Final thesis

Extending IMS specifications based on the charging needs of IPTV

by

Stefan Östergaard

LITH-IDA-EX--06/073--SE

2006-12-07
Final thesis

Extending IMS specifications based on the charging needs of IPTV

by

Stefan Östergaard

LITH-IDA-EX--06/073--SE

Supervisor: Åsa Detterfelt
Attentec AB

Examiner: Arne Jönsson
Dept. of Computer and Information Science
at Linköping University
Abstract

With the standardization of IP Multimedia Subsystem (IMS), the telecommunications scene becomes more and more converged and in the future we will most likely access our services from all kinds of devices and link them together. One important future access method that has so far been left out of the standardization is television. There is a need for Internet Protocol Television (IPTV) to work together with IMS and this thesis focuses on one aspect of that convergence, namely charging.

The problem explored in this thesis is if there is an efficient way of charging for IPTV services while taking advantage of the IMS charging functionality and this is done for two aspects of the problem. First, the possibility of an efficient Session Initiation Protocol (SIP) signaling schema is investigated and then a good charging Application Programming Interface (API) to be used when developing applications is investigated. The findings of these two investigations are then tested and improved during the implementation of a demo application.

This thesis delivers specifications for a signaling schema that enables a Set-Top Box (STB) to pass charging information to an IMS network via INFO requests inside a special charging session. The schema is small and extendable to ensure that it can be modified further on if necessary. The thesis also delivers an encapsulating and intuitive charging API to be used by developers who want to charge for their services.

Keywords: IMS, SIP, UE, INFO request, IPTV, STB, Charging, Services
Acknowledgements

This thesis would not have been what it is without the contributions of others and these individuals deserve to be mentioned and are gratefully thanked.

There are a few persons that have directly effected the content of the thesis such as my supervisor Åsa Detterfelt who I thank for giving me the opportunity to do this thesis and guiding me into the world of consulting. My examiner Arne Jönsson always made me consider the academic values of what I did and Niklas Östh deserves a thank you for his work as opponent.

Henrik Carlsson and Thomas Johannesson provided valuable insights on what would be interesting to look into from the IPTV vendor’s point of view and on what can be done in practice. A big thank you as well to Stefan Ingvarsson for all the help with reSIProcate.

For indirectly influencing the content while sharing their knowledge with me, I would like to thank Hannes Persson, who always provided useful insight during our discussions, and Tobias Gustafsson, who gave me the important first pieces of information on my path to learning IMS and IPTV.

I would like to thank everybody at Attentec for the wonderful spirit and solidarity. A special thank you to Anders Weiland for interesting Monday morning conversations and invigorating floorball practices and to Fredrik Bäckström and Anders Ivarsson for filling my days with humour and my lunches with tasty culinary experiences.

The last and most important thank you goes to my fiancée Jessica Heyman for all of her love, for all her understanding and for always being there for me when I need her.
# Contents

1 Introduction ............................................................................. 1  
1.1 Background ........................................................................ 1  
1.2 Purpose ............................................................................... 2  
1.3 Problem description .............................................................. 2  
1.4 Goal .................................................................................. 3  
1.5 Method ............................................................................... 3  
1.6 Limitations ......................................................................... 3  
1.7 Target audience ................................................................. 3  
1.8 Thesis outline ...................................................................... 4  

I Theoretical background ........................................................... 7  

2 Session Initiation Protocol ....................................................... 9  
2.1 SIP basics ........................................................................... 9  
2.2 SIP network architecture ..................................................... 10  
2.3 SIP messages ...................................................................... 11  
2.3.1 The start line ................................................................... 11  
2.3.2 The header fields ............................................................... 12  
2.3.3 The message body .............................................................. 13  
2.4 SDP .................................................................................. 13  
2.5 SIP extensions .................................................................... 15  

3 IP Multimedia Subsystem ....................................................... 17  
3.1 A first glance at IMS ............................................................ 17  
3.1.1 Vision ........................................................................... 17  
3.1.2 History .......................................................................... 17  
3.1.3 The benefits of IMS ......................................................... 18  
3.2 Protocols used in IMS ......................................................... 20  
3.2.1 SIP/SDP ........................................................................ 20  
3.2.2 Diameter ....................................................................... 20  

vii
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.2.3</td>
<td>RTP</td>
<td>21</td>
</tr>
<tr>
<td>3.2.4</td>
<td>COPS</td>
<td>21</td>
</tr>
<tr>
<td>3.2.5</td>
<td>XCAP</td>
<td>22</td>
</tr>
<tr>
<td>3.2.6</td>
<td>H.248</td>
<td>22</td>
</tr>
<tr>
<td>3.3</td>
<td>IMS architecture</td>
<td>22</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Layers</td>
<td>22</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Functions</td>
<td>24</td>
</tr>
<tr>
<td>3.3.3</td>
<td>Interfaces</td>
<td>27</td>
</tr>
<tr>
<td>3.4</td>
<td>Authentication and authorization</td>
<td>29</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Authentication with AKA</td>
<td>29</td>
</tr>
<tr>
<td>3.5</td>
<td>Charging</td>
<td>30</td>
</tr>
<tr>
<td>3.5.1</td>
<td>Offline charging</td>
<td>30</td>
</tr>
<tr>
<td>3.5.2</td>
<td>Online charging</td>
<td>32</td>
</tr>
<tr>
<td>4</td>
<td>IPTV</td>
<td>33</td>
</tr>
<tr>
<td>4.1</td>
<td>A first glance at IPTV</td>
<td>33</td>
</tr>
<tr>
<td>4.1.1</td>
<td>Defining IPTV</td>
<td>33</td>
</tr>
<tr>
<td>4.1.2</td>
<td>Possibilities and obstacles to overcome</td>
<td>34</td>
</tr>
<tr>
<td>4.2</td>
<td>Protocols used in IPTV</td>
<td>36</td>
</tr>
<tr>
<td>4.2.1</td>
<td>RTSP</td>
<td>36</td>
</tr>
<tr>
<td>4.2.2</td>
<td>IGMP/MLD</td>
<td>36</td>
</tr>
<tr>
<td>4.2.3</td>
<td>PIM</td>
<td>36</td>
</tr>
<tr>
<td>4.3</td>
<td>IPTV architecture</td>
<td>37</td>
</tr>
<tr>
<td>4.3.1</td>
<td>The different ways to cast information</td>
<td>37</td>
</tr>
<tr>
<td>4.3.2</td>
<td>Architecture</td>
<td>37</td>
</tr>
<tr>
<td>5</td>
<td>Analysis</td>
<td>41</td>
</tr>
<tr>
<td>II</td>
<td>Investigations</td>
<td>43</td>
</tr>
<tr>
<td>6</td>
<td>SIP signaling schema</td>
<td>45</td>
</tr>
<tr>
<td>6.1</td>
<td>Definition of the objective</td>
<td>45</td>
</tr>
<tr>
<td>6.2</td>
<td>The road towards the signaling schema</td>
<td>46</td>
</tr>
<tr>
<td>6.2.1</td>
<td>Registering the STB</td>
<td>46</td>
</tr>
<tr>
<td>6.2.2</td>
<td>Sending the information</td>
<td>46</td>
</tr>
<tr>
<td>6.2.3</td>
<td>A new content syntax</td>
<td>48</td>
</tr>
<tr>
<td>6.2.4</td>
<td>Information states</td>
<td>48</td>
</tr>
<tr>
<td>6.2.5</td>
<td>The information to send</td>
<td>49</td>
</tr>
<tr>
<td>6.2.6</td>
<td>Extensions to the schema</td>
<td>50</td>
</tr>
<tr>
<td>6.2.7</td>
<td>Summarizing the schema</td>
<td>51</td>
</tr>
</tbody>
</table>
CONTENTS

