signed by the owner of the resource. Write access operation over a write-private resource is illustrated in Figure 5.6.

- *Read Access Privileges*: In order to read a resource, a user has to send a read request to the node responsible for it. If the resource is read-public the request may neither be signed nor include the user's credentials. Nevertheless, if the resource is read-private the request should be signed with her PrK and include her credentials (PKC or PKC + AC). The responsible node should check that the request is authorized (PKC or PKC + AC have been signed by the system AA or by an entity authorized by the system administrators or, in case it is a delegated read access, PKC has been signed by the system AA and AC has been signed by the owner of the resource). Also, in case revocation of certificates is available, it should make sure that the certificate has not been revoked by checking

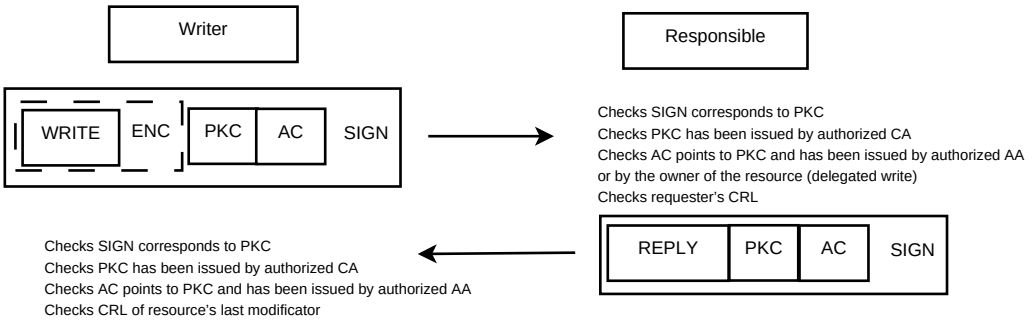


Figure 5.6: *Write operation over a write-private resource.*

the user's CRL (of both user in case of delegation). In both proposals (read-public and read-private) the responsible node should send the answer in a reply including the resource signed (if it is not an anonymous content) with the creator (or last changer) PrK along with her credentials. The requesting user should check the content has been signed by an authorized user (PKC or PKC + AC have been signed by the system AA or by an entity authorized by the system administrators or, in case the last modification of the resource was done by a delegated user, PKC has been signed by the an administrative entity and AC has been signed by the owner of the resource) to be sure that the content is valid. Finally, in case revocation of certificates is available, she should make sure that the last modificator's credentials have not been revoked by checking her CRL. Also, in case the resource is encrypted, user should decipher it.

5.3 Evaluation of Authorization Schemes

In this section we will conduct an evaluation of our authorization proposal. In order to evaluate it, not only in absolute terms but also in relative terms, we compare our proposal with the already presented RELOAD's identity-based approach. The evaluation starts with the flexibility of the proposals by presenting the main features our proposal has in comparison with identity-based approaches along with some scenarios of application. Then, we present an analysis of the performance of both models by studying their operational cost in terms of communicational and computational resources. This analysis shows that our proposal reduces the cost of the assignment of privileges, has only a slight overhead in its check and uses a very competitive distributed revocation mechanism. In a third point, we see that the infrastructure needed for the development of both proposal is the same. Then we point out the advantages of our proposal for using standardized methods. Finally, an study about the security of both schemes shows that our proposal not only maintains the security of identity-based authorization models but offers extra security functionalities.

5.3.1 Flexibility

The use of Attribute Certificates for authorization in P2PSIP represents an improvement in the user's privileges management and gives the system greater flexibility. Following we present the main features our proposal presents in contrast to identity-based authorization models along with examples of scenarios where these features could be useful.

- Separation of user's identity from user's privileges that may have different lifetime. Examples:
  1. A user of the system wants to sign up on several services of the network for different periods of time.
  2. A company wants to offer different services that can have different lifetime in a prepaid basis.
- External PKCs can be used as source for authentication. Examples:
  1. In certain company every employee holds a smartcard with a PKC that uniquely identifies herself. The company wants to establish an internal private P2PSIP system using these identity certificates as source of authentication.
  2. A telecommunication company wants to replace its old public switched telephone network (PSTN) in certain country for a more modern P2PSIP communication system using that country's Electronic Identity Card as source of authentication.
  3. A group of users (or a small company) want to develop a P2PSIP system but they lack the necessary infrastructure to securely authenticate users or do not want to carry on with the work of maintaining PKCs.
- Services may be provided by different entities (companies, users, etc.) using the same authorization scheme.
  1. A company wants one or more of its partners to be able to offer extra services within its P2PSIP system.
  2. Several companies want to offer services within the same P2PSIP system.
- Improved and anonymous access control policy over the system resources.
  1. A user publishes a resource in the network for its private use only. After a while she wants to give some friends access to this resource for different periods of time and with different privileges without making publicly accessible this delegation.

Table 5.3: *Operational performance.*

Operation	Performance
RSA_{verify}	110,000 verify/s
RSA_{sign}	6,000 sign/s

5.3.2 Performance

Following, we present an analysis of the cost of the main operations of our authorization proposal and we compare them with the cost of an identity-based approach. Cost is measured in terms of the number of communications that should established and the number of cryptographic operations that should be performed to carry out an operation. The computational cost to establish each communication is not included in our analysis because it depends on the protocol used (normal TCP/IP communication, DTLS/TLS, IPSec, etc.). However, since this cost is constant independently of the authorization proposal used, this simplification do not alter our comparison results. Before starting the analysis, we present some notation and the system configuration used during the test:

- We define the meter *Comm* to represent the cost in messages of establishing a communication between two entities of the system.
- The operational cost is measured in terms of cryptographic operations. Table 5.3 shows the performance[4] of main RSA-1024 operations.
- Since both proposals can use the same cryptography methods for the management of read access privileges, we do not include its use in our analysis.
- To simplify this evaluation, we assume that during the check of the user's privileges to access the system revocation information about them is not available. In the case that revocation would be available, one extra communication (inside the P2P system in our proposal and with the CRL server in the identity based one) would be needed to retrieve the revocation information of each certificate (AC in our proposal and PKC in the identity-based one) plus the correspondent operation (One RSA_{verify}) needed to check the CRL's integrity.
- The analysis of the identity-based approach is based on the RELOAD protocol [57], the usage for shared resources using delegation ACLs [277] and the CRL profile [262]. .

[4]Using the OpenSSL (version 0.9.8g) speed test in an Ubuntu 10.04 (lucid) 64-bits with kernel Linux 2.6.32-25 running over an Intel(R) Core(TM)2 Quad CPU Q8200 @ 2.33GHz with 4GB of RAM.

Assignment of Privileges

In this section we analyze the cost of the assignment of the existing privileges in both proposals:

- *Access to the System:* Cost of issuing the necessary credentials to access the system to a user.

  1. Communicational Cost: In both cases, one communication should be established between the user and the system AA or CA. So: One $Comm$.
  2. Computational Cost: In our proposal if the user does not already provides a PKC, she should sign her request (One RSA_{sign}), that should be checked by the AA (One RSA_{verify}) that creates the user's PKC and AC (Two RSA_{sign}) whose authenticity should be checked by the user (Two RSA_{verify}). If she already provides the PKC she should sign her request (One RSA_{sign}) and send it to the system's AA in conjunction with her PKC. The AA should check the request and the user's PKC (Two RSA_{verify}). Finally, the AA creates the user's AC (One RSA_{sign}) whose authenticity should be checked by the user (One RSA_{verify}). In the identity-based approach user should sign her request (One RSA_{sign}), that should be checked by the CA (One RSA_{verify}) that creates the user's PKC (One RSA_{sign}) whose authenticity should be checked by the user (One RSA_{verify}). So the cost is: $3RSA_{sign} + 3RSA_{verify}$ in our proposal when the user does not already have a PKC and $3RSA_{sign} + 2RSA_{verify}$ when she does. For the identity based approach, the cost is $2RSA_{sign} + 2RSA_{verify}$.

- *Services offered by the System:* Cost of issuing extra privileges (offered by the system) to a user.

  1. Communicational Cost: In our proposal, one communication should be established between the user and the system AA. The same communication should be established in the identity-based approach between the user and the system CA. Also, an extra communication is needed in the identity based approach to send the revocation information of the old user's credential to the CRL's distribution point. So: One $Comm$ in our proposal, Two $Comm$ in the identity-based approach.
  2. Computational Cost: In our proposal the user should sign her request (One RSA_{sign}), that should be checked by the AA (One RSA_{verify}) that creates the user's AC (One RSA_{sign}) whose authenticity should be checked by the user (One RSA_{verify}). In the identity-based approach the system administration should revoke the old user's PKC and issue a new one including the new privileges. So, apart from the operations needed to request the new PKC including the new privileges to the system CA (the

same that in our proposal to obtain the AC), the system CA has to create a CRL revoking the old user's PKC (One RSA_{sign}) whose authenticity should be checked by the CRL's distribution point (One RSA_{verify}). Finally, the CRL server has to create a signed answer (One RSA_{sign}) whose authenticity should be checked by the AA (One RSA_{verify}) to be sure that the revocation was performed. So: $2RSA_{sign}$+$2RSA_{verify}$ in our proposal and $4RSA_{sign} + 4RSA_{verify}$ in the identity-based approach.

- *Services offered by others within the system:* Cost of issuing extra privileges (offered by a user or a company) to another user.

  1. Communicational Cost: In our proposal one communication should be established between the user and the issuer to obtain the AC granting the privilege. In the identity-based approach one communication should be established between the user and the issuer to request the privilege and an extra communication is needed to inform the node offering the services about it. So: One *Comm* in our proposal and Two *Comm* in the identity-based approach.
  2. Computational Cost: In our proposal the user should create a signed request (One RSA_{sign}) including her credentials (PKC). Both, the signature over the request and the user's credentials should be checked by the offerer (Two RSA_{verify}) that replies with the user's AC (One RSA_{sign}) and its credentials (PKC) whose authenticity should be checked by the user (Two RSA_{verify}). In the identity based approach the user creates signed request (One RSA_{sign}) including her credentials (PKC) that should be checked by the offerer (Two RSA_{verify}). Then the offerer creates and signs an ACL (One RSA_{sign}) that grants the user privileges and sends it to the responsible node that, after checking its authenticity (One RSA_{verify}), replies with a signed answer (One RSA_{sign}) whose authenticity should be also checked by the offerer (One RSA_{verify}). Finally, the offerer creates a signed answer (One RSA_{sign}) and includes its credentials (PKC). Finally, the user checks the credentials of the offerer and its answer (Two RSA_{verify}) to confirm that the assignment of privileges was, in fact, done. So, the cost is $2RSA_{sign} + 4RSA_{verify}$ in our proposal and $4RSA_{sign} + 6RSA_{verify}$ in the identity based approach.

- *Write Access Privileges:* Cost of delegation of write access privileges.

  1. Communicational Cost: In our proposal a communication should be established to send the AC granting the privilege to the user receiving it. In identity-based approaches, a communication should be established to modify the ACL of the resource referenced by the delegation plus a communication to actualize each of the resource's

replicas[5]. Also, some kind of communication should established from the delegator to the user to inform her about the received privileges. So the cost is One $Comm$ in our proposal and $2 + NumReplicasComm$ in the identity-based approach.

2. Computational Cost: In our proposal the delegator should create an AC_2 (One RSA_{sign}) that should be sent to the delegate in conjunction with her credentials (PKC + AC_1). The delegate has to to check that the delegator has in fact privileges over the resource and that the delegation is valid (Two RSA_{verify} to check the delegator's PKC and AC_1 and another RSA_{verify} to check AC_2). Finally, the delegate has to sign (One RSA_{sign}) and send a reply confirming the reception of the delegation whose authenticity should be checked by the delegator (Two RSA_{verify} to check the delegate credentials, PKC + AC, and another RSA_{verify} to check the signature of the answer). In the identity based approach the delegator has to create a signed ACL (One RSA_{sign}) that should be sent with her credentials (PKC) to the node responsible for the resource and its replicas. The responsible for the resource and the replicas have to check that the ACL is valid by checking the delegator credentials and the signature over the ACL ($2 \times (1 + NumReplicas) RSA_{verify}$) and create signed replies ($(1 + NumReplicas) RSA_{sign}$) including their credentials, whose authenticity should be checked by the delegator ($2 \times (1 + NumReplicas) RSA_{verify}$). Finally, the delegator sends the signed ACL (to save the creation of an extra signed message) to the delegate that should check its authenticity (One RSA_{verify} to check the delegator's PKC and another RSA_{verify} to check the signature of the ACL) and reply with a signed confirmation of reception (One RSA_{sign}) whose authenticity should be checked by the delegator (One RSA_{verify} to check the delegate PKC and another RSA_{verify} to the check the signature of the answer). So, the cost is $2RSA_{sign} + 6RSA_{verify}$ in our proposal and $(3 + NumReplicas) RSA_{sign} + (4 + 4 \times (1 + NumReplicas)) RSA_{verify}$ in the identity based approach.

- *Read Access Privileges:* Cost to grant read access privileges. The process followed for the assignment of read access privileges is the same than for the assignment of write access ones. Therefore, the cost presented in this section derives from the previous point.

1. Communicational Cost: The communications needed for the delegation of read access privileges are the same than for the delegation of write access ones. So the cost is One $Comm$ in our proposal and $2 + NumReplicasComm$ in the identity-based approach.

[5]. These communications should be performed by the delegator to prevent a single malicious node (the responsible for the resource) from denying the delegation. Anyway, in case they were performed by the nodes involved in the replication themselves, they would suppose an overhead for the system too.

Table 5.4: *Performance Analysis of the Assignment of Privileges.*

		ASSIGNMENT PERFORMANCE	
		OUR	IDENTITY
Access System	COMM	*One* (centralized)	*One* (centralized)
	COMPT	$3RSA_{sign} + 2RSA_{verify}$ (centralized)	$2RSA_{sign} + 2RSA_{verify}$ (centralized)
Services System	COMM	*One* (centralized)	*Two* (centralized)
	COMPT	$2RSA_{sign} + 2RSA_{verify}$ (centralized)	$4RSA_{sign} + 4RSA_{verify}$ (centralized)
Services Others	COMM	*One*	*Two*
	COMPT	$2RSA_{sign} + 4RSA_{verify}$	$4RSA_{sign} + 6RSA_{verify}$
Write	COMM	*One*	$(2 + NumReplicas)$
	COMPT	$2RSA_{sign} + 6RSA_{verify}$	$(3 + NumReplicas)RSA_{sign} + (8 + 4NumReplicas)RSA_{verify})$
Read	COMM	*One*	$(2 + NumReplicas)$
	COMPT	$2RSA_{sign} + 6RSA_{verify}$	$(3 + NumReplicas)RSA_{sign} + (8 + 4NumReplicas)RSA_{verify})$

2. Computational Cost: Again, the cost is the same than for the assignment of write access privileges, so it is $2RSA_{sign}+6RSA_{verify}$ in our proposal. In the identity-based approach it is $(3 + NumReplicas)RSA_{sign} + (4 + 4 \times (1 + NumReplicas))RSA_{verify}$.

Table 5.4 summarizes this analysis related to the performance cost of the assignment of privileges.

Revocation of Privileges

As we have already noted in the description of our approach, we recommend the use of short-lived AC for the vast majority of scenarios. However, they might be some scenarios or special privileges that may need revocation. Following, we analyze the cost of revocation in our proposal in comparison to identity-based approaches.

- *Access to the System:* Cost of revoking the user's credential to access the system.

1. Communicational Cost: In our approach one communication is needed to send the revocation information from the administrative entity to the node responsible for

the resource plus a communication to actualize each of the resource replicas. In the identity based approach one communication is needed to send the revocation information to the CRL server. So: $1 + NumReplicasComm$ in our proposal and One $Comm$ in the centralized method used used by identity-based approaches.