7 Charging API
   7.1 Properties .................................. 53
   7.2 Designing the API .............................. 54
      7.2.1 Asynchronous calls ....................... 54
      7.2.2 Registering the STB ..................... 54
      7.2.3 Registering the application .......... 55
      7.2.4 Sending charging information .......... 55
      7.2.5 Summarizing the API .................. 56

8 Use case ........................................ 57

9 Demo application .............................. 63
   9.1 The IMS network ............................. 63
      9.1.1 The simulation architecture .......... 63
      9.1.2 Server design ........................... 65
      9.1.3 Choosing the software ................. 66
      9.1.4 Experiences from the implementation 68
   9.2 The Set-Top Box ............................. 68
      9.2.1 Client design ............................ 68
      9.2.2 Experiences from the implementation .. 69

III Conclusions .................................. 71

10 Results and discussion ....................... 73
   10.1 Results .................................. 73
   10.2 Discussion ................................. 73
      10.2.1 SIP signaling schema .................. 73
      10.2.2 Charging API ........................... 74

11 For further studies ......................... 75
   11.1 STB authentication and authorization .... 75
   11.2 Protecting the content .................... 75
   11.3 Detecting blocked messages ............... 76
   11.4 Protection against charging frenzy ....... 76
   11.5 Finally, last minute thoughts .......... 76

IV Appendices .................................... 77

A Security ........................................ 79
   A.1 IPsec .................................. 79
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.1.1</td>
<td>Security Associations</td>
<td>79</td>
</tr>
<tr>
<td>A.1.2</td>
<td>IKE</td>
<td>80</td>
</tr>
<tr>
<td>A.1.3</td>
<td>ESP</td>
<td>80</td>
</tr>
<tr>
<td>A.2</td>
<td>SIP Security Mechanism Agreement</td>
<td>80</td>
</tr>
<tr>
<td>A.3</td>
<td>Interoperator security</td>
<td>81</td>
</tr>
<tr>
<td>B</td>
<td>IMS Examples</td>
<td>83</td>
</tr>
<tr>
<td>B.1</td>
<td>Registration examples</td>
<td>83</td>
</tr>
<tr>
<td>B.1.1</td>
<td>Setting up security</td>
<td>85</td>
</tr>
<tr>
<td>B.1.2</td>
<td>Subscription to registration state</td>
<td>87</td>
</tr>
<tr>
<td>B.1.3</td>
<td>Deregistration</td>
<td>87</td>
</tr>
<tr>
<td>B.2</td>
<td>Session initiation examples</td>
<td>87</td>
</tr>
<tr>
<td>B.2.1</td>
<td>Media negotiation</td>
<td>89</td>
</tr>
<tr>
<td>B.3</td>
<td>Terminating a session</td>
<td>91</td>
</tr>
<tr>
<td>C</td>
<td>SIP signaling schema</td>
<td>93</td>
</tr>
<tr>
<td>C.1</td>
<td>General description</td>
<td>93</td>
</tr>
<tr>
<td>C.2</td>
<td>Definitions</td>
<td>94</td>
</tr>
<tr>
<td>C.3</td>
<td>Content of messages</td>
<td>94</td>
</tr>
<tr>
<td>C.3.1</td>
<td>Content of REGISTER requests</td>
<td>94</td>
</tr>
<tr>
<td>C.3.2</td>
<td>Content of INVITE and BYE requests</td>
<td>94</td>
</tr>
<tr>
<td>C.3.3</td>
<td>Content of INFO requests</td>
<td>95</td>
</tr>
<tr>
<td>C.4</td>
<td>URIs</td>
<td>96</td>
</tr>
<tr>
<td>C.5</td>
<td>SIP signaling</td>
<td>97</td>
</tr>
<tr>
<td>C.5.1</td>
<td>Registration</td>
<td>97</td>
</tr>
<tr>
<td>C.5.2</td>
<td>Charging</td>
<td>97</td>
</tr>
<tr>
<td>C.6</td>
<td>Possible error responses</td>
<td>104</td>
</tr>
<tr>
<td>D</td>
<td>Charging API</td>
<td>107</td>
</tr>
<tr>
<td>D.1</td>
<td>ICharging.h</td>
<td>107</td>
</tr>
<tr>
<td>D.2</td>
<td>IChargingObserver.h</td>
<td>109</td>
</tr>
<tr>
<td></td>
<td>Acronyms</td>
<td>111</td>
</tr>
<tr>
<td></td>
<td>Bibliography</td>
<td>115</td>
</tr>
<tr>
<td></td>
<td>Index</td>
<td>119</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

This chapter gives information about this thesis concerning questions such as why and how it was done.

1.1 Background

The world of communication is growing fast. This puts great pressure on the players to innovate and to be able to collect ideas from adjacent areas. This has resulted in what is today called Triple Play and Quad Play. Triple Play refers to offering telephony, TV and data services as a bundle and Quad Play also includes mobile telephony. Many operators have seen the benefit of offering services that cover all of their customers’ needs and are doing the best they can to bundle their services together.

There is of course always a downside and in this case it is that customers expect a price cut if they decide to use a bundled service. This lowers the Average Return Per User (ARPU) for the operators and cut profits dramatically. To counteract this, operators innovate new services to deliver and of course charge for to their users. This service-oriented way of thinking is supported by one of the largest standardization projects today the 3rd Generation Partnership Project (3GPP). 3GPP is the organization responsible for developing the new 3rd Generation (3G) mobile system called Universal Mobile Telecommunications System (UMTS) and one part of UMTS is IP Multimedia Subsystem (IMS). IMS is a platform built on the Internet Protocol (IP) and the primary use is to route calls, but it is also a well-developed platform for introducing new services rapidly. Being a service platform makes it suited for a converged network where all traffic is sent over IP.

One of the services that could be sent through IMS is TV. The concept Internet Protocol Television (IPTV) exists today, but it is not standardized
and there several different ways to deliver it. This means that there is a need for among other things a standardized method of charging for IPTV services. It is also important to point out that IPTV today does not use IMS to set up multimedia streams and will probably not do so for a few years. Some kind of cutover solution will be needed if IPTV is to be integrated with IMS.

Several operators in Sweden offer IPTV today through their broadband and they all compete with different technical solutions. One common denominator between all solutions is that they use a Set-Top Box (STB) to decrypt and decode the signals at the user end.

1.2 Purpose

With the background laid out we can begin to discuss the purpose of this thesis. Among IPTV vendors there is a growing interest in SIP and IMS with all of its benefits as a service platform. Especially the parts concerning charging are of interest since it might be possible to use them to provide an easy way of charging for the services provided by applications developed for the software in STBs. The applications built are high-level which means that the charging functionality must also be high-level to fit in. The purpose of this thesis is to investigate how STBs can use SIP/IMS in an effective and easy-to-use way to be able to offer charging as a part of the software platform. Lastly, it is important to remember that IMS is something new and not many implementations exist. Therefore the solution needs to work with both a full IMS implementation as well as a smaller tailor-made implementation as a temporary solution until a full IMS is deemed necessary.

1.3 Problem description

The problem can be expressed as:

- Can SIP/IMS be used to charge for IPTV services in an efficient way?

With the help of the purpose described in Section 1.2 it can be broken down into two different aspects:

- What is the most efficient way of sending charging information with SIP from the STB to an IMS network?

- What functions are needed in a charging Application Programming Interface (API) for it to be efficient and easy to use for a developer?
These two aspects need further defining of the word efficient. For the first aspect this means that the signaling should first of all be reliable and secondly not flood the network with constant signaling. As for the second aspect the focus should be to provide an API that enables easy and intuitive developing of applications using the charging functionality provided by the API.

1.4 Goal

This thesis should present a well investigated solution that enables IPTV vendors to incorporate charging functionality in STB software.

1.5 Method

To solve the problem, two investigations will be needed, one for each of the aspects in Section 1.3. Each of these two investigations will have two steps:

1. Identify and investigate different possible existing and new solutions.

2. Choose the most suitable solution with a clear motivation.

After the solutions have been chosen a demo application will be implemented that tests the solution as a whole. The demo application will consist of a server part that simulates an IMS network and a client part that is an STB with a test application that uses the charging API to send information to the simulated network. This testing is done to draw practical experiences from using the solution and make adjustments according to those experiences.

1.6 Limitations

This thesis focuses on using IMS to enable charging functionality for IPTV services. That might seem like a small enough area at first, but several side issues arise along the way. This thesis will not handle those side issues, but focus on the main problem description. It means that security issues like authentication, authorization, information protection and protection against misuse will not be considered.

1.7 Target audience

This thesis is intended to be read by researchers and developers, who have basic knowledge about IMS and/or IPTV and are interested in combining
the two technologies. Some technical background will be presented, but if the reader has no knowledge at all about IMS and IPTV, additional reading might be required to understand all concepts.

1.8 Thesis outline

The outline of the thesis is as follows. Note that there is a list of all acronyms used at the end the thesis.

1. **Introduction** is this chapter and gives detail about the thesis such as background and formulates the problem discription.

**Part I** Theoretical Background

2. **Session Initiation Protocol** is a theoretical introduction to the signaling protocol central to the thesis.

3. **IP Multimedia Subsystem** is a theoretical introduction to IMS and the parts it consists of.

4. **IPTV** is a theoretical introduction to IPTV and the most common technologies used.

5. **Analysis** reasons to see if the problem can be solved with the current functionality within IMS and IPTV.

**Part II** Investigations

7. **SIP signaling schema** describes the process of inventing the new SIP signaling schema.

8. **Charging API** describes the process of designing the new charging API.

9. **Use case** shows how all parts interact with each other.

10. **Demo application** describes how the demo application was implemented and the experiences from it.

**Part III** Conclusions

11. **Discussion** weighs pros and cons with the results of the investigations.

12. **Results** gives the answers to the problem description.

13. **Further studies** describes the identified areas that need further study.
Part IV Appendices

A. Security explains the security mechanisms in IMS.
B. IMS examples gives examples of common IMS signaling.
C. SIP signaling schema specification is the specification of the invented schema.
D. Charging API specification is a specification of the designed API.
Part I

Theoretical background
Chapter 2

Session Initiation Protocol

Session Initiation Protocol (SIP) is a protocol defined by the Internet Engineering Task Force (IETF). IETF is behind most of the standards used on the Internet and is more or less a collection of different players with interest in standardizing protocols that focuses on evolving the Internet [14]. IETF is behind well-known protocols like Hypertext Transfer Protocol (HTTP), Simple Mail Transfer Protocol (SMTP) and File Transfer Protocol (FTP) [40]. IETFs work is first released in drafts and when the suggestions are stable and thoroughly analyzed they are released as a Request For Comments (RFC).

2.1 SIP basics

SIP is a text-based protocol that uses an request/response model. In its core form, six requests are specified as follows. [27]

REGISTER is used to register contact information

INVITE is used to invite another user to a session

ACK is used to acknowledge that a final response has been received

CANCEL is used to cancel an ongoing INVITE request

BYE is used for terminating sessions

OPTIONS is used for querying

The responses are given in the form of a three-number status code. These status codes are divided into groups according to the first number of the status code. The six different groups are listed below. A SIP transaction is all messages sent between the UAs starting with a request and ending with a
final response. All responses except the provisional responses are non-final, meaning that a response in the range 200-699 terminates a transaction. [27]

100-199 Provisional responses

200-299 Success responses

300-399 Redirection responses

400-499 Client error responses

500-599 Server error responses

600-699 Global failure responses

Every user in a network using SIP needs a Unified Resource Identifier (URI) that identifies the user. A SIP URI or a SIPS URI are most commonly used although any general URI that complies to RFC 2396 (see [21]) can be used. SIPS URI indicates that the user must be contacted using Secure SIP (SIPS) and Transport Layer Security (TLS) in every hop between two nodes. The SIP URI and SIPS URI has the following basic form. [27]

```
sip:name@domain
sips:name@domain
```

Port number can be added after a colon and parameters can be added at the end separated by semi-colons.

```
sip:name@domain:1234;transport=udp
```

### 2.2 SIP network architecture

The core SIP standard includes definitions of the logical entities that build up the architecture. These are defined as follows. [27]

A User Agent (UA) are defined as the two endpoints in the communication. A User Agent Client (UAC) is the entity generating a request while a User Agent Server (UAS) is the recipient of the request and thus generates the responses. The terms UAC and UAS are only applicable while a request is being processed. At all other times they are both considered just UAs.

A Registrar is a SIP server that receives REGISTER requests and stores information received in those requests, thus keeping track of registered users and the information they have provided.
Proxy servers are used to forward messages. While doing so they assume the role of a UAS against the real UAC and the role of a UAC against the real UAS in the sense that it receives and generates both requests and responses as a UAS and a UAC respectively. A proxy can be either stateful maintaining a server and client state machine or stateless. The main purpose of the proxy is routing, but they could perform other tasks like enforcing policies as well. All proxy servers interpret and if necessary rewrite messages.

Redirect servers generate redirection responses (300-399) to requests they receive.

2.3 SIP messages

Every SIP message has the same structure. It begins with a start line that describes if it is a request or a response. The start line is followed by a number of lines called header fields. An empty line separates the header fields from the message body. The message body is optional, but the rest must be present (including the empty line). The structure is shown below. [27]

Start line
Header fields
Message body

2.3.1 The start line

The start line is different depending on whether it is a request or a response. A typical request start line follows.

INVITE sip:name@domain SIP/2.0

First we have the request type, in this case INVITE. After that comes the request-URI which is the address that the request is intended for and last the version of SIP that should always be 2.0 since this is the current standard. A typical response start line follows.

SIP/2.0 200 OK

Here the SIP version (always 2.0) comes first followed by a response status code. The textual version of the response code does not have to be interpreted and can thus be anything, but usually it is the corresponding name of the response to simplify human reading. [27]
2.3.2 The header fields

The format of the header fields are always the same format.

\texttt{field-name: field-value *(;parameter-name=parameter-value)}

The field-names are always case-insensitive and unless otherwise stated so are field-values, parameter-names and parameter-values, with the exception of values inside quotes. The number of possible parameters is specified for each header field. The six mandatory header fields that every SIP message must include are explained briefly below.\[27]\n
\textbf{To} contains the logical identity of the intended recipient. This is most often a SIP URI, but could be any URI that is supported by the system. The specification also allows for a display name. A simple example of a \textit{To} header field is

\texttt{To: Real Name <sip:name@receivingdomain>}

\textbf{From} contains the logical identity of the sender. As in \textit{To} this may be a SIP URI, but if a call in IMS originated in the Public Switched Telephone Network (PSTN) this would be a TEL URL\[1]. A simple example of a \textit{From} header field is

\texttt{From: Real Name <sip:name@sendingdomain>}

\textbf{Call-ID} serves as a unique identifier and it must remain the same in all requests and responses sent during a dialog\[2] between two UAs. It is however recommended that a UA always uses the same \textit{Call-ID}. \textit{Call-IDs} are compared byte-by-byte and a simple example is

\texttt{Call-ID: jdhGkBf7806HN64g4}

\textbf{CSeq} is a field that maintains order among transactions between two UAs. It consists of a number that indicates which message in a sequence the current is and the request name. This makes matching of response to request easy. As before, a simple example is

\texttt{CSeq: 36 INVITE}

\[1\text{See }[25]\text{ for more info on the TEL URL.}\n
\[2\text{A dialog is here defined as all messages sent from an INVITE to a BYE inclusive.}\]
Max-Forwards indicates how many times a message can be forwarded before it is deleted. Every entity that forwards the message deducts one from this number until it reaches zero. This ensures that messages that can not find their recipient will not circulate forever. An example is

Max-Forwards: 50

Via is used for routing responses back the same way as the request came. Each proxy that handles the request and forwards it adds another Via header with the necessary information such as IP address, port number, etc for finding the proxy again. A simple example is