2. Computational Cost: The CRL used to revoke the privilege should be signed (One RSA_{sign}) in both proposals. Also, the receptor of the CRL should check that its signature is valid (One RSA_{verify} and send a signed reply (One RSA_{sign}) whose authenticity should be checked by the sender of the CRL (One RSA_{verify}). In our proposal the receptors are the node responsible for the resource and its replicas while in the identity-based approach is the CRL server. It is important to note that in both approaches all entities involved in these operations already have the credentials (PKC or PKC + AC) of the others and do not have to check them again. So: $(2+NumReplicas)RSA_{sign}+2\times(1+NumReplicas)RSA_{verify})$ in our proposal and $2RSA_{sign}+2RSA_{verify}$ in the centralized method used by identity-based approaches.

- *Services offered by the System:* Cost of revoking extra services offered by the system.

1. Communicational Cost: In our approach one communication is needed to send the revocation information from the administrative entity to the node responsible for the resource plus a communication to actualize each of the resource replicas. In the identity-based approach one communication is needed to send the revocation information to the CRL server and another one to issue a new PKC to the user with all her old privileges but the revoked one. So: $1+NumReplicasComm$ in our proposal and Two $Comm$ in the centralized method used by identity-based approaches.

2. Computational Cost: In our approach the administrative entity has to create a CRL (One RSA_{sign}) and send it to the node responsible for the resource and its replicas that should check the authenticity of the CRL ($(1 + NumReplicas)RSA_{verify}$) and reply with a signed answer ($(1+NumReplicas)RSA_{sign}$). Finally the administrative entity should check the answers ($(1 + NumReplicas)RSA_{verify}$). In the identity-based approach the administrative entity has to create a CRL (One RSA_{sign}) and send it to the CRL server that should check its authenticity (One RSA_{verify}) and reply with a signed answer (One RSA_{sign}). Finally the administrative entity should check the answer (One RSA_{verify}). Also the administrative entity has to create a new PKC (One RSA_{sign}) for the user and send it to her. The user has to check the PKC (One RSA_{verify}) and reply with a signed answer (One RSA_{sign}). Again, the communication ends with the administrative entity checking the answer (One RSA_{verify}). It is important to note that in both approaches all entities involved in

these operations already have the credentials (PKC or PKC + AC) of the others and do not have to check them again. So: $(2 + NumReplicas)RSA_{sign} + 2 \times (1 + NumReplicas)RSA_{verify}$ in our proposal and $4 \times RSA_{sign} + 4 \times RSA_{verify}$ in the centralized method used by identity-based approaches.

- *Services offered by others within the system:* Cost of revoking extra privileges (offered by a user or a company) to another user.

  1. Communicational Cost: One communication should be established between the issuer of the CRL and the node offering the service in our proposal (except when they are the same entity). In the identity-based approach the same communication is needed to pass the modified ACL to the node responsible for the service. So: One $Comm$ in both proposals.
  2. Computational Cost: A CRL should be signed to revoke the privilege in our proposal and so does the ACL in the identity-based approach (One RSA_{sign}). The revocation information should be send to the node offering the service that should check its authenticity (One RSA_{verify}) and reply with a signed answer (One RSA_{sign}). Finally the issuer of the revocation information should check the answer (One RSA_{verify}). In both approaches all entities involved in these operations already have the credentials (PKC or PKC + AC) of the others and do not have to check them again. So: $2RSA_{sign} + 2RSA_{verify}$ in both proposals.

- *Write Access Privileges:* Cost of revocation of delegated write access privileges.

  1. Communicational Cost: In our proposal the delegator establishes a communication with the responsible for the resource to send her the CRL. Also, an extra communication should be generated to pass the CRL to each of the resource's replicas[5]. In the identity-based approach a communication should be established to modify the ACL of the resource referenced by the revocation plus an extra communication to actualize each of the resource's replicas[5]. So, the cost in both cases is $1 + NumReplicas$ $Comm$.
  2. Computational Cost: A CRL should be issued in our proposal by the delegator (One RSA_{sign}) and sent it to the responsible node and its replicas. They should check that the user's credentials (PKC + AC) and the signature over the CRL are valid $(3 \times (1 + NumReplicas)RSA_{verify})$ and reply with a signed answer $((1 + NumReplicas)RSA_{sign})$. Finally, the delegator should check that the credentials of the responsible and the replicas (PKC +AC) and the answers are valid $(3 \times (1 + NumReplicas)RSA_{verify})$. In the identity-based approach the delegator has

to create a ACL (One RSA_{sign}) and sent it to the responsible node and its replicas. They should check that the user's credentials (PKC) and the signature over the CRL are valid ($2 \times (1 + NumReplicas)RSA_{verify}$) and reply with a signed answer ($(1 + NumReplicas)RSA_{sign}$). Finally, the delegator should check that the credentials of the responsible and the replicas (PKC) and the answers are valid ($2 \times (1 + NumReplicas)RSA_{verify}$) So, the cost is $(2 + NumReplicas)RSA_{sign} + 6 \times (1 + NumReplicas)RSA_{verify}$ in our proposal and $(2 + NumReplicas)RSA_{sign} + 4 \times (1 + NumReplicas)RSA_{verify}$ in the identity-based approach.

- *Read Access Privileges:* Cost of revocation of read access privileges. The process followed for the revocation of read access privileges is the same than for the revocation of write access ones. Therefore, the cost presented in this section derives from the previous point.

  1. Communicational Cost: The communications needed for the revocation of read access privileges are the same than for the revocation of write access ones. So the cost is $1 + NumReplicas\ Comm$ in both approaches.
  2. Computational Cost: Again, the cost is the same than for the revocation of write access privileges, so it is $(2 + NumReplicas)RSA_{sign} + 6 \times (1 + NumReplicas)RSA_{verify}$ in our proposal and $(2 + NumReplicas)RSA_{sign} + 4 \times (1 + NumReplicas)RSA_{verify}$ in the identity-based approach.

Table 5.5 summarizes this analysis related to the performance cost of the revocation of privileges.

Use/Check of Privileges

In this section we analyze the cost of using the existing privileges in both proposals:

- *Access to the System:* Cost of accessing the system.

  1. Communicational Cost: In both proposals, when a user wants to join the system she has to establish a communication with a node already member of it. So the cost is One $Comm$ in both proposals.
  2. Computational Cost: In our proposal the joining user has to create a signed request (One RSA_{sign}) including her credentials (PKC + AC) and send it to a node already member of the system. The admitting node has to check that the user's credentials are valid (Two RSA_{verify}) and that the signature over the request is correct (One

Table 5.5: *Performance Analysis of the Revocation of Privileges.*

		REVOCATION PERFORMANCE	
		OUR	IDENTITY
Access System	COMM	$(1 + NumReplicas)$	*One* (centralized)
	COMPT	$(2+NumReplicas)RSA_{sign} + 2(1+NumReplicas)RSA_{verify})$	$2RSA_{sign} + 2RSA_{verify}$ (centralized)
Services System	COMM	$(1 + NumReplicas)$	*Two* (centralized)
	COMPT	$(2+NumReplicas)RSA_{sign} + 2(1+NumReplicas)RSA_{verify})$	$4RSA_{sign} + 4RSA_{verify}$ (centralized)
Services Others	COMM	*One*	*One*
	COMPT	$2RSA_{sign} + 2RSA_{verify}$	$2RSA_{sign} + 2RSA_{verify}$
Write	COMM	$(1 + NumReplicas)$	$(1 + NumReplicas)$
	COMPT	$(2+NumReplicas)RSA_{sign} + 6(1+NumReplicas)RSA_{verify})$	$(2+NumReplicas)RSA_{sign} + 4(1+NumReplicas)RSA_{verify})$
Read	COMM	$(1 + NumReplicas)$	$(1 + NumReplicas)$
	COMPT	$(2+NumReplicas)RSA_{sign} + 6(1+NumReplicas)RSA_{verify})$	$(2+NumReplicas)RSA_{sign} + 4(1+NumReplicas)RSA_{verify})$

RSA_{verify}). Then it creates a signed answer (One RSA_{sign}) including its credentials (PKC + AC) and sent it to the joining user. Finally, the joining user checks that the node's credentials are valid (Two RSA_{verify}) and that the signature over the answer is correct (One RSA_{verify}). In the identity-based approach the followed process is almost the same with the difference that the user credentials are only a PKC. So the cost is $2RSA_{sign} + 6RSA_{verify}$ in our proposal and $2RSA_{sign} + 4RSA_{verify}$ in the identity-based approach.

- *Services offered by the System:* Cost of accessing the services offered by the system.

  1. Communicational Cost: In both proposals, when a user wants to access a service offered by the system she has present her credentials to the node responsible for it. So, the cost in both proposals is One $Comm$.

  2. Computational Cost: In our proposal the user trying to access the service has to create a signed request (One RSA_{sign}) including her credentials (PKC + AC) and send it to the node responsible for the service requested. The responsible node has to check that the user's credentials are valid (Two RSA_{verify}) and that the signature over the request is correct (One RSA_{verify}). Then it creates a signed answer (One RSA_{sign}) including its credentials (PKC + AC) and sent it to the requester user.

Finally, the user checks that the node's credentials are valid (Two RSA_{verify}) and that the signature over the answer is correct (One RSA_{verify}). In the identity-based approach the followed process is almost the same with the difference that the user credentials are only a PKC. So the cost is $2RSA_{sign} + 6RSA_{verify}$ in our proposal and $2RSA_{sign} + 4RSA_{verify}$ in the identity-based approach.

- *Services offered by others within the system:* Cost of accessing extra services (offered by a user or a company) to another user.

  1. Communicational Cost: In both proposals, when a user wants to access a service offered by others within the system she has present her credentials to the node offering this service. So, the cost in both proposals is One $Comm$.
  2. Computational Cost: In our proposal the user trying to access the service has to create a signed request (One RSA_{sign}) including her credentials (PKC + AC granting her access to the service) and send it to the node responsible for the service requested. The responsible node has to check that the user's credentials are valid (Two RSA_{verify}) and that the signature over the request is correct (One RSA_{verify}). Then it creates a signed answer (One RSA_{sign}) including its credentials (PKC + AC that accredits it as offerer) and sent it to the requester user. Finally, the user checks that the node's credentials are valid (Two RSA_{verify}) and that the signature over the answer is correct (One RSA_{verify}). In the identity-based proposal the user trying to access the service has to create a signed request (One RSA_{sign}) including her credentials (PKC) and send it to the node responsible for the service requested. The responsible node has to check that the user's credentials are valid (One RSA_{verify}), that she appears in its ACL as authorized user and that the signature over the request is correct (One RSA_{verify}). Then it creates a signed answer (One RSA_{sign}) including its credentials (PKC) and sent it to the requester user. Finally, the user checks that the node's credentials are valid (One RSA_{verify}) and that the signature over the request is correct (One RSA_{verify}). So the cost is $2RSA_{sign} + 6RSA_{verify}$ in our proposal and $2RSA_{sign} + 4RSA_{verify}$ in the identity-based approach.

- *Write Access Privileges:* Cost of performing a write access operation.

  1. Communicational Cost: In both proposals, when a user wants to perform a write operation over a resource she has to establish a communication with the node responsible for it. So, the cost in both proposals is One $Comm$.
  2. Computational Cost: In our proposal the user trying to write the resource has to create a signed request (One RSA_{sign}) including her credentials (PKC + AC granting

her write privileges over the resource) and send it to the node responsible for the resource. The responsible node has to check that the user's credentials are valid (user's PKC has been signed by an administrative entity and an AC has been signed either by the system AA or by the owner of the resource that delegates access, Two RSA_{verify}) and that the signature over the write request is correct (One RSA_{verify}). Then it creates a signed answer (One RSA_{sign}) including its credentials (PKC + AC that accredit it as responsible) and sent it to the requester user. Finally, the user checks that the node's credentials are valid (Two RSA_{verify}) and that the signature over the answer is correct (One RSA_{verify}). In the identity-based proposal the user trying to perform the write operation has to create a signed request (One RSA_{sign}) including her credentials (PKC) and send it to the node responsible for the resource. The responsible node has to check that the user's credentials are valid (One RSA_{verify}), that she appears in its ACL as authorized writer and that the signature over the request is correct (One RSA_{verify}). Then it creates a signed answer (One RSA_{sign}) including its credentials (PKC) and sent it to the requester user. Finally, the user checks that the node's credentials are valid (One RSA_{verify}) and that the signature over the answer is correct (One RSA_{verify}). So the cost is $2RSA_{sign} + 6RSA_{verify}$ in our proposal and $2RSA_{sign} + 4RSA_{verify}$ in the identity-based approach.

- *Read Access Privileges:* Cost of performing a read access operation.

  1. Communicational Cost: In both proposals, when a user wants to perform a read operation over a resource she has to establish a communication with the node responsible for it. So, the cost in both proposals is One $Comm$.

  2. Computational Cost: In our proposal the user trying to read the resource has to create a signed request (One RSA_{sign}) including her credentials (PKC + AC granting her read privileges over the resource) and send it to the node responsible for the resource. The responsible node has to check that the user's credentials are valid (user's PKC has been signed by an administrative entity and an AC has been signed either by the system AA or by the owner of the resource that delegates access, Two RSA_{verify}) and that the signature over the read request is correct (One RSA_{verify}). Then, it sends the signed resource (this does not cost any operation to the responsible since the resource has already been signed in its last write operation) along with credentials (PKC + AC) of the last writer. Finally, the user checks that the credentials of the last writer are valid (writer's PKC has been signed by an administrative entity and an AC has been signed either by the system AA or by the owner of the resource that delegates access, Two RSA_{verify}) and that the signature over the resource is correct

Table 5.6: *Performance Analysis of the Use of Privileges.*

		USE PERFORMANCE	
		OUR	IDENTITY
Access System	COMM	*One*	*One*
	COMPT	$2RSA_{sign} + 6RSA_{verify}$	$2RSA_{sign} + 4RSA_{verify}$
Services System	COMM	*One*	*One*
	COMPT	$2RSA_{sign} + 6RSA_{verify}$	$2RSA_{sign} + 4RSA_{verify}$
Services Others	COMM	*One*	*One*
	COMPT	$2RSA_{sign} + 6RSA_{verify}$	$2RSA_{sign} + 4RSA_{verify}$
Write	COMM	*One*	*One*
	COMPT	$2RSA_{sign} + 6RSA_{verify}$	$2RSA_{sign} + 4RSA_{verify}$
Read	COMM	*One*	*One*
	COMPT	$RSA_{sign} + 6RSA_{verify}$	$RSA_{sign} + 4RSA_{verify}$

(One RSA_{verify}).

In the identity-based proposal the user trying to perform the read operation has to create a signed request (One RSA_{sign}) including her credentials (PKC) and send it to the node responsible for the resource. The responsible node has to check that the user's credentials are valid (One RSA_{verify}), that she appears in its ACL as authorized reader and that the signature over the request is correct (One RSA_{verify}). Then, it sends the signed resource (this does not cost any operation to the responsible since the resource has already been signed in its last write operation) along with credentials (PKC) of the last writer and the ACL of the resource. Finally, the user checks that the credentials of the last writer are valid (writer's PKC has been signed by an administrative entity and, in case last writer is not the owner, ACL has been signed by the owner of the resource and shows last writer as authorized writer, One or Two RSA_{verify}) and that the signature over the resource is correct (One RSA_{verify}). So the cost is $RSA_{sign} + 6RSA_{verify}$ in our proposal and $RSA_{sign} + 4RSA_{verify}$ or $RSA_{sign} + 5RSA_{verify}$ (when the last writer is not the owner of the resource) in the identity-based approach.

Table 5.6 summarizes this analysis related to the performance cost of the revocation of privileges.