Via: SIP/2.0/UDP [1080:3:24::45:5353]:5060;branch=jf53n5H

2.3.3 The message body

The message body is not defined in SIP, since SIP does not care about what kind of session it is setting up. Therefore you need to use some other protocol to specify the nature of the session. It is very common that the description of the session is done by using SDP (Session Description Protocol). More on this protocol in Section 2.4 [27]

2.4 SDP

Session Description Protocol (SDP) is a way to describe session content that is defined in RFC 2327 [20]. It is made up of lines where each line is a single letter describing the type followed by an equals sign and then the content of the line. The SDP is made up of a session-level part and a media-level part. If we look at a simple SDP example we can identify and describe the basic parts. [20]

IPv6 is used in the example (inside the brackets) as this IP version is to be used in IMS.
\[v=0\]
\[o=\text{username 1234 67890 IN IP4 192.168.1.1}\]
\[s=\text{An example session}\]
\[c=\text{IN IP4 192.168.1.1}\]
\[t=0 0\]
\[m=\text{audio 1024 RTP/AVP 0}\]
\[a=\text{sendrecv}\]

\[v=0\] gives the SDP version used.

\[o=\text{user 12345 6789 IN IP4 192.168.1.1}\] informs about the session’s origin and consists of a username (username), a session id (12345), a version number (6789), a network type (IN, meaning Internet), an address type (IP4, meaning IPv4) and finally an address for the machine initiating the session (192.168.1.1).

\[s=\text{An example session}\] is the name of the session.

\[c=\text{IN IP4 192.168.1.1}\] specifies the connection information of the sender’s end of the session and consists of network type (IN meaning Internet), address type (IP4 meaning IPv4) and finally an address for the machine that is going to be on the sender’s side of the media stream (192.168.1.1).

\[t=0 0\] specifies the start and end times of the session. In this case the session starts immediately and has no specified end time.

\[m=\text{audio 1024 RTP/AVP 0}\] is the first line of the media-level part and describes a media stream. It consists of a media type (audio), a port number (1024), a transport protocol (RTP/AVP, meaning Realtime Transport Protocol with the Audio/Video Profile) and the payload type (0 meaning u-law PCM-coded single-channel audio sampled at 8 kHz)\(^4\)

\[a=\text{sendrecv}\] Every media stream can have an optional number of attributes and this particular one says that the stream should be two-way.

This is SDP in one of its simplest forms. It can handle more session-level types and any number of media streams.

\(^4\)RTP/AVP and different payloads are described in section 3.2.3
2.5 SIP extensions

Since SIP was included in IMS, several extensions has been added to the core specification. Each extension adds functionality and often new types of requests to SIP. A few important extensions are described below.

RFC 2976 - The SIP INFO Method This RFC defines the new method INFO which can be used inside a session to send any information that does not affect the session itself. [26]

RFC 3262 - Reliability of provisional responses In core SIP, no acknowledgments are sent for provisional responses. If the underlying network uses an unreliable transport protocol like User Datagram Protocol (UDP), the response might not reach the UAC and the UAC will think that its request never arrived to the UAS and will retransmit. To avoid having multiple requests using up the bandwidth, RFC 3262 introduces the request PRACK which is to be sent whenever a provisional response\(^5\) is received. PRACK is to be treated as any other non-INVITE method and should be answered with a 200 (OK) response if correctly received. Examples on how PRACK is used in IMS can be found is Appendix B.2. [28]

RFC 3265 - Session Initiation Protocol (SIP)-Specific Event Notification The need for event notification is not hard to imagine and RFC 3265 defines two new requests SUBSCRIBE and NOTIFY and new headers Allow-Events, Event and Subscription-State that make event notification possible. IMS uses events to subscribe to registration states which is illustrated in Appendix B.1.2. [29]

RFC 3312 - Integration of Resource Management and Session Initiation Protocol (SIP) This extension is more commonly known as preconditions and it is used during the initiation of a session. Before you start a media session it is important that Quality of Service (QoS) is negotiated to make sure that the session has the resources that it requires. RFC 3312 defines a new option tag for SIP and new attributes for SDP in addition to describing a message flow that should take place to establish QoS before the callee is alerted to avoid unnecessary calls that cannot be set up anyway. [31]

RFC 3323 - A Privacy Mechanism for the Session Initiation Protocol (SIP) If you want your identity protected in SIP, you could

\(^5\) Actually 100 (Trying) can not be sent reliably (eg with an acknowledgement) so this applies only to responses 101-199. See [28] for further info.
provide a anonymous name in the *From* header field. However, your identity could be revealed in many other header fields of a SIP message and a mechanism for removing all that information is provided in RFC 3323. A new header field *Privacy* is introduced and enables a user to indicate whether the user wants to remain anonymous with different privacy types. [32]

**RFC 3325 - Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks** Inside an administrative domain there is a need for authenticating a user. However, there will be to much overhead if every entity that handles a message needs to authenticate the user. RFC 3325 enables a new header field *P-Asserted-Identity* that a proxy can set when it has asserted the identity of the user. Subsequent processing of the message can now rely on this information and does not need to authenticate again. If the user has several different SIP identities in the UA another header field *P-Preferred-Identity* is introduced by which the user can indicate which identity is preferred for this call. Finally a new privacy type *"id"* is defined to be used together with RFC 3323. [33]

**RFC 3329 - Security Mechanism Agreement for the Session Initiation Protocol (SIP)** This extension is used to negotiate a security mechanism to use for secure communication between two entities. How this works is described in Section [A.2] [35]

---

Chapter 3

IP Multimedia Subsystem

3.1 A first glance at IMS

This section is intended to give the reader a quick introduction to why IMS is so interesting in the future telecommunication world. After that we will look at the main protocols used in IMS and later go more in depth on IMS architecture.

3.1.1 Vision

The IMS vision is best described as a telecommunication network that combines the mobility of the cellular networks and all the useful services possible in an IP network such as the Internet. It is done in a system that is fully standardized by open standards. This vision includes as a consequence of using IP and open standards that new services will be easy to develop, fast to deploy and accessible wherever you are in the world. [14]

3.1.2 History

In 1998, the 3GPP standardization organization was founded. The goal of 3GPP is to develop a third generation mobile system that uses Wide-band Code Division Multiple Access (WCDMA), Time Division Code Division Multiple Access (TD-CDMA) and an evolved Global System for Mobile Communication (GSM)[1]core network.[2] The name of this new mobile system from

---

1The original name of the system was "Groupe Spécial Mobile" which was the name of the French group doing the first standardizations. The name was later changed to the current—more global—name.

2There is a sister organization called 3rd Generation Partnership Project 2 (3GPP2) that is developing a similar system called CDMA2000 with the American Code Division
3GPP is UMTS and it is now part of the International Telecommunication Union (ITU) specification IMT-2000\textsuperscript{3} \cite{16}.

3GPP does not do all the work on their own. Several protocols have been standardized by IETF prior to being adopted into IMS and opinions are received from among others Open Mobile Alliance (OMA) which was created in June 2002 to be a forum comprised of vendors, service providers and content providers in the mobile industry promoting interoperable mobile data services with a focus on usability. The papers drafted by 3GPP are released as either a Technical Specification (TS) or a Technical Report (TR) depending on the content. \textsuperscript{16}

The standards from 3GPP are regularly frozen into releases that fulfill some defined milestone. The first release (Release 1999) from 3GPP specifying UMTS did not include IMS, but during the work on the second release (called Release 2000 at the time) IMS was introduced. However, it was soon realized that one year was not enough to complete the release and Release 2000 was split into Release 4 and Release 5\textsuperscript{4}. IMS was included in Release 5 and since 2002 when Release 5 was frozen IMS has been a central part of UMTS. \textsuperscript{16}

\subsection{The benefits of IMS}

There is a lot of talk about IMS. Several operators already have it and the many other are either building it or preparing for it. But what are really the benefits of using IMS?

IMS is a platform upon which services can be built, services that are intended for usage in a packet-switched, mobile domain. There are of course already ways\textsuperscript{5} of doing this so why introduce IMS? Because there are certain things that make IMS a better choice than other existing alternatives. The next subsections describe the most important. \textsuperscript{14}

\begin{footnotesize}
\begin{itemize}
\item Multiple Access (CDMA) systems as starting point. CDMA2000 also includes a version of IMS, but these two different IMSs are not entirely compatible.
\item IMT-2000 (International Mobile Telecommunications-2000) is a global specification by ITU that includes five specifications for 3G of which UMTS and CDMA2000 are the two dominating.
\item There where releases prior to UMTS concerning GSM and EDGE (Enhanced Data rates for Global Evolution) using the same numbering.
\item One way is data transfer using General Packet Radio Service (GPRS) in the current GSM systems
\end{itemize}
\end{footnotesize}
CHAPTER 3. IP MULTIMEDIA SUBSYSTEM

Access transparency

IMS is built on an IP network. This means that IMS is access independent, meaning that you can access it by other means than just using General Packet Radio Service (GPRS) in your mobile. As long as your access method can handle the Internet Protocol you can access the network and thus IMS. Of course your terminal needs to support the higher-level application protocols as well as being able to encode and decode the codecs used in IMS. [14]

There are specifications and requirements of interworking with the circuit-switched domain (e.g. PSTN and GSM) in IMS. This is to be able to communicate with already existing networks. [14]

The vision is that there should exist only one standard of IMS and if different vendors implement IMS according to this standard all IMS network should be able to interwork. Thus anyone should be able to access their services in any other IMS network with a roaming agreement. [14]

Based on sessions

It is easy to set up multimedia sessions between two users and multimedia communications such as voice and video calls are supported by IMS. [14]

One of the objections against IMS is that there is already free Voice over IP (VoIP) applications that run over Internet. However these have no way of guaranteeing any Quality of Service (QoS) since the provider does not control the network itself. Instead they provide a "best effort" service which works fine until the network gets overloaded. In IMS, QoS can be negotiated and guaranteed. This is even something that IMS handles automatically so the creator of a service does not need to think about it. However the operator of the IMS network can control what QoS a certain user can get and thus differentiate customers. [14]

Every session that is set up in IMS goes through a set of functions that control that the user is allowed to access the network and that the session requested is permitted. This gives the operator full control over what services a certain user can access and also enables different charging for each service that the operator offers. [14]

Rapid service introduction

Another advantage of having an open standard of IMS is that is layered. It will be easy for any operator or third-party provider to create new services

---

6 More on these IMS entities in Section 3.3.2
7 More on the IMS layer structure in Section 3.3.1
and deploy them fast. Since the IMS basically handles all signaling, the developer needs only to consider the main functionality of the service. By having a single standard IMS, the developer can rely on the fact that the service will work in any implementation of IMS. [14]

3.2 Protocols used in IMS

3.2.1 SIP/SDP

SIP is the protocol used for session control in IMS. This protocol has already been introduced in Chapter 2 and no further information about the protocol itself will be given here.

3.2.2 Diameter

Diameter is a binary protocol used in IMS to perform Authentication, Authorization and Accounting (AAA) services. It is built on the highly successful protocol Remote Authentication Dial In User Service (RADIUS) used by many dial-up Internet Service Provider (ISP). The base protocol (defined in RFC 3588 [38]) specifies the basic functionality and it can be extended using different applications. Diameter is specified to run over a reliable transport protocol like Transmission Control Protocol (TCP). [14]

The communication is built on a Request/Answer model where each request has a specified answer and the base specification includes seven Request/Answer pairs. Each message contains a header with fields that are required and fixed. Among other things they indicate the Diameter version and the length of the message. The header is followed by a number of Attribute Value Pair (AVP) which convey the information in the message. There are numerous different AVPs and they are built up with a header and a body. The header tells which AVP it is, how long the data part is, etc and the body contains the data associated with the AVP. For a full description of message syntax and AVP syntax the reader is encouraged to read RFC 3588. [38]

There are a couple of Diameter applications specific for IMS. These are not standardized by IETF, but are instead described by 3GPP since they are used in specific interfaces. We however, only note that these specific applications exist and do not describe them further. [14]
3.2.3 RTP

For the media sessions in IMS the Real Time Protocol (RTP), specified in RFC 3550 [37], is used which is a protocol specifically designed to transport real-time data. RTP uses an unreliable transport protocol like UDP to send out packets of data. Each packet contains a timestamp to allow the receiver to sort the packets to be able to play them in the right order. The receiver will start to play the packets shortly after the first arrives, but packets may be very late or even dropped entirely. This means that the receiver needs to find a good offset after the first packet is received before starting to play and also needs to interpolate over packets that do not make it in time. There are codecs that are designed with enough redundancy to allow that a certain amount of data is lost on the way and still keep a good playback quality. [14; 16]

Audio/Video Profile

To use RTP, a profile has to be specified. The profile used in IMS is the Audio/Video Profile (AVP) which is specified with audio and video in mind. It specifies several static payload types that each correspond to a certain codec and dynamic payload types that need to be defined with a codec when used. [14; 16]

RTCP

RTP is always used together with a protocol called RTP Control Protocol (RTCP) RFC 3550 [37] which provides several important functions. First of all it enables both sender and receivers to send reports on the ratio of packets received. This enables the sender to switch to a codec with more redundancy if too many packets are dropped. The second function is to provide a system-wide clock that all RTP timestamps can be related to so that all streams can be synchronized when played. Finally RTCP provides a mapping between the real name of the sender and the binary id that accompanies every packet. This is especially useful in audio/video conferences. [14; 16]

3.2.4 COPS

The Common Open Policy Service (COPS) protocol is used to control policies. In IMS, its specific use is to send policy requests, answers and updates between a Policy Decision Point (PDP) and a Policy Enforcement Point (PEP). [16]
3.2.5  XCAP

XML Configuration Access Protocol (XCAP) is a protocol based on Extensible Markup Language (XML) and it enables a user to access and update information on an XCAP server. [16]

3.2.6  H.248

The H.248 protocol is also known by the name Media Gateway Control Protocol (MEGACO) and is described in RFC 3525 [36]. It is used to signal between a media controller and a media gateway. [16]

3.3  IMS architecture

This chapter will go more in depth on how IMS works and what the important parts are. Figure 3.1 gives an overview of IMS and the important nodes and the interfaces linking them together. All of these is explained in the next couple of sections.

3.3.1  Layers

As is shown in Figure 3.1, IMS can be divided in three different layers. These layers are however not described in the specification which has led to a number of different versions of them. Camarillo and Garcia-Martin[14] does not discuss layers at all while Poikselkä et all [16] does. I have used the layer description in the book from Camarillo and Garcia-Martin[16] as a starting point and adapted it slightly in a way I feel is more logical. Starting from the top level and down these can be described as

Application layer is where IMS applications and services reside on Application Servers.

Control layer is where all control functionality such as session establishment, QoS reservations and authentication issues take place. Charging partly resides here as well, but is not included in the overview for clarity reasons. Charging is discussed in Section 3.5.