Summary

Several conclusions can be drawn from this performance analysis:

- Apart from the assignment of the privilege to access the system, where our proposal needs the creation of a PKC and an AC in comparison with the identity-based approaches where only a PKC has to be created, our proposal highly reduces the cost of the assignment of privileges in P2PSIP networks. This is for two reasons: The first reason is that the assignment of a privilege in our proposal only causes the creation of the AC that contains it, as opposed to have to create a new PKC with the user's identity and all her privileges and revoke the old one in the identity-based approach. The second is because write/read access privileges in our proposal are sent directly to the user in an AC, while in the identity-based approach they have to be granted by modifying the ACLs of the responsible node for the resource and all its replicas. An extra advantage of our proposal is that it can delegate the management of identity PKCs in a trusted CA that carries out all the required work.

- In relation to the revocation of privileges, the traditional client-server revocation scheme is not feasible for P2PSIP systems since a CRL server must exist that should be contacted every time a user want to check a PKC or AC. Our proposal presents a fully distributed scheme that introduces a slight cost in the system but eliminates the necessity of maintaining and contacting a server. Also, the fact that each CRL is related to a unique resource make the amount of data exchanged inappreciable in comparison with traditional CRLs. For the rest of privileges, the cost is similar in both proposals with a slight overhead in our proposal due the fact that two certificates (PKC + AC) should be checked in comparison to one (PKC) to prove a user privilege. However, this overhead is really small in comparison with the flexibility advantages the use of AC presents. Also, it could be cut out using an hybrid proposal.

- Finally, the cost related to the use of a privilege is similar in both proposals with the already commented slight overhead in our proposal due the necessity of checking of two certificates to prove the user's privileges. Again, this overhead could be cut out using an hybrid proposal.

5.3.3 Infrastructure

Our proposal adds an Attribute Authority to the infrastructure needed for the identity-based approaches. AA's technical operation does not differ from the operations to be performed by a CA designed for identity-based approaches, and therefore the requirements between the implementation of a identity-based approach with a Certification Authority and now with an

Attribute Authority are similar. Also, as with the identity-based approaches, the implementation of a complete PKI it is not required, with our proposal the implementation of a PMI (Privilege Management Infrastructure) is not required either. Furthermore, although the CA and the AA are two different entities (and should for security reasons usually be located on different systems and use different asymmetric key-pairs for certification purposes), they may be located physically on the same system. Also, it is possible to develop our certification model without the implementation of a CA when an external source of authentication is used. Similarly, it is also possible to develop it without the implementation of an AA using an external source of authorization. Finally, our proposal eliminates the need of a CRL server when revocation of certificates is needed.

5.3.4 Standardization

All the mechanism used in our proposal are standardized: PKC, AC and CRL. This is an advantage in comparison with identity-based approaches that use non-standardized and application specific ACLs that could present several issues, like interoperability problems, security threats or extra performance overhead due to an unclear definition.

5.3.5 Security

From the security point of view, our certification model maintains the security of the identity-based approaches related to the authentication of users due the fact that all the users of the system hold a PKC. Also, using a centralized entity (AA) for the assignment of the necessary credentials (AC including node-ID and username) to access the system grants the proposal's resistance against *Sybil Attacks* [70] and ID Mapping Attacks [62]. Besides, our proposal offers the possibility of taking advantage of external trusted sources of authentication, such us Electronic Identity Cards.

As commented before, during the joining process it may exist the possibility of a user contacting a malicious node (whose credentials have been revoked) and that connects the user to a fake network. However, this is also possible even if the malicious node credentials have not been revoked in both our proposal and the identity-based approach. Possible ways to avoid this issue are contacting, in order to join the network, trusted nodes remembered from past sessions (peer-caches) or suggested by the system administrative entity when it is contacted for the first time in order to get the necessary credentials to access the system. Another possibility is to start the bootstrapping process with more than one node member of the network and cross-check results.

In relation to the security of the routing mechanism of the P2P system, it security must be provided by the specific P2P routing algorithm used for the application. However, the fact that

all the users of the network hold a PKC can be used to grant the authentication, integrity and confidentiality of the communications using the PK/PrK included in the certificates. Protocols like TLS/DTLS or IPSec can be used to secure these communications.

In relation to the user's access over the system resources, to use AC signed by the system administrative entity or by the owner of the resource grants the authenticity of the user's credentials. Besides, encryption can be used to avoid a malicious node responsible for a resource to reveal its content to unauthorized users. Also, the signature over the resources grants its integrity and the use of replication (provided by the specific P2PSIP application) prevent a single malicious node to deny the access over the resources it is responsible for. Finally, the fact that in our proposal write/read access privileges are provided using a different ACs for each privilege that are private to users and should only be presented to the node responsible for the resources in contrast to the identity-based approaches that must maintain public ACLs (that reveal to every node in the system which users have privileges over a resource) protects the privacy of the services subscribed by a user, improving the security (privacy) of the system.

5.4 Conclusions and Contributions

In this chapter we have presented a new authorization scheme for P2PSIP systems based on a clear differentiation between the concepts of authentication and authorization. This differentiation is built on the use of attribute certificates that link the privileges of a user within the system with her identity represented by a public key certificate. Also, we present a distributed revocation system that can be established within the P2PSIP system and does not need the intervention of any external server.

Our proposal solves the limitations of identity-based approaches, allowing the establishment of different durations for the user's privileges, the use of external PKCs as authentication authorities and the definition of a finer-grain access control system over the system resources. The evaluation conducted of both proposals shows that our scheme is not only more flexible than the identity-based approaches but also more secure and efficient for the assignment of privileges while preserving its simple infrastructure and, even reducing it, when revocation of certificates is needed.

From the results of the research presented in this chapter derives the paper *On Authorization for P2P networks* that is under review in the journal *Computer Communications*.

CHAPTER 6

SECURITY RECOMMENDATIONS FOR P2PSIP SYSTEMS

> Every rule can be broken, but no rule may be ignored.

During the previous chapters of this thesis we have analyzed the security of P2PSIP communication systems [278] and presented several measures to improve the security of their access control service. Following, and based on this analysis, we present a discussion of the main security recommendations (Table 6.1) for the development of P2PSIP systems. These recommendations, as our initial analysis of the security of P2PSIP communication systems was, are presented independently for each of the services forming a P2PSIP system: access control, bootstrapping, routing, storage and communications. Also, an extra set of security recommendations for the security issues that affect the whole system is presented.

6.1 Access Control

As we have seen through this thesis, a secure access control and node-ID assignment are the key of the security of P2PSIP systems since the effectiveness of most of the defenses against the presented attacks depend on the ability of the system to assign a unique identity to each user and her device and, therefore, limit the number of users that are malicious.

In base of these needs, we have analyzed the existing access control schemes for P2PSIP systems and presented several new secure solutions to improve this service. These solutions were presented independently to facilitate understanding their motivation and advantages. However, in order to built a secure access control service they should be combined. Therefore, in the following paragraphs we present our recommendation for the access control of P2PSIP systems based on the previous presented specific secure solutions.

Table 6.1: *P2PSIP Security Recommendations.*

SERVICE	RECOMMENDATIONS
Access Control	*Centralized (offline CA) with split certification and authorization based on AC* for the vast majority of scenarios. *Proposal for on-the-fly systems based on AC* for decentralized scenarios.
Bootstrapping	*Peer-caches*, *Random Address Probing* and *Network Layer Mechanism*. Also, when possible, *Centralized Bootstrapping* and/or *Out of Band Mechanisms*.
Routing	Routing mechanisms' efficiency heavily depends on the scenario. For general cases: *Symmetric Recursive* and *Alternate Routing Path* or *Parallel Routing*. *Mixed Routing Tables* and *Cross Checking Routing Tables*. Finger-to-Finger (*TLS/DTLS* or *IPsec*) security. End-to-end (*Encryption*, *Digital Signature* and *Message Identifiers*) security. *Balancing Techniques* and *Routing Variations* (*External Anonymous Systems* are also possible).
Storage	*Digital Signatures* for resources' integrity and authenticity. *Replication* (small files) and *Erasure Codes* (large files) for resources' availability. Resources' access controlled by *Attribute Certificates* and *Cryptography*.
Communications	Simple and stateless applications. SIP sessions secured with *TLS/DTLS* or *IPSEC*. Media sessions secured with *SRTP* and, when available, a *Separate Link*. *Content Filtering*, *Buddy List (White List, Reputation Systems, Consent-based, Turing Tests)*, *Legal Action* and *Address Obfuscation* for SIP Spam.
Miscellaneous	*Adaptative Recovery*, *Reciprocity Systems* and *Ejecting Misbehaving Nodes*.

Due to the famous *Sybil Attack*, a centralized access control (based on an offline CA) is recommended for the vast majority of scenarios. With this in mind, we propose using the split certification model presented in Chapter 3 [279] in combination with the authorization model based on attribute certificates presented in Chapter 5.

As we can see in Figure 6.1, devices and users are identified by different PKCs, issued by a TTP that uniquely authenticate them. Devices' certificates can be issued by their manufacturer at production time (such as in the case of a mobile P2PSIP phone) or by the company or authority running the P2PSIP system (such as in the case of the device a telecommunication provider installs at the user office or home, or when a user requests to the administrative authority a PKC for the computer she intends to access the system from). In the same way, users' certificates can be issued by external trusted organizations (such as in the case of a user holding an electronic identity card issued by her country's government) or by the P2PSIP system administration authority.

Based in this authentication through PKCs, users and devices can obtain different privileges

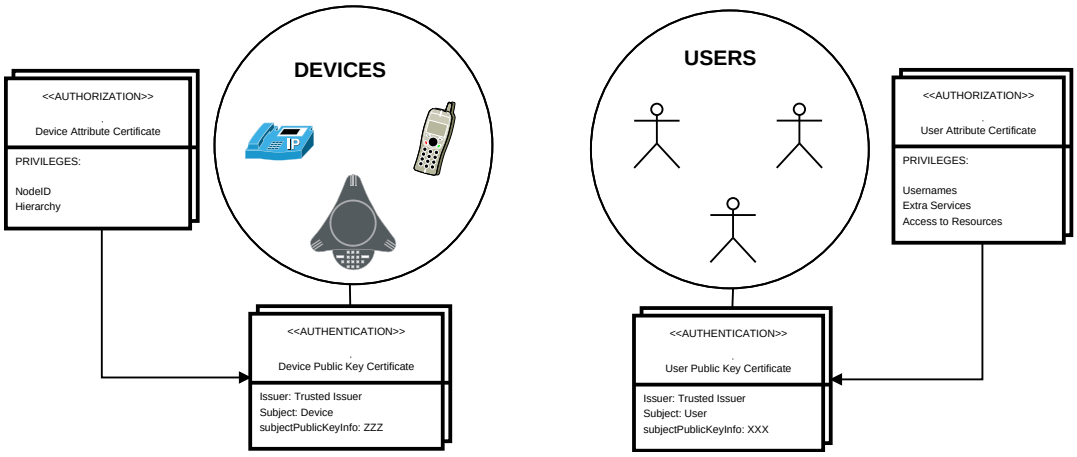


Figure 6.1: *Recommended Access Control Model for P2PSIP Systems.*

included in ACs to access the resources of the P2PSIP system. Nodes' access to the system is granted through ACs issued by the system administration that include a unique and random node-ID. These node-IDs also define their location in the network and the resources they are responsible for. Alike, extra ACs can be issued to define new privileges or roles of the nodes in the network (such as authorizing them to act as a gateways between two systems, defining their temporal role as super-peer in a hierarchical P2P network or as providers of STUN -Session Traversal Utilities for NAT- server capabilities).

Users' privileges are also defined trough ACs. Their access to the system is granted by an AC issued by the system administration including a unique SIP username that also defines where the users' contact information and resources are stored in the network. Extra authorization credentials to access to other services of the network, such as extra storage quota or authorization to establish communications to a external system (like a PSTN network) through a gateway, can be defined by different ACs issued by the system administration. Similarly, other companies providing extra services within the system can authorize users' access to these services by issuing ACs to them. Finally, users can also grant write or read access to their own resources by issuing ACs to the other users of the system, as described with more detail in Chapter 5.

This access control scheme requires the inclusion in the system of an offline certification authority. However, there exist scenarios, such as on-the-fly creation P2PSIP systems, where making the access control of the system dependent on such entity is not a viable option. For such cases, we recommend the adaptation of the proposal for on-the-fly creation P2PSIP systems presented in Chapter 4 (where the creator of the system substitutes the role of the offline certification authority) to the described access control scheme. In order to do so, the system's creator identified by a PKC (either obtained before the creation of the network from a TTP or self-signed) issues to herself an AC authorizing her as the system's administrator. Alike,

she issues to herself an AC including a username that authorizes her access to the system and another AC including a node-ID that authorizes her device for becoming a node of the network. In case her device does not have already a PKC that authenticate it, the creator should also issue a PKC for it. Once the system has been created and the system's creator has accredited himself as an administrative entity, other potential users of the system can contact her in order to get the credentials needed to join the system. These credential are an AC including a username for the joining user and an AC including a node-ID for the joining user's device. Also, if the joining user and her device do not already have a trusted PKC the creator of the system should issue two PKC for them (one for the user and another one for her device). Finally, as deeply explained in Chapter 4, the system's creator can issue ACs to other trusted users accrediting them as system's administrators to distribute the system's administrative functions among several entities.

6.2 Bootstrapping

In relation to the bootstrapping service, none of the existing techniques seems to give full warranties: peer-caches work well with short periods of disconnection, but should be combined with another methods in order to bootstrap after long periods or for the first time; random address probing need a large number of users to be connected to the system in order to be suitable and do not work well with NAT, network layer mechanisms depend on the information stored at the multicast or anycast routers and at the central SLP directory services, out of band mechanisms require accessing external systems that may not be accessible in scenarios with lack of connectivity, also their information can be outdated; centralized bootstrapping has serious scalability problems, etc.

With these problems in mind, the best option is to use a combination of the presented alternatives. For the access control scheme based on a centralized certification authority, we recommend to use the approach presented in RELOAD where users obtain the multicast group of bootstrap peers and the IP address of some of them at certification time. Since the nodes of the network may vary with time, we also recommend to combine this approach with peer-caches remembered from past sessions. Random address probing should be used in case neither of the approaches present before success. Finally, contacting an out of band mechanism such as a HTTP-based cache should be also considered as an option when possible. Figure 6.2 summarizes these recommendations.

When the inclusion of a centralized certification authority in the system is not possible, administrative users (following the access control scheme for on-the-fly P2PSIP systems presented in the previous section) should announce their presence using a well known network layer mechanism. Once the joining user has contacted one of the administrators to get the necessary credentials to access the system, the administrator should provide the joining user with a mul-

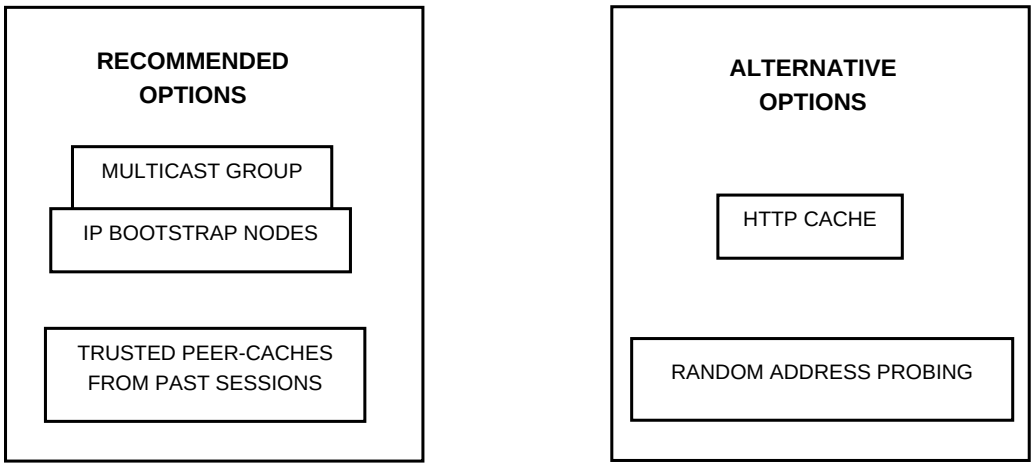


Figure 6.2: *Recommended Bootstrapping Methods for P2PSIP Systems.*

ticast group formed by the administrative users or a list of their nodes' IPs in order to help her joining the system in the future. Also, peer-caches and random address probing should be used when needed as described for the centralized version.

As a final recommendation and to prevent joining users from suffering a *Fake Bootstrapping Attack*, it is highly advisable to check that the credentials of the admitting nodes are valid. Besides, starting the bootstrapping process with several nodes and cross-check the results would help the user to be sure that she has actually join the correct system in a secure way.

6.3 Routing

The efficiency and correctness of message routing is of great importance for the behavior of P2PSIP systems. Several attacks have been presented during this thesis that can disrupt its functioning. Also, several defenses against these attacks have been described. However, no one of this defenses is definitive, and the characteristics of the scenario where the system is going to be deployed must be taken into account to decide what kind of routing should be selected. Besides, routing is not a monolithic service but is formed by different modules with several possible configurations that should be chosen in base on the system's requirements. Among these, the most important decisions for the routing configuration are to chose its forwarding technique, its routing technique, the routing tables used and how the routing messages are secured.

Two forwarding techniques have been studied: recursive and iterative routing; both having their advantages and drawbacks, as discussed in Chapter 2. Recursive routing performs better with NAT networks due the fact that no extra connections need to be created in order to route a message. Also, it is the unique viable alternative under unstable conditions [96]. However, it suffers from a possible amplification of DoS attacks and the little control the initiating user

has over the routing process. For its part, iterative routing consumes fewer resources for the intermediate peers (they only have to send redirect messages rather than forwarding requests and responses) and gives more control to the source user over the routing. However, it has a low global performance, mainly when nodes behind NAT are involved. With the commented advantages and drawbacks in mind, we recommend the use of (symmetric) recursive routing except under special conditions that explicitly do not advise to do so.

In relation to the routing technique to be used, several alternatives have been presented during this thesis. Choosing a secure routing mechanism, such as parallel routing, raises the habitual dilemma, security vs. performance, that should be analyzed specifically for every scenario where the system is going to be developed. Alternate routing path detection algorithm is not very accurate and can induce to unnecessary overhead in the system's routing. The same problem has parallel routing that induces to even more overhead in the system. For its part, social trust routing is inefficient and depends on an external social network. Therefore, as a general solution for non very restrictive (in terms of bandwidth, computational resources or battery) scenarios, we recommend launching an alternate routing path or parallel routing when the first routing option fail. Besides, in case iterative routing is used, it should be complemented with hop testing or trust diversity routing.

Several alternatives also exist for the routing tables. Since periodic routing tables are not a viable option, despite of its security; using mixed routing tables to combine the efficiency of flexible routing tables with the security of constrained routing tables seems to be the best option. Also, when bandwidth permit it, this technique can be combined with periodically cross-checking the routing tables of other randomly chosen nodes.

In order to secure the routing messages several methods can be combined as shown in Figure 6.3. Routing's messages should be secured node-by-node establishing TLS/DTLS or IPSEC tunnels between each node and the fingers in its routing table. Besides, user's messages should be secured end-to-end using digital signatures (to grant its authenticity and integrity) and asymmetric encryption (to grant their confidentiality). In order to grant user's messages freshness, message identifiers (such as nonces) should be included in the end-to-end part of the messages. To conclude with routing, we recommend to use balancing techniques to distribute the load of the network and implementing routing variations when anonymity is desirable. The inclusion of an external anonymous systems only seems to be an option when anonymity is really critical, due to their resource consumption and latency.

6.4 Storage

Resource management in P2PSIP system is a more challenging task than in traditional client-server systems. The fact that users' resources are distributed among the nodes of the network,

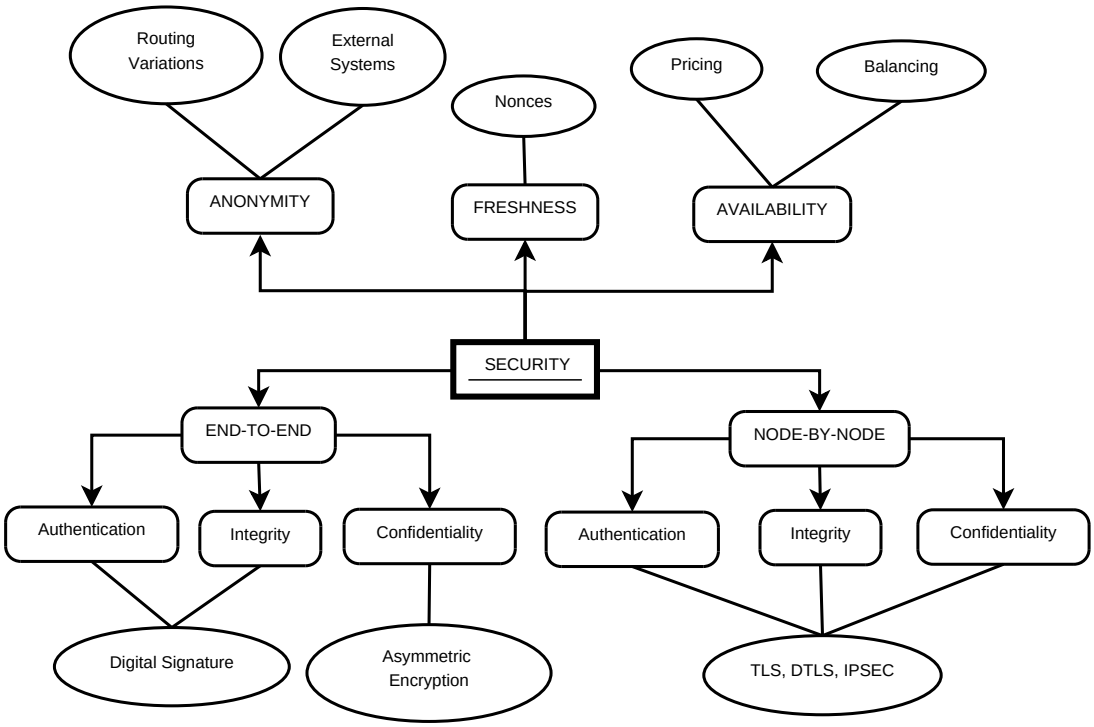


Figure 6.3: *Security of Routing Messages.*

that may be not trustworthy, and not stored in centralized trusted servers suggest the use of specific solutions for this kind of systems. As we have already presented in Chapter 5, existing resource access control models for P2PSIP systems are based on the adaptation of ACLs traditionally used in operating systems and client-server architectures. However, these models have serious privacy, interoperability and flexibility problems. Therefore, to manage the access control over the resources of the system we recommend using the model based on attribute certificates presented in Chapter 5.

Besides, in case of private resources, since the node responsible for them may be malicious and give free access to them to other non authorized nodes of the network, attribute certificates based access control should be complemented with cryptography: user's private resources should be symmetrically encrypted, private resources intended for a specific user should be asymmetrically encrypted with the user's PK, and private resources intended for a group of users should be symmetrically encrypted with a key shared by all the members of the group.

Another problem of resource management in P2PSIP systems due to is decentralized architecture is granting the integrity and authenticity of the data stored. However, due the fact that all the users of the system hold a certificate, this issue can be easily solved by performing a digital signature over each piece of data stored in the system. This way, any user trying to access a resource can check its integrity and authenticity by validating its signature.

To conclude with the recommendations related to the storage service, we look at the existing

solutions to provide data availability. Replication provides a great resource availability but may unnecessary increase the amount of data stored in the system. In turn, erasure codes reduces the storage overhead of replication but increases the computational one. With this in mind, our recommendation is to implement replication for small and highly accessed files, such as users' contact info, and erasure codes for large and non frequently accessed files, such as voicemails.

6.5 Communications

As we have seen, the communication service of P2PSIP systems can be affected by several attacks. In order to prevent them, several security solutions should be combined in its design. The first recommendation is to keep the user's application as simple and stateless as possible. Simplicity facilitates the application design and reduces the probability of errors or vulnerabilities in its code. Also, it eases the system analysis. However, simplicity should not compete with a good implementation to prevent malformed messages rendering the application inoperable. Within this concern, extra security modules and message size limits can be also used.

For its part, an stateless application improves the system performance. Besides, an stateless service in conjunction with the limitation of the number of conversations accepted from a single source, making use of the strong user's authentication presented in the access control section of these recommendations, reduces the probability of suffering DoS attacks.

The next step to be taken in order to secure the communication service, it is related to the SIP messages exchanged between two or more users that want to start a dialog. In order to do so, these SIP messages should be secured using well-known protocols such as TLS/DTLS or IPSEC to grant their confidentiality, integrity and authentication (based on the secure assignment of certificates described in the access control section). In the same way, signaling messages exchanged during the dialog should be also secured, using the same methods, to prevent a malicious user from altering or tearing down the session. Finally, the media conversation should be protected using SRTP (due to its better performance securing real-time data in comparison with IPSec) and, when possible, transported using a different network link.

To end with the recommendations for the communication layer, we present the measures that should be adopted to prevent SIP spam in all of its versions (SPIT, SPIM and SPPP). User's communications in a P2PSIP are usually based on a buddy list that contains user's contacts. Our recommendation is to use a buddy list based on a white list that defines which users of the system can contact with and know the state (connected, disconnected, etc.) of the user. In order to add a new contact to a user's buddy list, reputation systems and consent-based queries should be used. Also, buddies invitations should include Turing tests to avoid bot interventions. Finally, extra measures such as content filtering to block typical SPIM and SPPP messages, legal intervention by governments and address obfuscation could be used.

6.6 Miscellaneous

Until now, we have presented security recommendations for the development of the specific services forming a P2PSIP system. However, as we have seen during the course of this thesis, there exist security issues that may affect not only a single service but the system as a whole. Concerning this, the already commented measures to secure access control and routing should be deployed in order to prevent *Eclipse attacks*. Also, the use of degree observation may help to reduce the impact of these kind of attacks.

Another issue that could highly reduce the performance of the system is free-riding. Among the existing alternatives to prevent free-riders, using reciprocity (mainly reputation systems) seems to be the more extended and appropriate approach to stimulate the collaboration of free-riders. Also, when possible, this technique could be combined with the identification and the ejection of misbehaving nodes in order to improve the global performance of the system.

To conclude, an efficient recovery system should be developed to grant the self-organization and maintenance of the network. Reactive recovery responds well to the system's changes, but for bigger networks and high churn rates can lead to network collapse. Periodic recovery, however, reduces the impact of the maintenance process in the system performance but may not react well to different network conditions since the periods of maintenance are prefixed. Therefore, an adaptative approach, that takes into consideration the continuous evolution of the network, seems to be the more efficient and appropriate option.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

> I see my path, but I don't know where it leads. Not knowing where I'm going is what inspires me to travel it.
>
> Rosalía de Castro

7.1 Conclusions

During the course of this thesis we have studied the completely new problematic P2PSIP communication systems present. Based on the existing researches related to the traditional SIP systems and P2P networks, we have identified the most important security services within the architecture of P2PSIP communication systems. Once identified the different services, we have analyzed their security in base of the attacks that can affect each one of them, as well as the existing countermeasures that can be used to prevent or mitigate these attacks and their viability. From this security analysis follows that the access control service is the cornerstone of the security of P2PSIP communications systems and it should be enhanced.

To this end, we have designed new solutions related to the certification of users, the access control for on-the-fly systems and the user's authorization service. The evaluation conducted in terms of flexibility, security, infrastructure and performance has reflected the improvements presented by our solutions in comparison to existing alternatives.

Finally, with base on the security analysis, the designed solutions and their evaluation, we have defined a set of security recommendations that should be considered for the design, implementation and maintenance of P2PSIP communication systems.

Following, we describe with more detail the contributions and conclusions of the research conducted during the course of this thesis.

7.1.1 Security Analysis of P2PSIP Communications Systems

The new architecture of P2PSIP presents a completely different problematic in comparison with traditional SIP systems and file-sharing P2P networks. Nevertheless, most of the solutions used for P2PSIP systems are based in previous analysis done for this kind of networks. It was, therefore, necessary to carry out a specific analysis of the security of this new architecture as we have presented in Chapter 2.

After an overview of the services that conform P2PSIP systems' architecture (access control, bootstrap, routing, storage and communication) we have presented the different attacks that can be launched against each of the services described. Also, for each presented attack, we have reviewed the defense mechanisms that can be used to prevent it, summarizing their advantages and drawbacks.

This analysis presents a clear picture of the new security challenges which must be considered for the development of a Peer-to-Peer Session Initialization Protocol system and a revision of the security mechanisms that can be used to secure them; stating as a good starting point, before inexistent, for any research related to the security of P2PSIP communication systems.

7.1.2 Split Certification Model

The certification model used by actual P2PSIP systems permanently links a username with a nodeID. In this way, both a user and her device are identified by the same PKC and the same PK is used to secure their communications. Nevertheless, devices and users are different entities that carry out different roles within the system and therefore the identity of a user (represented by her username) and the devices she is using (represented by its node-IDs) should be separated. Also, communications performed by a device acting as a node member of the network (like routing messages, retrieving or storing of resources, etc.) should be separated from the communications performed by a user (making a call, updating his contact information in the network, etc.). It is unnecessary (and a security flaw) for a user contacting a node of the network just to route a message on her behalf to have the knowledge of the user using the contacted node. Likewise, there is no need for the contacted node to know which node the caller performing the request is operating from.

Following these observations, we have presented an alternative of certification based on the separation of certificates for devices and users that raises the security of the communications with a two-layer security, provides a improved anonymity to users and allows the establishment of a more secure network by using trusted devices with hard-coded certificates. Also, it adds extra features, such as letting several users be connected to the same device or allowing a user holding a single PKC to be connected to several devices.

7.1.3 New Access Control for on-the-fly P2PSIP Systems

Several alternative schemes have been presented in the literature to try to solve the access control problem in P2P systems when it is not possible to include a logically centralized authority (either online or offline) in the system. However, most of them are not suitable for on-the-fly P2PSIP systems and the most typical ones (IP based, shared secret and threshold cryptography) have several security and performance drawbacks.

From the deficiencies of the existing schemes, we have presented a new access control scheme for on-the-fly P2PSIP systems which is based on the recently published Internet Attribute Certificate Profile for Authorization [261]. In our proposal, the creator of the network initially acts as a CA issuing certificates for each new user of the network and, as the size of the systems grows, uses ACs to distribute the access control of the system among several trusted users.

Our proposal greatly improves the security and flexibility of IP based and shared secret schemes with no infrastructure cost and with a minimal performance charge. Also, it achieves a similar level of security than threshold cryptography while highly reducing its computational and communicational cost. All this facts position our proposal as an alternative to access control for on-the-fly P2P systems in non very hostile environments where performance and security are key factors.

7.1.4 New Authorization Model for P2PSIP Systems

Existing access control schemes for P2PSIP systems are based on a single PKC that represent two roles: user's authentication (who the user is) and user's authorization (privileges of the user in the network). However, the fact that PKCs are both used for authentication and authorization of users is not a good idea. Including the identity and the privileges of a user (username, nodeID, services contracted, etc.) in the same certificate determines that both the identity and any privileges should have the same lifetime and should be issued by the same authority. Also, every time a new privilege is added, removed or changed the certificate should be revoked and a new one should be created. This authorization approach is inefficient and does not consider scenarios where the identity of the users is granted by an external trusted certification authority (CA).