Transport layer is where breakout points to other networks exist and the user’s IP Connectivity Access Network (IP-CAN).
Figure 3.1: An overview of IMS [14; 16]
3.3.2 Functions

In IMS, the nodes are not specified as nodes, they are specified as functions. This means that functions can reside on different machines or be grouped together to save hardware in smaller systems. The possibility to split a function on several machines exist as well and will be used with the purpose to load-balance the system. [2, 14]

User Equipment

User Equipment (UE) is not really a function, but rather a collective name that is used for the user equipment connected to IMS no matter what kind it is. Access to IMS can be achieved in many ways, but since this thesis does not need any other access than an ordinary computer no other access method will be discussed here. [1]

Session Control Functions

Call/Session Control Function (CSCF) is the IMS backbone. All signaling goes through these functions and the IMS would not work without them. There are three different kinds of functions that perform specific tasks. [14]

Proxy-CSCF (P-CSCF) is the user’s contact in IMS. A user accessing an IMS network will only have direct contact with the P-CSCF during signaling. It is assigned to the user during registration and will remain assigned until the user deregisters. The tasks performed by the P-CSCF are many and include to authenticate the user and assert its identity to the rest of the network, check requests for errors and report charging events. [14]

Serving-CSCF (S-CSCF) is at the center of all signaling and it acts as a SIP registrar entity. All requests will pass through a S-CSCF and since the S-CSCF is the central node in IMS, it is connected to several other functions for a variety of tasks. Some of the most important are to handle authorization and registration of subscribers, route requests to the recipient’s home network and route requests to an appropriate application server if necessary. [14]

Interrogating-CSCF (I-CSCF) is a function that the P-CSCF send messages to and particularly message that it doesn’t know the destination of. These are typically INVITE requests without more routing information than a SIP recipient. The P-CSCF sends the messages to an I-CSCF that will, in its turn, ask the Home Subscriber Server (HSS)
which S-CSCF is responsible for the recipient and forward it to that S-CSCF. The I-CSCF essentially performs the task of finding the next hop for the request on the way towards the recipient. [14]

Application servers

An Application Server (AS) is a function that performs a certain task. ASs come in four different kinds. [14]

SIP Proxy mode is an AS that performs some kind of service before the message is routed to the recipient (e.g. keeping a record of all messages sent).

SIP Redirect mode is an AS that responds with redirects to requests destined for a recipient that cannot be reached (e.g. recipient has switched operator).

SIP UA mode is an AS that is acting as one end of the communication. The AS could either be at the terminating end (e.g. a voice mail server) or at the originating end (e.g. a server sending reminders).

SIP B2BUA mode is an AS that sit in between two users acting as terminating UA toward the originating user and vice versa (e.g. a server controlling the signaling between two users that can disconnect when they run out of credit, called Back-to-Back (B2B)).

The SIP AS is the AS where new services for IMS will be built. There are two other ASs, but these are mostly there for backwards compatibility. [14]

User information storage

In any system there is a need for storing information about its users. In IMS, this is performed by one or two functions depending on the size of the network. [14]

The Home Subscriber Server (HSS) is the actual storer of information in a long-term, persistent way. All user profiles are stored in an HSS including subscription data, security data and allocated S-CSCF. The HSS can be accessed by the I-CSCF, the S-CSCF and ASs in the network that need user information. [14]

If the network is so large that one HSS is not enough to store all user profiles, a Subscription Locator Function (SLF) is needed as well. The only

---

8Open Service Access–Service Capability Server (OSA-SCS) and IP Multimedia Service Switching Function (IM-SSF).
function it performs is to keep a mapping between each user and the HSS responsible for keeping this user’s user profile. [14]

Policy functions

By using policies the operator can control media access levels and negotiate reasonable QoS parameters. To do this, two different functions are specified. [14]

The Policy Decision Function (PDF) is where policy decisions are taken. The PDF assumes the role of a PDP. It receives information from the P-CSCF regarding the session initiation in progress and makes decisions based on that information. [14]

The Policy Enforcement Point (PEP) is where the policy is enforced. Since the functionality was first specified for a GPRS access method the PEP resides inside the Gateway GPRS Serving Node (GGSN). However since this is an entity that is specific for this access method it is better to single out the PEP instead. [14]

Media resources

The Media Resource Function (MRF) is, as the name implies, a media source in the network. The media could be prerecorded voice announcements, but the MRF could also perform tasks like mixing streams and transcode streams to a different codec. The MRF is split into two functions where the Media Resource Function Controller (MRFC) is the signaling function while the Media Resource Function Processor (MRFP) is the media function.

Interworking with other IMS networks

Interworking with other IMS networks in the world is a top priority and security is a vital part of this. Because of this, a Security Gateway (SEG) resides at the border of the network and encrypts, signs and verifies all data going between the networks. [14]

Interworking with circuit-switched systems

To be backwards compatible, IMS must be able to interwork with circuit-switched systems like PSTN. A set of functions accomplish this. [14]

The Breakout Gateway Control Function (BGCF) enables routing messages to the circuit-switched domain. It basically selects an appropriate
Media Gateway Control Function (MGCF) through which communication is to take place with the PSTN.

The MGCF is responsible for conversions between SIP in IMS and the high-level signaling protocols used in PSTN as well as controlling resources in the Media GateWay (MGW).

The Signaling GateWay (SGW) acts as a translator between IMS and PSTN for the lower level protocols in the signaling plane.

The MGW acts as a translator as well, but instead of translating signaling protocols, it translates media protocols between the two domains.

**Interworking with IPv4 networks**

Since IMS is designed to be used with IPv6 while most IP networks today use IPv4, IMS needs to be able to translate between them. This is done by using two functions. [2; 14]

The Interconnection Border Control Function (IBCF) is essentially a SIP B2BUA AS. One side connects to the IPv6 IMS network and the other towards the IPv4 external network. It inspects all messages going between the different networks and changes network information such as IP addresses and port numbers so that all media traffic is sent through the Transition GateWay (TrGW).

The TrGW acts like a Network Address Port Translator-Protocol Translator (NAPT-PT) and maps connections on the IPv6 side together with connections on the IPv4 side of the TrGW. This enables media streams to pass between the different networks without problem.

**3.3.3 Interfaces**

In IMS, there are interfaces specified between all functions that need to send messages to each other. These are mainly named by two letters, but although they have a name not all of them are specified. A summary of all these interfaces are given in Table 3.1. [1; 14]
<table>
<thead>
<tr>
<th>Name</th>
<th>Functions</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cx</td>
<td>HSS S/I-CSCF</td>
<td>Diameter</td>
</tr>
<tr>
<td>Dx</td>
<td>SLF S/I-CSCF</td>
<td>Diameter</td>
</tr>
<tr>
<td>Gm</td>
<td>P-CSCF UE via IP-CAN</td>
<td>SIP</td>
</tr>
<tr>
<td>Go</td>
<td>PDF PEP</td>
<td>COPS</td>
</tr>
<tr>
<td>Gq</td>
<td>P-CSCF PDF</td>
<td>Diameter</td>
</tr>
<tr>
<td>ISC</td>
<td>AS S-CSCF</td>
<td>SIP</td>
</tr>
<tr>
<td>Mg</td>
<td>MGCF CSCF</td>
<td>SIP</td>
</tr>
<tr>
<td>Mi</td>
<td>S-CSCF BGCF</td>
<td>SIP</td>
</tr>
<tr>
<td>Mj</td>
<td>BGCF MGCF</td>
<td>SIP</td>
</tr>
<tr>
<td>Mn</td>
<td>MGCF MGW</td>
<td>MEGACO/H.248</td>
</tr>
<tr>
<td>Mp</td>
<td>MRCF MRFP</td>
<td>MEGACO/H.248</td>
</tr>
<tr>
<td>Mr</td>
<td>MRCF S-CSCF</td>
<td>SIP</td>
</tr>
<tr>
<td>Mw</td>
<td>CSCF CSCF</td>
<td>SIP</td>
</tr>
<tr>
<td>Mx</td>
<td>IBCF S/I-CSCF</td>
<td>SIP</td>
</tr>
<tr>
<td>Sh</td>
<td>HSS SIP AS</td>
<td>Diameter</td>
</tr>
<tr>
<td>Ut</td>
<td>AS UE UE via IP-CAN</td>
<td>XCAP</td>
</tr>
<tr>
<td>Za</td>
<td>SEG SEG</td>
<td>SIP with IPsec (authentication, integrity and encryption is mandatory)</td>
</tr>
<tr>
<td>Zb</td>
<td>SEG S-CSCF</td>
<td>SIP with IPsec (authentication, integrity mandatory, encryption optional)</td>
</tr>
</tbody>
</table>

Table 3.1: Interfaces [1; 2; 16]
3.4 Authentication and authorization

Authentication, Authorization and Accounting (AAA) are central in IMS. This section describes the mechanisms used during the thesis. Further information about security in IMS is located in Appendix A.