With these deficiencies of the traditional identity-based authorization models in mind, we have presented a complete framework for authorization on P2PSIP systems based on the recently published Attribute Certificate Profile for Authorization [261] that links the privileges of a user within the system with her identity represented by a PKC. Also, we present a distributed revocation system that can be established within the P2PSIP network and does not need the intervention of any external server. The evaluations conducted show that this separation between authentication and authorization outlines a more flexible and secure authorization scheme while

improving the efficiency of the assignment of privileges. Examples of scenarios of application of our proposal are: situations where users' privileges may have different lifetime, the use of external CAs as source of authentication or a system where different entities (companies, users, etc.) want to provide services.

7.1.5 Security Recommendations P2PSIP Systems

Based on the security analysis conducted and the new solutions developed to enhance the access control service of P2PSIP systems, we have presented a set of security recommendations that should be considered for the design, implementation and maintenance of P2PSIP communication systems. These recommendations are not unique, but vary depending on the scenario and its security requirements.

Several defenses exist against each one of the presented attacks, however, the choice of the ones to be used and their implementation is a complicated task. Most of the solutions presented so far to secure P2PSIP systems are adaptations of security mechanisms developed for P2P file-sharing systems or traditional SIP systems. And, despite its effectiveness on those environments, it is still early to affirm that they are the most appropriate for P2PSIP systems. Also, each security measure has drawbacks: central servers limit the decentralized nature of the network, cryptographic protocols need extra computational resources, secure routing and maintenance mechanisms increase the load of the network, etc. These drawbacks limit the network capabilities, and, in some scenarios, it may not be possible to implement them. It is, therefore, of great importance to conduct a thorough analysis of the environment where the system is going to be developed in order to find the most appropriate measures to secure it.

7.2 Future Work

As we have already commented during the course of this thesis, a deep analysis of all the services involved in a P2PSIP systems and the design of specific security solutions for all of them is a topic too broad to be addressed in a single thesis. Therefore, the research already conducted opens various ways of future work, on the one hand related to the designed solutions for the access control service and in the other hand related to the other services forming a P2PSIP system, that we detail in the following paragraphs.

7.2.1 IETF Draft

Our research is clearly related to RELOAD, the proposed standard for P2PSIP communication systems currently being developed by the IETF. With this in mind, we intend to submit our improvements over its access control service in a draft document for the consideration of the

IETF P2PSIP working group. The draft would propose the substitution of the RELOAD's certification model by our split certification model, the inclusion of our authorization model based in attribute certificates at the sacrifice of the actual identity-based approach and the replacement of the shared secret scheme by our proposal for on-the-fly systems.

7.2.2 Specific Research in other Services

We have limited the design of new secure solutions for P2PSIP communication systems to the access control service. However, the security analysis conducted at the beginning of this thesis has shown possible areas of research in relation to the other services that we will try to address in the near future:

- *Bootstrapping in P2PSIP systems:* In relation to the bootstrapping service, we think in the possibility of testing the security and efficiency of the existing bootstrapping methods under different adversarial models and network conditions, and designing the necessary solutions in case the results not be satisfactory.

- *Routing improvements for P2PSIP systems:* For the routing service, it necessary a deeper analysis of the different routing algorithms under adversarial conditions to test their capabilities of achieving successful searches in a reasonable time for the P2PSIP system to be usable.

- *Efficient Storage for P2PSIP:* For the storage scheme, we are thinking in the design of new alternatives to reduce the overhead in the network of large files like voicemails. Also, the analysis of the different cryptographic alternatives that can be used to secure the resources of the network and the distribution of the keys needed to access them appears as an interesting area of research.

- *Media Communications through the P2PSIP overlay network:* For the communication layer, a possible are of research is studying the feasibility of using the underlaying P2P network not only for the location of users but also to forward the media communications when a direct connection is not possible between the participants.

BIBLIOGRAPHY

[1] J. Postel. Internet Protocol. RFC 791 (Standard), September 1981. Updated by RFC 1349.

[2] D. Cohen. Specifications for the Network Voice Protocol (NVP). RFC 741, November 1977.

[3] ITU-T. Recommendation H.323. Technical report, ITU, 2006.

[4] ITU-T. Recommendation H.225.0: Call signalling protocols and media stream packetization for packet-based multimedia communication systems. Technical report, ITU, 2006.

[5] ITU-T. Recommendation H.245: Control protocol for multimedia communication. Technical report, ITU, 2008.

[6] M. Spencer, B. Capouch, Ed. E. Guy, F. Miller, and K. Shumard. IAX: Inter-Asterisk eXchange Version 2. RFC 5456 (Informational), February 2010.

[7] Mark Spencer. Asterisk: The Open Source PBX & Telephony Plataform Website. http://www.asterisk.org/, 1999.

[8] J. Postel. Transmission Control Protocol. RFC 793 (Standard), September 1981. Updated by RFCs 1122, 3168.

[9] J. Postel. User Datagram Protocol. RFC 768 (Standard), August 1980.

[10] R. Stewart. Stream Control Transmission Protocol. RFC 4960 (Proposed Standard), September 2007.

[11] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. RFC 3261 (Proposed Standard), June 2002. Updated by RFCs 3265, 3853, 4320, 4916, 5393, 5621, 5626, 5630.

[12] A. B. Roach. Session Initiation Protocol (SIP)-Specific Event Notification. RFC 3265 (Proposed Standard), June 2002. Updated by RFC 5367.

[13] J. Peterson. S/MIME Advanced Encryption Standard (AES) Requirement for the Session Initiation Protocol (SIP). RFC 3853 (Proposed Standard), July 2004.

[14] Information Technology Laboratory, NIST, Gaithersburg, USA. *FIPS 46-3. Data Encryption Standard*, 1998.

[15] Information Technology Laboratory, NIST, Gaithersburg, USA. *FIPS 197. Advanced Encryption Standard (AES)*, 2001.

[16] R. Sparks. Actions Addressing Identified Issues with the Session Initiation Protocol's (SIP) Non-INVITE Transaction. RFC 4320 (Proposed Standard), January 2006.

[17] J. Elwell. Connected Identity in the Session Initiation Protocol (SIP). RFC 4916 (Proposed Standard), June 2007.

[18] SIP Working Group Status Page. http://tools.ietf.org/wg/sip/, 2005.

[19] David A. Bryan, Bruce B. Lowekamp, and Cullen Jennings. SOSIMPLE: A Serverless, Standards-based, P2P SIP Communication System. In *Proceedings of the First International Workshop on Advanced Architectures and Algorithms for Internet Delivery and Applications*, pages 42–49, Washington, DC, USA, 2005. IEEE Computer Society.

[20] The seti@home project website. http://setiathome.berkeley.edu, 1997.

[21] W. Sullivan III, D. Werthimer, S. Bowyer, J.Cobb, D. Geyde, and D. Anderson. A new major seti project based on project serendip data and 100,000 personal computers. In *Proceedings of the 5th International Conference on Bioastronomy*, 1997.

[22] V. Pande, C. Snow, and S. Larson. Folding@home and gnome@home: Using distributed computing to tackle previously intractable problems in computational biology. *Modern Methods in Computational Biology, Horizon Press*, 2003.

[23] The genome@home Project Website. http://genomeathome.stanford.edu, 2000.

[24] R. HueBsch, J. Hellerstein, N. Lanham, and B. Thau Loo. Querying the internet with pier. In *Proceedings of the 29th VLDB Conference*, 2003.

[25] A.Halevy, Z. Ives, P.Mork, and I. Tatarinov. Piazza: Data management infraestructure for semantic web applications. In *Proceedings of the 12th International Conference on World Wide Web*, pages 556–567, 2003.

[26] The Napster Website. http://free.napster.com/, 2003.

[27] Gnutella A Protocol for a Revolution. http://rfc-gnutella.sourceforge.net/index.html, 2000.

[28] Ian Clarke, Oskar Sandberg, Brandon Wiley, and Theodore W. Hong. Freenet: A Distributed Anonymous Information Storage and Retrieval System. In *International workshop on Designing privacy enhancing technologies: design issues in anonymity and unobservability*, pages 46–66, New York, NY, USA, 2001. Springer-Verlag New York, Inc.

[29] D. Bryan, P. Matthews, E. Shim, D. Willis, and S. Dawkins. Internet-Draft: Concepts and Terminology for Peer to Peer SIP. draft-ietf-p2psip-concepts-03 (work in progress), October 2010.

[30] D. Chopra, H. Schulzrinne, E. Marocco, and E. Ivov. Peer-to-Peer Overlays for Real-Time Communication: Security Issues and Solutions. *Communications Surveys & Tutorials, IEEE*, 11(1):4–12, First Quarter 2009.

[31] H. Schulzrinne, E. Marocco, and E. Ivov. Security Issues and Solutions in Peer-to-Peer Systems for Realtime Communications. RFC 5765 (Informational), February 2010.

[32] Rick Lehtinen. *Computer Security Basics, 2nd edition*. O'Reilly, 2006.

[33] CERT/CC Computer Emergency Readiness Team Coordination Center. http://www.cert.org.

[34] J.P. Holbrook and J.K. Reynolds. Site Security Handbook. RFC 1244 (Informational), July 1991. Obsoleted by RFC 2196.

[35] B. Fraser. Site Security Handbook. RFC 2196 (Informational), September 1997.

[36] Special Publication 800-12. An Introduction to Computer Security: The NIST Handbook. Technical report, NIST, 1996.

[37] ITU-T. Recommendation X.800: Security Architecture for open Systems Inteconnection for CITT. Technical report, ITU, 1991.

[38] H.J. Schumacher and S. Ghosh. A fundamental framework for network security. *Journal of Network and Computer Applications*, 1997.

[39] R. Shirey. Internet Security Glossary, Version 2. RFC 4949 (Informational), August 2007.

[40] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. RFC 2616 (Draft Standard), June 1999. Updated by RFC 2817.

[41] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard), August 2008.

[42] S. Kent and K. Seo. Security Architecture for the Internet Protocol. RFC 4301 (Proposed Standard), December 2005.

[43] Joachim Posegga, NewAuthor2, and NewAuthor3. Voice Over IP: Unsafe at any Bandwidth? In *In Proc. EURESCOM Summit 2005: Ubiquitous Services and Applications*, 2005.

[44] J. Rosenberg and C. Jennings. The Session Initiation Protocol (SIP) and Spam. RFC 5039 (Informational), January 2008.

[45] R. Barbieri, D. Bruschi, and E. Rosti. Voice over IPsec: Analysis and Solutions. In *Proceedings of 18th Annual Computer Security Applications Conference*, 2002.

[46] S Salsano, L. Veltri, and D. Papalilo. SIP security issues: the SIP authentication procedure and its processing load. *IEEE Network*, 2002.

[47] Feng Cao and S Malik. Security analysis and solutions for deploying IP telephony in the critical infrastructure. In *Workshop of the 1st International Conference on Security and Privacy for Emerging Areas in Communication Networks*, 2005.

[48] D. Richard Kuhn, Thomas J. Walsh, and Steffen Fries. Special Publication 800-58. Security Considerations for Voice over IP Systems. Technical report, NIST, 2005.

[49] CCN-STIC-414 Seguridad en Voz sobre IP. Technical report, CCN, 2007.

[50] Stephanos Androutsellis-Theotokis and Diomidis Spinellis. A Survey of Peer-to-Peer Content Distribution Technologies. *ACM Comput. Surv.*, 36:335–371, December 2004.

[51] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '01, pages 149–160, New York, NY, USA, 2001. ACM.

[52] Petar Maymounkov and David Mazières. Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, IPTPS '02, pages 53–65, London, UK, 2002. Springer-Verlag.

[53] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. A Scalable Content-Addressable Network. In *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '01, pages 161–172, New York, NY, USA, 2001. ACM.

[54] Antony I. T. Rowstron and Peter Druschel. Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems. In *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg*, Middleware '01, pages 329–350, London, UK, 2001. Springer-Verlag.

[55] B. Zhao, J. Kubiatowicz, and A. Joseph. Tapestry: An Infrastructure for Fault-Tolerant Wide-Area Location and Routing. Technical Report UCB/CSD-01-1141, University of California, Berkeley: Computer Science, Berkeley, CA, USA, 2001.

[56] Eng Keong Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim. A survey and comparison of peer-to-peer overlay network schemes. *Communications Surveys Tutorials, IEEE*, 7(2):72–93, quarter 2005.

[57] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset, and H. Schulzrinne. Internet-Draft: REsource LOcation And Discovery (RELOAD) base protocol. draft-ietf-p2psip-base-13 (work in progress), March 2011.

[58] Steve Bellovin. Security aspects of Naspter and Gnutella. In *Usenix Annual Technical Conference*, 2001.

[59] Naspter Page. http://www.naspter.com, 2003.

[60] Emil Sit and Robert Morris. Security Considerations for Peer-to-Peer Distributed Hash Tables. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, IPTPS '01, pages 261–269, London, UK, 2002. Springer-Verlag.

[61] Mudhakar Srivatsa and Ling Liu. Vulnerabilities and Security Threats in Structured Overlay Networks: A Quantitative Analysis. In *Proceedings of the 20th Annual Computer Security Applications Conference*, ACSAC '04, pages 252–261, Washington, DC, USA, 2004. IEEE Computer Society.

[62] D. Cerri, A.Ghioni, S. Paraboschi, and S. Tiraboschi. ID Mapping Attacks in P2P Networks. In *IEEE Global Telecommunications Conference, GLOBECOM'05*, volume 3, December 2005.

[63] Christian Scheideler. How to Spread Adversarial Nodes?: Rotate! In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, STOC '05, pages 704–713, New York, NY, USA, 2005. ACM.

[64] M. Engle and J. I. Khan. Vulnerabilities of P2P Systems and a Critical look at Their Solutions. Technical report, Technical Report, Department of Computer Science, Kent State University, 2006.

[65] Miguel Castro, Peter Druschel, Ayalvadi Ganesh, Antony Rowstron, and Dan S. Wallach. Secure routing for structured peer-to-peer overlay networks. In *Proceedings of the 5th symposium on Operating systems design and implementation*, OSDI '02, pages 299–314, New York, NY, USA, 2002. ACM.

[66] Dan S. Wallach. A Survey of Peer-to-Peer Security Issues. In *Proceedings of the 2003 Mext-NSF-JSPS international conference on Software security: theories and systems*, ISSS'03, pages 42–57, Heidelberg, Germany, 2003. Springer-Verlag.

[67] K. Hildrum and J. Kubiatowicz. Asymptotically Efficient Approaches to Fault-Tolerance in Peer-to-Peer Networks. In *Proceedings of 17th International Symposium on Distributed Computing*, volume 2848 of *Lecture Notes in Computer Science*, pages 321–336, Heidelberg, Germany, October 2003. Springer-Verlag.

[68] L. Zhao and B. Y. Ganesh. Identity theft protection in structured overlays. In *Proceedings of the 1st Workshop on Secure Network Protocols*, 2005.

[69] G. Danezis, C. Lesniewski-Laas, M.F. Kaashoek, and R. Anderson. Sybil-Resistant DHT Routing. In *10th European Symposium on Research in Computer Security*, Lecture Notes in Computer Science, Heidelberg, Germany, September 2005. Springer-Verlag.

[70] John R. Douceur. The Sybil Attack. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, IPTPS '02, pages 251–260, London, UK, 2002. Springer-Verlag.

[71] H. Rowaihy, W. Enck, P. McDaniel, and T.L. Porta. Limiting Sybil Attacks in Structured Peer-to-Peer Networks. Technical Report NAS-TR-0017-2005, Network and Security Research Center, Department of Computer Science and Engineering, Pennsylvania State University, 2005.

[72] Ingmar Baumgart and Sebastian Mies. S/Kademlia: A Practicable Approach Towards Secure Key-Based Routing. In *Proceedings of the 13th International Conference on Parallel and Distributed Systems - Volume 02*, pages 1–8, Washington, DC, USA, 2007. IEEE Computer Society.

[73] Atul Singh, Miguel Castro, Peter Druschel, and Antony Rowstron. Defending against Eclipse Attacks on Overlay Networks. In *Proceedings of the 11th workshop on ACM SIGOPS European workshop*, EW 11, New York, NY, USA, 2004. ACM.

[74] A. Singh, T-W. Ngan, P. Druschel, and D. S. Wallach. Eclipse Attacks on Overlay Networks: Threats and Defenses. In *Proceedings of the 25th IEEE International Conference on Computer Communications*, INFOCOM'06, pages 1–12, Washington, DC, USA, April 2006. IEEE Computer Society.

[75] Baruch Awerbuch and Christian Scheideler. Towards a scalable and robust DHT. In *Proceedings of the eighteenth annual ACM symposium on Parallelism in algorithms and architectures*, SPAA '06, pages 318–327, New York, NY, USA, 2006. ACM.

[76] L. P. Gaspary, M. P. Barcellos, A. Detsch, and R. S. Antunes. Flexible security in peer-to-peer applications: Enabling new opportunities beyond file sharing. *Computer Networks*, 51, 2007.

[77] Seth James Nielson and Scott A. Crosby. A Taxonomy of Rational Attacks. In *4th Int Workshop on Peer-to-Peer Systems*, volume 3640 of *Lecture Notes in Computer Science*, pages 36–46, Heidelberg, Germany, 2005. Springer-Verlag.

[78] Yongdae Kim, Daniele Mazzocchi, and Gene Tsudik. Admission Control in Peer Groups. In *Proceedings of the Second IEEE International Symposium on Network Computing and Applications*, NCA '03, Washington, DC, USA, 2003. IEEE Computer Society.

[79] Nitesh Saxena, Gene Tsudik, and Jeong Hyun Yi. Admission Control in Peer-to-Peer: Design and Performance Evaluation. In *Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*, SASN '03, pages 104–113, New York, NY, USA, 2003. ACM.

[80] Guido Urdaneta, Guillaume Pierre, and Maarten Van Steen. A survey of DHT security techniques. *ACM Comput. Surv.*, 43:8:1–8:49, February 2011.

[81] Skype Official Website. http://www.skype.com, 2003.

[82] Salman A. Baset and Henning Schulzrinne. An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol. In *Proceedings of the 25th IEEE International Conference on Computer Communications*, pages 1–11, Washington, DC, USA, April 2006. IEEE Computer Society.

[83] The Kazaa Website. http://www.kazaa.com, 2001.

[84] Kundan Singh and Henning Schulzrinne. Peer-to-peer internet telephony using SIP. In *Proceedings of the international workshop on Network and operating systems support for digital audio and video*, 2005.

[85] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset, and H. Schulzrinne. Internet-Draft: A SIP Usage for RELOAD. draft-ietf-p2psip-sip-05 (work in progress), July 2010.

[86] H. Song, X. Jiang, R. Even, and D. Bryan. Internet-Draft: P2PSIP Overlay Diagnostics. draft-ietf-p2psip-diagnostics-05 (work in progress), January 2011.

[87] J. Maenpaa, G. Camarillo, and J. Hautakorpi. Internet-Draft: A Self-tuning Distributed Hash Table (DHT) for REsource LOcation And Discovery (RELOAD). draft-maenpaa-p2psip-self-tuning-03 (work in progress), January 2011.

[88] J. Maenpaa and G. Camarillo. Internet-Draft: Service Discovery Usage for REsource LOcation And Discovery (RELOAD). draft-ietf-p2psip-service-discovery-02.txt (work in progress), January 2011.

[89] ITU-T. Recommendation X.509: The Directory: Public key and attribute certificate frameworks. Technical report, ITU, 2000.

[90] E. Rescorla and N. Modadugu. Datagram Transport Layer Security. RFC 4347 (Proposed Standard), April 2006.

[91] P. Eronen and H. Tschofenig. Pre-Shared Key Ciphersuites for Transport Layer Security (TLS). RFC 4279 (Proposed Standard), December 2005.

[92] D. Taylor, T. Wu, N. Mavrogiannopoulos, and T. Perrin. Using the Secure Remote Password (SRP) Protocol for TLS Authentication. RFC 5054 (Informational), November 2007.

[93] X. Jiang, N. Zong, R. Even, and Y. Zhang. Internet-Draft: An extension to RELOAD to support Direct Response and Relay Peer routing. draft-jiang-p2psip-relay-05 (work in progress), March 2011.

[94] Jan Seedorf. Security Challenges for Peer-to-Peer SIP. *IEEE Network*, 20(5):38–45, October 2006.

[95] David A. Bryan, Marcia Zangrilli, and Bruce B. Lowekamp. Challenges of DHT Design for a Public Communications Systems. Technical Report WM-CS-2006-03, William and Mary, June 2006.

[96] D. Bryan, B. Lowekamp, and M. Zangrilli. The Design of a Versatile, Secure P2PSIP Communications Architecture for the Public Internet. In *IEEE International Symposium on Parallel and Distributed Processing, IPDPS*, pages 1–8, Washington, DC, USA, April 2008. IEEE Computer Society.

[97] J. Seedorf. Using Cryptographically Generated SIP-URIs to Protect the Integrity of Content in P2P-SIP. In *3rd Annual VoIP Security Workshop*, New York, NY, USA, June 2006. ACM.

[98] H. Song, X. Jiang, M. Matuszewski, J-E. Ekberg, and P. Laitinen. Internet-Draft: Security requirements in Peer-to-Peer Session Initiation Protocol. draft-matuszewski-p2psip-security-requirements-06 (work in progress), September 2009.

[99] Information Technology Laboratory, NIST, Gaithersburg, USA. *FIPS 180-3. Secure Hash Standard*, 2008.

[100] Tyson Condie, Varun Kacholia, Sriram Sankararaman, Petros Maniatis, and Joseph M. Hellerstein. Maelstrom: Churn as Shelter. Technical Report UCB/EECS-2005-11, University of California Berkeley, November 2005.

[101] Ralph C. Merkle. Secure Communications Over Insecure Channels. *Commun. ACM*, 21:294–299, April 1978.

[102] Nikita Borisov. Computational Puzzles as Sybil Defenses. In *Proceedings of the Sixth IEEE International Conference on Peer-to-Peer Computing*, P2P '06, pages 171–176, Washington, DC, USA, 2006. IEEE Computer Society.

[103] Luis von Ahn, Manuel Blum, Nicholas Hopper, and John Langford. The Official CAPTCHA Site. http://www.captcha.net/.

[104] Philip R. Zimmermann. *The official PGP user's guide*. MIT Press, Cambridge, MA, 1995.

[105] Srdjan Čapkun, Levente Buttyán, and Jean-Pierre Hubaux. Self-Organized Public-Key Management for Mobile Ad Hoc Networks. *IEEE Transactions on Mobile Computing*, 2(1):52–64, January 2003.

[106] Lidong Zhou and Zygmunt J. Haas. Securing Ad Hoc Networks. *IEEE Network Magazine*, 13(6):24–30, November 1999.

[107] Yvo Desmedt and Yair Frankel. Threshold Cryptosystems. In *Proceedings of the 9th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO'89, pages 307–315, London, UK, 1989. Springer-Verlag.

[108] Jiejun Kong, Petros Zerfos, Haiyun Luo, Songwu Lu, and Lixia Zhang. Providing Robust and Ubiquitous Security Support for Mobile Ad Hoc Networks. In *Proceedings of the Ninth International Conference on Network Protocols*, ICNP '01, Washington, DC, USA, 2001. IEEE Computer Society.

[109] Adi Shamir. How to Share a Secret. *Communications of the ACM*, 22(11):612–613, November 1979.

[110] Torben Pryds Pedersen. A Threshold Cryptosystem without a Trusted Party. In *Proceedings of the 10th annual international conference on Theory and application of cryptographic techniques*, EUROCRYPT'91, pages 522–526, Berlin, Heidelberg, 1991. Springer-Verlag.

[111] Nitesh Saxena, Gene Tsudik, and Jeong Hyun Yi. Threshold cryptography in P2P and MANETs: The case of access control. *Computer Networks*, 51(12):3632–3649, 2007.

[112] Johann Van Der Merwe, Dawoud Dawoud, and Stephen McDonald. A Survey on Peer-to-Peer Key Management for Mobile Ad Hoc Networks. *ACM Comput. Surv.*, 39(1), 2007.

[113] Kevin R. B. Butler, Sunam Ryu, Patrick Traynor, and Patrick D. McDaniel. Leveraging Identity-Based Cryptography for Node ID Assignment in Structured P2P Systems. *IEEE Trans. Parallel Distrib. Syst.*, 20(12):1803–1815, 2009.

[114] Adi Shamir. Identity-based cryptosystems and signature schemes. In *Proceedings of CRYPTO 84 on Advances in cryptology*, pages 47–53, New York, NY, USA, 1985. Springer-Verlag New York, Inc.

[115] Haifeng Yu, Michael Kaminsky, Phillip B. Gibbons, and Abraham Flaxman. SybilGuard: Defending Against Sybil Attacks via Social Networks. *SIGCOMM Comput. Commun. Rev.*, 36:267–278, August 2006.

[116] Haifeng Yu, Phillip B. Gibbons, Michael Kaminsky, and Feng Xiao. SybilLimit: A Near-Optimal Social Network Defense against Sybil Attacks. In *Proceedings of the 2008 IEEE Symposium on Security and Privacy*, pages 3–17, Washington, DC, USA, 2008. IEEE Computer Society.

[117] George Danezis and Prateek Mittal. SybilInfer: Detecting Sybil Nodes using Social Networks. In *16th Annual Network & Distributed System Security Symposium*, NDSS'09. The Internet Society, February 2009.

[118] N. Asokan and P. Ginzboorg. Key-Agreement in Ad Hoc Networks. *Computer Communications*, 23(17):1627–1637, November 2000.

[119] Newso James, Elaine Shi, Dawn Song, and Adrian Perrig. The Sybil Attack in Sensor Networks: Analysis & Defenses. In *Proceedings of the 3rd international symposium on Information processing in sensor networks*, IPSN '04, pages 259–268, New York, NY, USA, 2004. ACM.

[120] Ravi Sandhu and Xinwen Zhang. Peer-to-Peer Access Control Architecture Using Trusted Computing Technology. In *Proceedings of the tenth ACM symposium on Access control models and technologies*, SACMAT '05, pages 147–158, New York, NY, USA, 2005. ACM.

[121] Yatin Chawathe, Sylvia Ratnasamy, Lee Breslau, Nick Lanham, and Scott Shenker. Making Gnutella-like P2P Systems Scalable. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '03, pages 407–418, New York, NY, USA, 2003. ACM.

[122] Jochen Dinger and Oliver P. Waldhorst. Decentralized Bootstrapping of P2P Systems: A Practical View. In *Proceedings of the 8th International IFIP-TC 6 Networking Conference*, NETWORKING '09, pages 703–715, Heidelberg, Germany, 2009. Springer-Verlag.

[123] C. Cramer, K. Kutzner, and T. Fuhrmann. Bootstrapping locality-aware P2P networks. In *Proceedings of the 12th IEEE International Conference on Networks*, volume 1 of *ICON 2004*, pages 357–361, Nov. 2004.

[124] Chris Gauthier Dickey and Christian Grothoff. Bootstrapping of Peer-to-Peer Networks. In *International Workshop on Dependable and Sustainable Peer-to-Peer Systems*, DAS-P2P 2008, Washington, DC, USA, August 2008. IEEE Computer Society.

[125] Leonard Newton Foner. *Political artifacts and personal privacy: the yenta multiagent distributed matchmaking system*. PhD thesis, Massachusetts Institute of Technology, 1999. Supervisor-Maes, Pattie.

[126] Erik Guttman. Service Location Protocol: Automatic Discovery of IP Network Services. *IEEE Internet Computing*, 3:71–80, July 1999.

[127] J. Oikarinen and D. Reed. Internet Relay Chat Protocol. RFC 1459 (Experimental), May 1993. Updated by RFCs 2810, 2811, 2812, 2813.

[128] Gnutella Web Caching System. http://www.gnucleus.com/gwebcache/index.html, 2008.

[129] P. Karbhari, M. Ammar, A. Dhamdhere, H. Raj, G. Riley, and E. Zepra. Bootstrapping in Gnutella: A Measurement Study. In *Proceedings of the Passive and Active Measurements Workshop*, Lecture Notes in Computer Science, Heidelberg, Germany, April 2004. Springer.

[130] G. Garcia. Internet-Draft: P2PSIP bootstrapping using DNS-SD. draft-garcia-p2psip-dns-sd-bootstraping-00 (work in progress), October 2007.

[131] Mirko Knoll, Arno Wacker, Gregor Schiele, and Torben Weis. Bootstrapping in Peer-to-Peer Systems. In *Proceedings of the 2008 14th IEEE International Conference on Parallel and Distributed Systems*, ICPADS '08, pages 271–278, Washington, DC, USA, 2008. IEEE Computer Society.

[132] Villu Arak. What happened on August 16. http://heartbeat.skype.com/2007/08/what_happened_on_august_16.html, 2007.

[133] Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, and Antony Rowstron. One Ring to Rule Them All: Service Discovery and Binding in Structured Peer-to-Peer Overlay Networks. In *Proceedings of the 10th workshop on ACM SIGOPS European workshop*, EW 10, pages 140–145, New York, NY, USA, 2002. ACM.

[134] Mark Jelasity, Alberto Montresor, and Ozalp Babaoglu. The Bootstrapping Service. In *Proceedings of the 26th IEEE International ConferenceWorkshops on Distributed Computing Systems*, ICDCSW '06, Washington, DC, USA, 2006. IEEE Computer Society.

[135] Michael Conrad and Hans-Joachim Hof. A Generic, Self-organizing, and Distributed Bootstrap Service for Peer-to-Peer Networks. In *International Workshop on Self-Organizing Systems*, volume 4725 of *Lecture Notes in Computer Science*, pages 59–72, Heidelberg, Germany, August 2007. Springer-Verlag.

[136] Zengming Tian, Xiangming Wen, Wei Zheng, Yong Sun, and Yinbo Cheng. Evaluation and Simulation on the Performance of DHTs Required by P2PSIP. In *Proceedings of the 2009 Fifth International Conference on Information Assurance and Security - Volume 01*, IAS '09, pages 701–704, Washington, DC, USA, 2009. IEEE Computer Society.

[137] J. Jiang, R. Pan, C. Liang, and W. Wang. BiChord: An Improved Approach for Lookup Routing in Chord. In *Proceedings of the 9th Conference on Advances in Databases and Information Systems*, volume 3631 of *Lecture Notes in Computer Science*, Heidelberg, Germany, September 2005. Springer-Verlag.

[138] B. Leong, B. Liskov, and E. D. Demaine. EpiChord: Parallelizing the Chord Lookup Algorithm with Reactive Routing State Management. In *Proceedings of the 12th IEEE In-*

ternational Conference on Networks, ICN'04, Heidelberg, Germany, March 2004. Springer-Verlag.

[139] Sergio Marti, NewAuthor2, and Hector Garcia-Molina. SPROUT: P2P Routing with Social Networks. In *Current Trends in Database Technology - EDBT 2004 Workshops*, volume 3268 of *Lecture Notes in Computer Science*, Heidelberg, Germany, March 2004. Springer-Verlag.

[140] Sergio Marti Prasanna, Prasanna Ganesan, and Hector Garcia-molina. DHT Routing Using Social Links. In *International Workshop on Peer-To-Peer Systems (IPTPS)*, Lecture Notes in Computer Science, pages 100–111, Heidelberg, Germany, February 2004. Springer-Verlag.

[141] K. Gummadi, R. Gummadi, S. Gribble, S. Ratnasamy, S. Shenker, and I. Stoica. The impact of DHT routing geometry on resilience and proximity. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '03, pages 381–394, New York, NY, USA, 2003. ACM.