This section describes the two first parts of AAA issues in IMS. The third part, accounting, is described in Section 3.5, because it is very central to this thesis. All of these three subjects are closely linked together and if you want to perform accounting you need to do authentication and authorization first. However, authentication and authorization are more closely linked together and are more integrated with each other in IMS, and therefore they are handled under a single section here. [14]

Authentication and authorization in the IMS network involves the UE, the I-CSCF, the S-CSCF, the HSS and if it is needed an SLF. In this section it is assumed that the network is small enough to only use one HSS. Adapting the section to a multiple HSS solution would simply involve a set of requests sent to the SLF before they can be sent to the appropriate HSS. [14]

The basic idea in authentication and authorization is that the HSS is the keeper of information and the I-CSCF and the S-CSCF requests this information. The I-CSCF is interested in finding the correct S-CSCF to forward its request to, while the S-CSCF handles the authorization. [14]

3.4.1 Authentication with AKA

When a user tries to register with the network, the user will be authenticated using the 3GPP Authentication and Key Agreement (AKA) which is described in RFC 3310[30]. The algorithm is based on a shared secret between the UE and the network and works as follows. [10; 16]

When the S-CSCF receives a REGISTER request it will fetch the Authentication Vector (AV) from the HSS using the Cx interface. From the AV the S-CSCF gets a random challenge (RAND) based on the shared secret (K) and a sequence number (SQN) and the expected response (XRES) to that challenge as well as a network authentication token (AUTN), an integrity key (IK) and a cipher key (CK). Observe that the S-CSCF never knows the K or the SQN.

The S-CSCF sends the RAND, AUTN, IK and CK in a 401 Unauthorized response back to the UE and the P-CSCF assigned to the UE removes the IK and the CK from the 401. These two will be used to create IPsec SAs that it will expect the UE to use after it has received the 401. The P-CSCF then sends the 401 to the UE without the IK and the CK and using a non-secure connection.
Upon receiving the 401 the UE will verify the AUTN so it can trust the network’s identity and calculate the result (RES) to the received RAND. Before it sends a new REGISTER with the RES it must calculate the IK and the CK from the K and the SQN stored in the unit. The UE uses the SAs that are now established and sends a new REGISTER that includes the RES.

The P-CSCF just proxies the REGISTER on its way and when the S-CSCF gets the new REGISTER it compares the RES to the XRES and if it is a match the user is authenticated.

3.5 Charging

Accounting is a central concept in IMS and is usually named charging within the specifications. Figure 3.2 gives an overview of what charging functions there are and how they are separated in two different kinds of charging online and offline. These two are fundamentally different and will be described separately in the two following sections. In this section only the really IMS specific charging will be discussed. This has the effect that charging information contributed by for example the GPRS will not be discussed.

There are a two functions that are used in both offline and online charging and these are Charging Trigger Function (CTF) which is a collection name for functions that trigger on certain events and send charging events to the charging system. In offline charging, they are S-CSCF, I-CSCF, P-CSCF, MRFC, MGCF, BGCF and AS where online charging only includes S-CSCF, MRFC and AS as CTFS. The other function is the Billing Domain (BD) which is responsible for summarizing all of the Charging Data Record (CDR) and creating the actual bill. [5; 9]

3.5.1 Offline charging

Offline charging is when an invoice is created and sent at the end of the month. Because of this, most of the functions in offline charging are to collecting information that can be transformed into a sum that the user owes the operator. [16]

Functions

In addition to the functions already described, there is one function that contribute to offline charging and that is the Charging Collection Function (CCF) which is responsible for collecting charging information from all the
An overview of charging functions

other involved functions that are visible in the overview in Figure 3.2. From this information it creates a CDR full specification in TS 32.297 [7] that is sent to the BD. [5]

Interfaces

There are two IMS interfaces in use in the offline charging as can be seen in Figure 3.2. The Bi interface is used by the CCF to send CDRs to the BD. In TS 32.297 [7], File Transfer Protocol (FTP) is defined as the transfer protocol to be used in this interface. The other interface is Rf and it is used by the CTFs to send charging information to the CCF. The protocol used there is Diameter. [5]

How it works

The offline charging consists of Charging Data Transfer messages. These messages are triggered by for example SIP messages and are in fact Diameter requests that the different CTFs send to the CCF. There are four types: start, interim, stop and event. With these four types the CTFs can send information to the CCF, which starts to correlate the information it receives. When a record can be finished it will create and send a CDR to the BD which then takes care of summarizing the records and creating a bill to be sent to the user. [5; 9]
3.5.2 Online charging

Online charging is when the service provided depends on some kind of credits in realtime. Typically this is a prepaid account. [16]

Functions

The Online Charging System (OCS) is the central node in online charging and can be compared to the CCF. The OCS can also be broken down in several pieces which are specified in TS 32.296. There is also the IMS Gate-Way Function (IMS-GWF) used as a translator between the ISC and the Ro interfaces. This function can be incorporated in the S-CSCF if the S-CSCF can use Diameter directly. [5; 6]

Interfaces

The Bo interface is used by the OCS to send CDRs to the BD. Just like the Bi interface it uses FTP to transfer the CDRs. The ISC interface is the same interface that the S-CSCF uses to communicate with an AS, but here it is used to communicate with an IMS-GWF instead. Finally the Ro interface is used by CTFs in the same manner as the Rf interface, which sends charging information using the Diameter protocol. In online charging, the recipient is the OCS. [5; 7]

How it works

The online charging can work in three different ways. These are described below. A CDR can be created by the OCS and sent to the BD using the Bo interface. [5]

Immediate Event Charging (IEC) is when credits are granted and immediately removed from the account so that the user is allowed to use a single event-based service.

Event Charging with Unit Reservation (ECUR) is when credits are instead reserved before the event-based service is allowed. After the service is finished, credits are deducted and/or returned to the account depending on how many credits the service actually required.

Session Charging with Unit Reservation (SCUR) is when credits are reserved for a session. Credits can also be additionally reserved during the session if necessary. Unused credits are returned to the account after the session stops.
Chapter 4

IPTV

4.1 A first glance at IPTV

IPTV is happening today. In Hong Kong, the operator PCCW has the world’s largest IPTV subscriber list with 700,000 subscribers and many big companies in Asia, Europe and USA are preparing for the next TV generation. In Sweden it is already possible to get IPTV from some operators (eg. Bredbandsbolaget [13]). [18; 45]

4.1.1 Defining IPTV

Because IPTV is not standardized like IMS, we first have to define what IPTV is. Although the acronym is easily understood, the concept of it is not. This can be read in *IPTV: Still to new to define* by Leslie Ellis [15]. To continue we need a definition and from all the articles about IPTV I have tried to sum it up into a sentence:

IPTV is TV delivered over an IP network and utilizes two-way communication to provide new services together with a more personalized TV experience.

The definition encompasses the key point of IPTV: it is delivered over IP which enables two-way communication (ie. interactivity). The interactivity part is what brings up a whole new range of possible services and by actively telling the TV your preferences, the services can be more personalized than we are used to.

Despite all the possibilities, it is however wise to first create and launch a simpler version of IPTV before experimenting with services that the customer may or may not want to have. Watching TV is and will most likely be the
most important feature of a TV in the foreseeable future and if you cannot provide good quality TV you will not get many customers by offering cool features. [18; 46]

4.1.2 Possibilities and obstacles to overcome

With IPTV, a number of opportunities arise for companies, but they all have a downside that needs to be considered before going after them head over heels.