[142] Information Technology Laboratory, NIST, Gaithersburg, USA. *FIPS 186-3: Digital Signature Standard (DSS)*, 2009.

[143] Jianhong Zhang, Dan Chen, and Yumin Wang. On the security of a digital signature with message recovery using self-certified public key. In *Proceedings of the 4th WSEAS International Conference on Applied Mathematics and Computer Science*, pages 24:1–24:8, Stevens Point, Wisconsin, USA, 2005. World Scientific and Engineering Academy and Society (WSEAS).

[144] Miguel Viedma Astudillo, Jesús Téllez Isaac, Diego Suarez Touceda, and Héctor Plaza López. Evaluation of a Client Centric Payment Protocol Using Digital Signature Scheme with Message Recovery Using Self-Certified Public Key. In *Proceedings of the International Conference on Computational Science and Its Applications: Part II*, ICCSA '09, pages 155–163, Berlin, Heidelberg, 2009. Springer-Verlag.

[145] Paul Syverson. A Taxonomy of Replay Attacks. In *Proceedings of the 7th IEEE Computer Security Foundations Workshop*, CSFW'94, pages 187–191, Washington, DC, USA, June 1994. IEEE Computer Society.

[146] Li Gong. Variations on the Message Freshness and Replay. In *In Proceedings of the 6th IEEE Computer Security Foundations Workshop*, CSFW'93, pages 131–136, Washington, DC, USA, 1993. IEEE Computer Society.

[147] David Chaum, Communications Of The Acm, R. Rivest, and David L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24:84–90, February 1981.

[148] GNUnet: GNU's decentralized anonymous and censorship-resistant P2P framework. http://gnunet.org/, 2002.

[149] Bennett and Grothoff. GAP – Practical Anonymous Networking. In *International Workshop on Privacy Enhancing Technologies (PET)*, volume 3483 of *Lecture Notes in Computer Science*, Heidelberg, Germany, 2003. Springer-Verlag.

[150] Roger Dingledine, Nick Mathewson, and Paul Syverson. TOR: The Second-Generation Onion Router. In *Proceedings of the 13th conference on USENIX Security Symposium - Volume 13*, SSYM'04, pages 21–21, Berkeley, CA, USA, 2004. USENIX Association.

[151] Michael G. Reed, Paul F. Syverson, and David M. Goldschlag. Anonymous Connections and Onion Routing. *IEEE Journal on Selected Areas in Communications*, 16:482–494, 1998.

[152] Michael J. Freedman and Robert Morris. Tarzan: A Peer-to-Peer Anonymizing Network Layer. In *Proceedings of the 9th ACM conference on Computer and communications security*, CCS '02, pages 193–206, New York, NY, USA, 2002. ACM.

[153] Marc Rennhard and Bernhard Plattner. Introducing MorphMix: Peer-to-Peer Based Anonymous Internet Usage with Collusion Detection. In Sabrina De Capitani di Vimercati and Pierangela Samarati, editors, *Proceeding of the ACM workshop on Privacy in the Electronic Society (WPES-02)*, pages 91–102, New York, November 2002. ACM Press.

[154] Ali Fessi, Nathan Evans, Heiko Niedermayer, and Ralph Holz. Pr2-P2PSIP: privacy preserving P2P signaling for VoIP and IM. In *Principles, Systems and Applications of IP Telecommunications*, IPTComm '10, pages 134–145, New York, NY, USA, 2010. ACM.

[155] Charles W. O'Donnell and Vinod Vaikuntanathan. Information Leak in the Chord Lookup Protocol. In *Proceedings of the Fourth International Conference on Peer-to-Peer Computing*, P2P '04, pages 28–35, Washington, DC, USA, 2004. IEEE Computer Society.

[156] Nikita Borisov and Jason Waddle. Anonymity in Structured Peer-to-Peer Networks. Technical Report UCB/CSD-05-1390, EECS Department, University of California, Berkeley, May 2005.

[157] Alan Mislove, Gaurav Oberoi, Ansley Post, Charles Reis, Peter Druschel, and Dan S. Wallach. AP3: cooperative, decentralized anonymous communication. In *Proceedings of*

the 11th workshop on ACM SIGOPS European workshop, EW 11, New York, NY, USA, September 2004. ACM.

[158] Neil Daswani and Hector Garcia-Molina. Query-Flood DoS Attacks in Gnutella. In *Proceedings of the 9th ACM conference on Computer and communications security*, CCS '02, pages 181–192, New York, NY, USA, 2002. ACM.

[159] Nicholas A. Fraser, Douglas J. Kelly, Richard A. Raines, Rusty O. Baldwin, and Barry E. Mullins. Using Client Puzzles to Mitigate Distributed Denial of Service Attacks in the Tor Anonymous Routing Environment. In *Proceedings of IEEE International Conference on Communications*, ICC 2007, pages 1197–1202, Washington, DC, USA, 2007. IEEE Computer Society.

[160] Drew Dean and Adam Stubblefield. Using Client Puzzles to Protect TLS. In *Proceedings of the 10th conference on USENIX Security Symposium*, volume 10 of *SSYM'01*, Berkeley, CA, USA, 2001. USENIX Association.

[161] XiaoFeng Wang and Michael K. Reiter. A Multi-Layer Framework for Puzzle-Based Denial-of-Service Defense. *Int. J. Inf. Secur.*, 7:243–263, July 2008.

[162] Wu chi Feng, Edward C. Kaiser, and A. Luu. Design and implementation of network puzzles. In *24th Annual Joint Conference of the IEEE Computer and Communications Societies*, INFOCOM 2005, pages 2372–2382, Washington, DC, USA, March 2005. IEEE Computer Society.

[163] Roger M. Needham. Denial of Service. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, CCS '93, pages 151–153, New York, NY, USA, 1993. ACM.

[164] Lee Garber. Denial-of-Service Attacks Rip the Internet. *IEEE Computer*, 33:12–17, April 2000.

[165] P. Ferguson and D. Senie. Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing. RFC 2827 (Best Current Practice), May 2000. Updated by RFC 3704.

[166] M. Handley, E. Rescorla, and IAB. Internet Denial-of-Service Considerations. RFC 4732 (Informational), December 2006.

[167] John Kubiatowicz, David Bindel, Yan Chen, Steven Czerwinski, Patrick Eaton, Dennis Geels, Ramakrishan Gummadi, Sean Rhea, Hakim Weatherspoon, Westley Weimer, Chris

Wells, and Ben Zhao. OceanStore: An Architecture for Global-Scale Persistent Storage. *ACM SIGPLAN Notices*, 35(11):190–201, 2000.

[168] Atul Adya, William J. Bolosky, Miguel Castro, Gerald Cermak, Ronnie Chaiken, John R. Douceur, Jon Howell, Jacob R. Lorch, Marvin Theimer, and Roger P. Wattenhofer. FARSITE: Federated, Available, and Reliable Storage for an Incompletely Trusted Environment. In *Proceedings of the 5th symposium on Operating systems design and implementation*, OSDI '02, pages 1–14, New York, NY, USA, 2002. ACM.

[169] David A. Bryan and Bruce Lowekamp. Innovations in Peer-to-Peer Communications. In *Proceedings of the 2006 Virginia Space Grant Consortium Research Conference*, April 2006.

[170] Feng Cao, David A. Bryan, and Bruce B. Lowekamp. Providing Secure Services in Peer-to-Peer Communications Networks with Central Security Servers. In *AICT-ICIW '06: Proceedings of the Advanced Int'l Conference on Telecommunications and Int'l Conference on Internet and Web Applications and Services*, page 105, Washington, DC, USA, 2006. IEEE Computer Society.

[171] Peter Druschel and Antony Rowstron. PAST: A Large-Scale, Persistent Peer-to-Peer Storage Utility. In *Proceedings of the Eighth Workshop on Hot Topics in Operating Systems*, pages 75–80, Washington, DC, USA, 2001. IEEE Computer Society.

[172] Kevin Hoffman, David Zage, and Cristina Nita-Rotaru. A Survey of Attack and Defense Techniques for Reputation Systems. *ACM Computing Surveys*, 42(1), 2010.

[173] Ali Ghodsi, Luc Onana Alima, and Seif Haridi. Symmetric Replication for Structured Peer-to-Peer Systems. In *Proceedings of the 2005/2006 international conference on Databases, information systems, and peer-to-peer computing*, DBISP2P'05/06, pages 74–85, Heidelberg, Germany, 2007. Springer-Verlag.

[174] Di Wu, Ye Tian, Kam-Wing Ng, and Anwitaman Datta. Stochastic analysis of the interplay between object maintenance and churn. *Computer Communications*, 31(2):220–239, 2008.

[175] Irving Reed and Solomon Golomb. Polynomial codes over certain finite fields. *Joint Society of Industrial and Applied Mathematics Journal*, 8(2):300–304, jun 1960.

[176] John W. Byers, Michael Luby, Michael Mitzenmacher, and Ashutosh Rege. A digital fountain approach to reliable distribution of bulk data. *SIGCOMM Comput. Commun. Rev.*, 28(4):56–67, 1998.

[177] Hakim Weatherspoon and John Kubiatowicz. Erasure Coding Vs. Replication: A Quantitative Comparison. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, IPTPS '01, pages 328–338, London, UK, 2002. Springer-Verlag.

[178] Ranjita Bhagwan, Kiran Tati, Yu-Chung Cheng, Stefan Savage, and Geoffrey M. Voelker. Total Recall: System Support for Automated Availability Management. In *Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation*, NSDI'04, Berkeley, CA, USA, 2004. USENIX Association.

[179] Charles Blake and Rodrigo Rodrigues. High Availability, Scalable Storage, Dynamic Peer Networks: Pick Two. In *Proceedings of the 9th conference on Hot Topics in Operating Systems*, HOTOS'03, Berkeley, CA, USA, 2003. USENIX Association.

[180] Rodrigo Rodrigues and Barbara Liskov. High Availability in DHTs: Erasure Coding vs. Replication. In *Peer-to-Peer Systems IV 4th International Workshop*, IPTPS 2005, Ithaca, New York, feb 2005.

[181] Anwitaman Datta and Karl Aberer. Internet-Scale Storage Systems under Churn – A Study of the Steady-State using Markov Models. In *Proceedings of the Sixth IEEE International Conference on Peer-to-Peer Computing*, P2P '06, pages 133–144, Washington, DC, USA, 2006. IEEE Computer Society.

[182] W. K. Lin, D. M. Chiu, and Y. B. Lee. Erasure Code Replication Revisited. In *Proceedings of the Fourth International Conference on Peer-to-Peer Computing*, P2P '04, pages 90–97, Washington, DC, USA, 2004. IEEE Computer Society.

[183] Frank Dabek, Jinyang Li, Emil Sit, James Robertson, M. Frans Kaashoek, and Robert Morris. Designing a DHT for low latency and high throughput. In *Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation*, NSDI'04, Berkeley, CA, USA, 2004. USENIX Association.

[184] Dorgham Sisalem, Jiri Kuthan, and Sven Ehlert. Denial of service attacks targeting a SIP VoIP infrastructure: Attack Scenarios and Prevention Mechanisms. *IEEE Network*, 20(5):26–31, 2006.

[185] B. Iancu. PIKE Excessive Traffic Monitorig Module. http://www.iptel.org/ser/doc/modules/pike, 2003.

[186] D. J. Bernstein. SYN Cookies. http://cr.yp.to/syncookies.html, 2002.

[187] Dimitris Geneiatakis, Tasos Dagiuklas, Georgios Kambourakis, Costas Lambrinoudakis, Stefanos Gritzalis, Sven Ehlert, and Dorgham Sisalem. Survey of Security Vulnerabilities

in Session Initiation Protocol. *IEEE Communications Surveys and Tutorials*, 8(1-4):68–81, 2006.

[188] Christian Wieser, Marko Laakso, and Henning Schulzrinne. Security testing of SIP implementations. Technical Report CUCS-24-03, Dept. of Computer Science, Columbia Univ., 2003.

[189] Dimitris Geneiatakis, Georgios Kambourakis, Costas Lambrinoudakis, Tasos Dagiuklas, and Stefanos Gritzalis. A framework for protecting a SIP-based infrastructure against malformed message attacks. *Computer Networks*, 51(10):2580–2593, 2007.

[190] Sven Ehlert, Ge Zhang, Dimitris Geneiatakis, Georgios Kambourakis, Tasos Dagiuklas, Jiří Markl, and Dorgham Sisalem. Two layer Denial of Service prevention on SIP VoIP infrastructures. *Computer Communications*, 31(10):2443–2456, 2008.

[191] D. Butcher, X. Li, and Jinhua Guo. Security Challenge and Defense in VoIP Infrastructures. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 37(6):1152–1162, 2007.

[192] Francesco Palmieri and Ugo Fiore. Providing true end-to-end security in converged voice over IP infrastructures. *Computers & Security*, 28(6):433–449, 2009.

[193] B. Ramsdell. Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification. RFC 3851 (Proposed Standard), July 2004.

[194] G. Camarillo. The Internet Assigned Number Authority (IANA) Uniform Resource Identifier (URI) Parameter Registry for the Session Initiation Protocol (SIP). RFC 3969 (Best Current Practice), December 2004.

[195] B. Ramsdell. Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Certificate Handling. RFC 3850 (Proposed Standard), July 2004.

[196] Johan Bilien, Erik Eliasson, Joachim Orrblad, and Jon olov Vatn. Secure VoIP: Call Establishment and Media Protection. In *2nd Workshop on Securing Voice over IP*, June 2005.

[197] V.K. Gurbani and V. Kolesnikov. A Survey and Analysis of Media Keying Techniques in the Session Initiation Protocol (SIP). *Communications Surveys and Tutorials, IEEE*, 13(2):183–198, quarter 2011.

[198] Dorgham Sisalem, John Floroiu, Jiri Kuthan, Ulrich Abend, and Henning Schulzrinne. *SIP Security*. WILEY, 2009.

[199] J. Quittek, S. Niccolini, S. Tartarelli, and R. Schlegel. On Spam over Internet Telephony (SPIT) Prevention. *Communications Magazine, IEEE*, 46(8):80–86, august 2008.

[200] P. Wolfe, C. Scott, and M. Erwin. *Anti-Spam Tool Kit.* McGraw-Hill Osborne Media, New York, 2004.

[201] J. Rosenberg, G. Camarillo, and D. Willis. Requirements for Consent-Based Communications in the Session Initiation Protocol (SIP). RFC 4453 (Informational), April 2006.

[202] J. Rosenberg, G. Camarillo, and D. Willis. A Framework for Consent-Based Communications in the Session Initiation Protocol (SIP). RFC 5360 (Proposed Standard), October 2008.

[203] Paul Resnick, Ko Kuwabara, Richard Zeckhauser, and Eric Friedman. Reputation Systems. *Commun. ACM*, 43(12):45–48, 2000.

[204] Joakim Koskela, Juho Heikkilä, and Andrei Gurtov. Poster abstract: a secure P2P SIP system with SPAM prevention. *SIGMOBILE Mob. Comput. Commun. Rev.*, 13:26–29, January 2010.

[205] Juho Heikkilä and Andrei Gurtov. Filtering SPAM in P2PSIP Communities with Web of Trust. In *Proceedings of the MobiSec'09*, pages 110–121, Heidelberg, Germany, 2009. Springer-Verlag.

[206] J. Rosenberg. A Framework for Application Interaction in the Session Initiation Protocol (SIP). RFC 5629 (Proposed Standard), October 2009.

[207] C. Jennings. Internet-Draft: Computational Puzzles for SPAM Reduction in SIP. draft-jennings-sip-hashcash-06 (work in progress), July 2007.

[208] Ben Laurie and Richard Clayton. Proof-of-Work Proves Not to Work. In *Third Annual Workshop on Economics and Information Security*, WEIS'04, May 2004.

[209] Martín Abadi, Andrew Birrell, Mike Burrows, Frank Dabek, and Ted Wobber. Bankable Postage for Network Services. In *In Proc. Asian Computing Science Conference*, volume 2896 of *Lecture Notes in Computer Science*, pages 72–90, Heidelberg, Germany, 2003. Springer-Verlag.

[210] C. Jennings, J. Fischl, H. Tschofenig, and G. Jun. Internet-Draft: Payment for Services in Session Initiation Protocol (SIP). draft-jennings-sipping-pay-06 (work in progress), July 2007.

[211] Diego Suarez, Joaquín Torres Márquez, Mildrey Carbonell, and Jesús Téllez. A new domain-based payment model for emerging mobile commerce scenarios. In *International Conference on Database and Expert Systems Applications*, DEXA Workshops 2007, pages 713–717, 2007.

[212] European Parliament. Directive 2002/58/EC of the European Parliament and of the Council of 12 July 2002 Concerning the Processing of Personal Data and the Protection of Privacy in the Electronic Communications Sector (Directive on Privacy and Electronic Communications). Official Journal of the European Communities, July 2002.

[213] Federal Trade Commission. The CAN-SPAM act: Requirements for commercial emailers. Public Law, April 2004.

[214] M. Handley and V. Jacobson. SDP: Session Description Protocol. RFC 2327 (Proposed Standard), April 1998. Obsoleted by RFC 4566, updated by RFC 3266.

[215] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. RFC 3550 (Standard), July 2003. Updated by RFC 5506.

[216] M. Baugher, D. McGrew, M. Naslund, E. Carrara, and K. Norrman. The Secure Real-time Transport Protocol (SRTP). RFC 3711 (Proposed Standard), March 2004. Updated by RFC 5506.

[217] Ram Dantu, Sonia Fahmy, Henning Schulzrinne, and João Cangussu. Issues and challenges in securing VoIP. *Computers & Security*, 28(8):743–753, November 2009.

[218] VoIP Security Alliance. VoIP Security and Privacy Threat Taxonomy. Public Release, Oct 2005.

[219] A. Keromytis. A Comprehensive Survey of Voice over IP Security Research. *Communications Surveys Tutorials, IEEE*, PP(99):1–24, 2011.

[220] Sven Ehlert, Dimitris Geneiatakis, and Thomas Magedanz. Survey of network security systems to counter SIP-based denial-of-service attacks. *Computers & Security*, 29(2):225–243, 2010.

[221] A. Johnston and D. Piscitello. *Understanding VoIP Security.* Artech House, Boston, MA, 2006.

[222] F. Andreasen, M. Baugher, and D. Wing. Session Description Protocol (SDP) Security Descriptions for Media Streams. RFC 4568 (Proposed Standard), July 2006.

[223] P. Zimmermann, A. Johnston, and J. Callas. ZRTP: Media Path Key Agreement for Unicast Secure RTP. RFC 6189 (Proposed Standard), April 2011.

[224] D. McGrew and E. Rescorla. Datagram Transport Layer Security (DTLS) Extension to Establish Keys for the Secure Real-time Transport Protocol (SRTP). RFC 5764 (Proposed Standard), May 2010.

[225] S. Bradner, E. L. Conroy, and K. Fujiwara. The E.164 to Uniform Resource Identifiers (URI) Dynamic Delegation Discovery System (DDDS) Application (ENUM). RFC 6116 (Proposed Standard), March 2011.

[226] Philippe Golle, Kevin Leyton-Brown, and Ilya Mironov. Incentives for Sharing in Peer-to-Peer Networks. In *Proceedings of the 3rd ACM conference on Electronic Commerce*, EC '01, pages 264–267, New York, NY, USA, 2001. ACM.

[227] Michal Feldman, John Chuang, Ion Stoica, and Scott Shenker. Hidden-Action in Multi-Hop Routing. In *Proceedings of the 6th ACM conference on Electronic commerce*, EC '05, pages 117–126, New York, NY, USA, 2005. ACM.

[228] Markus Jakobsson, Jean-Pierre Hubaux, and Levente Buttyan. A Micro-Payment Scheme Encouraging Collaboration in Multi-Hop Cellular Networks. In *Proceedings of International Financial Cryptography Conference*. International Financial Cryptography Association, January 2003.

[229] Michal Feldman and John Chuang. Overcoming Free-riding Behavior in Peer-to-Peer Systems. *ACM SIGecom Exch.*, 5:41–50, July 2005.

[230] Bram Cohen. Incentives Build Robustness in BitTorrent. In *Proceedings of the 1st Workshop on Economics of Peer-to-Peer Systems*, P2PECON'03, June 2003.

[231] T.-W. J. Ngan, D. S. Wallach, and P. Druschel. Incentive-compatible Peer-to-Peer multicast. In *Proceedings of the 2nd Workshop on Economics of Peer-to-Peer Systems*, P2PECON'04, June 2004.

[232] Yang hua Chu, John Chuang, and Hui Zhang. A Case for Taxation in Peer-to-Peer Streaming Broadcast. In *Proceedings of the ACM SIGCOMM workshop on Practice and theory of incentives in networked systems*, PINS '04, pages 205–212, New York, NY, USA, 2004. ACM.

[233] Michal Feldman, Kevin Lai, Ion Stoica, and John Chuang. Robust Incentive Techniques for Peer-to-Peer Networks. In *Proceedings of the 5th ACM conference on Electronic commerce*, EC '04, pages 102–111, New York, NY, USA, 2004. ACM.

[234] Murat Karakaya, İbrahim Körpeoğlu, and Özgür Ulusoy. Counteracting free riding in Peer-to-Peer networks. *Comput. Netw.*, 52(3):675–694, 2008.

[235] Wangkee Poon and Rocky K. C. Chang. Robust Forwarding in Structured Peer to Peer Overlay Networks. In *Proceedings of ACM SIGCOMM 2004*, New York, NY, USA, August 2004. ACM.

[236] Xianghan Zheng and V. Oleshchuk. Trust-based framework for security enhancement of P2PSIP communication systems. In *International Conference for Internet Technology and Secured Transactions*, ICITST 2009, pages 1–6, Washington, DC, USA, 2009. IEEE Computer Society.

[237] Sean Rhea, Brighten Godfrey, Brad Karp, John Kubiatowicz, Sylvia Ratnasamy, Scott Shenker, Ion Stoica, and Harlan Yu. OpenDHT: a public DHT service and its uses. In *Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '05, pages 73–84, New York, NY, USA, 2005. ACM.

[238] Sean Rhea, Patrick Eaton, Dennis Geels, Hakim Weatherspoon, Ben Zhao, and John Kubiatowicz. Pond: The OceanStore Prototype. In *Proceedings of the 2nd USENIX Conference on File and Storage Technologies*, FAST '03, pages 1–14, Berkeley, CA, USA, 2003. USENIX Association.

[239] Sean Rhea, Dennis Geels, Timothy Roscoe, and John Kubiatowicz. Handling churn in a DHT. In *Proceedings of the annual conference on USENIX Annual Technical Conference*, ATEC '04, Berkeley, CA, USA, 2004. USENIX Association.

[240] Supriya Krishnamurthy, Sameh El-Ansary, Erik Aurell, and Seif Haridi. Comparing Maintenance Strategies for Overlays. In *Proceedings of the 16th Euromicro Conference on Parallel, Distributed and Network-Based Processing*, PDP '08, pages 473–482, Washington, DC, USA, 2008. IEEE Computer Society.

[241] Emil Sit, Andreas Haeberlen, Frank Dabek, Byung gon Chun, Hakim Weatherspoon, Robert Morris, M. Frans Kaashoek, and John Kubiatowicz. Proactive Replication for Data Durability. In *Proceedings of the 5th Int'l Workshop on Peer-to-Peer Systems*, IPTPS'06, Berkeley, CA, USA, February 2006. USENIX Association.

[242] Gabriel Ghinita and Yong Meng Teo. An adaptive stabilization framework for distributed hash tables. In *Proceedings of the 20th international conference on Parallel and distributed processing*, IPDPS'06, pages 29–29, Washington, DC, USA, 2006. IEEE Computer Society.

[243] Ratul Mahajan, Miguel Castro, and Antony Rowstron. Controlling the Cost of Reliability in Peer-to-Peer Overlays. In *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems*, volume 2735 of *Lecture Notes in Computer Science*, pages 21–32, Heidelberg, Germany, 2003. Springer-Verlag.

[244] James W. Mickens and Brian D. Noble. Exploiting availability prediction in distributed systems. In *Proceedings of the 3rd conference on Networked Systems Design & Implementation*, NSDI'06, Berkeley, CA, USA, 2006. USENIX Association.

[245] J. Li, J. Stribling, F. Kaashoek, R. Morris, and T. Gil. A Performance vs. Cost Framework for Evaluating DHT Design Tradeoffs under Churn. In *24th Annual Joint Conference of the IEEE Computer and Communications Societies*, INFOCOM '05, Washington, DC, USA, March 2005. IEEE Computer Society.

[246] Garrett Hardin. The Tragedy of the Commons. *Science*, 162(3859):1243–1248, 1968.

[247] Eytan Adar and Bernardo A. Huberman. Free Riding on Gnutella. *First Monday*, 5(10), 2000.

[248] Stefan Saroiu, P. Krishna Gummadi, and Steven D. Gribble. A Measurement Study of Peer-to-Peer File Sharing Systems. In *Proceedings of Multimedia Computing and Networking 2002*, MMCN '02, New York, NY, USA, January 2002. ACM.

[249] Daniel Hughes, Geoff Coulson, and James Walkerdine. Free Riding on Gnutella Revisited: The Bell Tolls? *IEEE Distributed Systems Online*, 6(6), 2005.

[250] R.L. Xia and J.K. Muppala. A Survey of BitTorrent Performance. *Communications Surveys and Tutorials, IEEE*, 12(2):140–158, quarter 2010.

[251] R. Guerraoui, K. Huguenin, A.-M. Kermarrec, and M. Monod. On tracking freeriders in gossip protocols. In *IEEE Ninth International Conference on Peer-to-Peer Computing*, P2P '09, Washington, DC, USA, September 2009. IEEE Computer Society.

[252] Jouni Maenpaa and Gonzalo Camarillo. Study on Maintenance Operations in a Chord-Based Peer-to-Peer Session Initiation Protocol Overlay Network. In *Proceedings of the 2009 IEEE International Symposium on Parallel&Distributed Processing*, pages 1–9, Washington, DC, USA, 2009. IEEE Computer Society.

[253] J. Maenpaa and G. Camarillo. Analysis of Delays in a Peer-to-Peer Session Initiation Protocol Overlay Network. In *7th IEEE Consumer Communications and Networking Conference*, CCNC'10, pages 1–6, Washington, DC, USA, 2010. IEEE Computer Society.

[254] J. Maenpaa and G. Camarillo. Estimating Operating Conditions in a Peer-to-Peer Session Initiation Protocol Overlay Network. In *IEEE International Symposium on Parallel Distributed Processing, Workshops and Phd Forum*, IPDPSW'10, pages 1–8, Washington, DC, USA, 2010. IEEE Computer Society.

[255] Yanxiang He, Qiang Cao, Yi Han, Libing Wu, and Tao Liu. Reduction of Quality (RoQ) Attacks on Structured Peer-to-Peer Networks. In *Proceedings of the 2009 IEEE International Symposium on Parallel&Distributed Processing*, IPDPS '09, pages 1–9, Washington, DC, USA, 2009. IEEE Computer Society.

[256] David Liben-Nowell, Hari Balakrishnan, and David Karger. Analysis of the Evolution of Peer-to-Peer Systems. In *PODC '02: Proceedings of the twenty-first annual symposium on Principles of distributed computing*, pages 233–242, New York, NY, USA, 2002. ACM.

[257] Orange Press. A new Orange booth comes to Paris. http://www.orange.com/en_EN/press/press_releases/cp100409en.jsp, Apr 2010.

[258] T.N. Ellahi, B. Hudzia, L. Mcdermott, and T. Kechadi. Security Framework for P2P Based Grid Systems. In *Parallel and Distributed Computing, 2006. ISPDC '06. The Fifth International Symposium on*, pages 230–237, July 2006.

[259] Vipul Gupta, Sumit Gupta, Sheueling Chang, and Douglas Stebila. Performance analysis of elliptic curve cryptography for SSL. In *WiSE '02: Proceedings of the 1st ACM workshop on Wireless security*, pages 87–94, New York, NY, USA, 2002. ACM.

[260] ITU-T. Key Global Telecom Indicators for the World Telecommunication Service Sector. http://www.itu.int/ITU-D/ict/statistics/at_glance/KeyTelecom99.html, 2009.

[261] S. Farrell, R. Housley, and S. Turner. An Internet Attribute Certificate Profile for Authorization. RFC 5755 (Standard), January 2010.

[262] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280 (Proposed Standard), May 2008.

[263] Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Robust threshold DSS signatures. In *Proceedings of the 15th annual international conference on Theory and application of cryptographic techniques*, EUROCRYPT'96, pages 354–371, Berlin, Heidelberg, 1996. Springer-Verlag.

[264] Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Secure Distributed Key Generation for Discrete-Log Based Cryptosystems. *J. Cryptol.*, 20(1):51–83, 2007.

[265] Thomas Blum and Christof Paar. Montgomery modular exponentiation on reconfigurable hardware. In *14th IEEE Symposium on Computer Arithmetic*, ARITH-14, pages 70–77, Washington, DC, USA, 1999. IEEE Computer Society.

[266] René Schoof. Four primality testing algorithms. *Surveys in algorithmic number theory*, 44:101–126, 2008.

[267] B. Lampson. Protection. In *Proc. 5th Princeton Conf. on Information Sciences and Systems*, 1971.

[268] Bell DE and LaPadula LJ. Secure computer system: Unified exposition and Multics interpretation. Technical report, MITRE Corporation, 1976.

[269] Ravi S. Sandhu, Edward J. Coyne, Hal L. Feinstein, and Charles E. Youman. Role-Based Access Control Models. *Computer*, 29:38–47, February 1996.

[270] C. Neuman, T. Yu, S. Hartman, and K. Raeburn. The Kerberos Network Authentication Service (V5). RFC 4120 (Proposed Standard), July 2005. Updated by RFCs 4537, 5021.

[271] Microsoft. .NET Passport: balances authentication solutions. Technical report, Microsoft, 2002.

[272] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP. RFC 2560 (Proposed Standard), June 1999.

[273] S. Santesson and P. Hallam-Baker. Online Certificate Status Protocol Algorithm Agility. RFC 6277 (Proposed Standard), June 2011.

[274] Javier Lopez, Rolf Oppliger, and Günther Pernul. Authentication and Authorization Infrastructures (AAIs): A Comparative Survey. *Computers & Security*, 23(7):578–590, 2004.

[275] M. Francisca Hinarejos, Jose L. Muñoz, Jordi Forné, and Oscar Esparza. Preon: An efficient cascade revocation mechanism for delegation paths. *Computers & Security*, 29(6):697–711, 2010.

[276] M. Carbonell, J. Torres, D. Suarez, and J.M Sierra. Secure e-payment protocol with new involved entities. In *Collaborative Technologies and Systems*, CTS'08, pages 103–11, 2008.

[277] A. Knauf, G. Hege, and T C. Schmidt. Internet-Draft: A Usage for Shared Resources in RELOAD (ShaRe). draft-knauf-p2psip-share-00 (work in progress), March 2011.

[278] D. Touceda, J.M. Sierra, A. Izquierdo, and H. Schulzrinne. Survey of Attacks and Defenses on P2PSIP Communications. *Communications Surveys and Tutorials, IEEE*, Accepted for Publication, 2011.

[279] D.S. Touceda, J.M.S. Camara, L.J.G. Villalba, and J.T. Marquez. Advantages of identity certificate segregation in P2PSIP systems. *Communications, IET*, 5(6):879–889, April 2011.