Converged network

With the upcom of IPTV, there is now a possibility to create a single IP network to take care of all your services. You can offer triple play voice, TV and internet access across a single network which reduces maintenance costs. It is also a good opportunity to collect all services in one single system that handles charging, which would also reduce costs. Other benefits include the ability to blend services together. This can be done either by adding old services to the TV like Instant Messaging (IM) or a telephone service in the TV or by creating new services like caller id on TV and displaying the children’s location on the TV. [42]

There is however a downside to integrating services in a single IP network: the demands for bandwidth will grow fast. Currently around 4 Gbps [12] of capacity would be required from the Video High-Ends\(^1\) to the central node on Friday and Saturday nights in a 10,000-subscriber network and this will grow as the customer base grows. Adding to this is the fact that TV viewers are not that error-friendly. A single one second blackout in the final scene in a movie could be enough to ruin the experience. This puts very high demands on the networks ability to provide a guaranteed QoS and to recover from dropouts very fast. As for the ability to create new services there is no way of telling if they will be a hit or not since the TV has always been a passive medium. [42]

Video on Demand

The most spoken-of new service to in IPTV is Video on Demand (VoD). This can be described as viewing video whenever you want to. It is like renting a movie, without having to leave the house. If this is provided in an easy and comfortable way to the subscriber, there is no reason for the subscriber to go

\(^1\)The high-performance servers that store and send Video on Demand titles to the viewers.
to the store and rent a DVD instead. This is also one of the few real edges
the IPTV companies have towards traditional TV companies.

There is a saying in the TV world that "Content is king" and in the case
of VoD this is very true. If you do not have good content to offer, the masses
of subscribers will go to someone who has. As British Telecom found out, it
is not always easy to acquire the content since the producers of the material
still like to deal with the same persons as before and the IPTV companies
does not have much to offer at the moment. [44]

Digital Video Recorder

A Digital Video Recorder (DVR) or Personal Video Recorder (PVR) is
something similar to VoD, but with one big difference: it only contains what
the user records to it. Today many DVD products contain a harddrive and
can thus act as a DVR, enabling services like simple recording and advanced
time shifting\(^2\). Today the American DVR producer TiVo has a deal with
Yahoo enabling the users to schedule recordings online. This is something the
IPTV companies can take one step further by having the DVRs on a server
in the network so that the user can access the recordings from anywhere,
not just on the TV to which the DVR is hooked in.\(^3\) Since recording is
something used today, subscribers will understand the benefits of DVR and
possibly even demand it from their IPTV subscription. [41]

There are of course not only positive effects with DVRs. They demand
large storage servers and these servers need to be able to search and respond
to commands fast since the customers will not accept a slower system than
what they are used to. The positive side is that storage becomes larger and
cheaper all the time which makes upgrading an affordable business. [41]

Interactive services

With the introduction of two-way communication in the IP network there is a
possibility to create truly interactive services in the TV. These services must
not necessarily be services that make the TV watching interactive. They
could instead be other services like surfing the Internet, playing games, video
conferencing, etc. The problem with these is the same as with converged
services: there is no way of telling which will be popular and that can fur-
thermore vary very much depending on country and region. Examples of

\(^2\)Pausing live TV and resuming later, by recording the stream and watching the
recorded stream while it is simultaneously recorded

\(^3\)It could even explore the possibility that the content saved by one subscriber could
be offered to all the other subscribers via a kind of VoD.
4.2 Protocols used in IPTV

There are a number of protocols that can be used in IPTV and the most common are presented here.

4.2.1 RTSP

Real Time Streaming Protocol (RTSP) RFC 2326 [19] gives the user the ability to control streaming media with commands like play, pause, record, etc. The protocol does not require a specific transport protocol and therefore different IPTV systems use different protocols, among them RTP which is described in Section 3.2.3. RTSP has an obvious application to VoD, which is the area in IPTV where it is used. Its message structure is very similar to SIP as it is a Request/Response protocol.

4.2.2 IGMP/MLD

Internet Group Management Protocol (IGMP) the latest is version 3 specified in RFC 3376 [34] is a protocol used to subscribe to a multicast group. This is useful for media that is not controlled by a single receiving user, but continues to stream until finished. IGMP allows IPTV to use multicast efficiently when transmitting traditional channels. [17; 34]

While IGMP is constructed for an IPv4 network, Multicast Listener Discovery (MLD) the latest is version 2 in RFC 3820[39] is designed for an IPv6 network. The basic functionality is the same as in IGMP. [17; 39]

4.2.3 PIM

Since IGMP/MLD is only used between a device and its router, there is a need for a multicast routing protocol to enable routers in a structure to efficiently route streams. The most widely used of these protocols is Protocol Independent Multicast (PIM). PIM is actually a collection of protocols that are independent of which unicast protocol is used between individual routers. [17]
4.3 IPTV architecture

The IPTV architecture is very much depending on multicast to be successful and bandwidth-efficient. The benefits of multicast will therefore be discussed in this section and then the general IPTV network architecture will be presented. The problem with describing IPTV networks is the lack of standards [11]. This means that every network has its own structure, but the one described here is a general architecture which includes the important parts.

4.3.1 The different ways to cast information

There are basically three different ways to cast information to the users who are interested. These are illustrated in figure 4.1 and described below. [17]

**Unicast** is the traditional way of IP transmission, one sender to one receiver. When the same information is sent to a number of users this is very bandwidth-consuming since every interested user will get its own signal.

**Broadcast** is the way TV is traditionally casted; A transmitter sends the signal to everyone around. This means that even uninterested viewers receive the signal which would be a very bandwidth-wasteful way of transmitting in a IP network.

**Multicast** is the way to go in an IPTV solution. The transmitter sends one signal that will be multiplied when it needs to be branched at a router. This ensures that only one signal will be sent through each connection which saves bandwidth.

4.3.2 Architecture

Figure 4.2 shows a general IPTV architecture. It contains several entities described below.

**The Streaming server** is the server streaming out the current content on the content channels that are being offered.

**A VoD server** contains the selection of movies that are available for VoD access are all stored on this server, which is ready to start streaming when requested.

**The Application server** hosts the additional services provided IM, online gambling, etc are all hosted on various application servers.
Figure 4.1: Casting information. Observe that in this case only UE2 and UE4 are interested in the information.
The **Router** needs to be multicast capable.

A **Local VoD server** is needed in networks where many subscribers use VoD. A local VoD server caches the most popular movies and take a lot of load from the core network. This can also be complemented with a local streaming server.

The **Gateway** sits on the edge of the home network and needs to be multicast compliant. It is responsible for diverting the traffic to the correct equipment. It could also contain functionality to guarantee the bandwidth for the real-time multimedia streams.

The **Set-Top Box (STB)** is responsible for transcoding the digital signals it receives from the network into a signal that the TV can understand and show. This signal used to be analog, but newer TVs can also handle a standardized digital signal.

The big difference between this network and an IP network in general is that the owner of the network is in control and can guarantee a certain QoS for the subscriber by not letting the network get overloaded with traffic and by prioritizing real-time packets. This is very important because the TV viewers are generally very demanding about quality as stated in Section 4.1.2.
Chapter 5

Analysis

With the theoretical background on IMS and IPTV it is time to start analyzing the possibilities of using them together to create a possible solution for the problem at hand.

Let us consider a possible scenario where IMS is only used to provide charging and everything else, such as setting up a the session, is handled by the existing IPTV network architecture. The problem arising from this is that IMS sessions are set up using SIP that traverses at least the P-CSCF and S-CSCF. This means that there are at least two functions that know of the session and report charging information to the CCF or the OCS. If the IPTV handles all of this in other ways no function in IMS will know of the session, except for the STB which acts as a UE. This means that the responsibility falls on the STB to report charging information to the network. It also means that a new way of sending this information has to be invented since the UE is completely outside of the charging architecture in IMS, as it is not a CTF.

From this logical reasoning it is clear that the investigation into the signaling part must be aimed at inventing new functionality that extends the current IMS specification without creating possible problems when used together with the full IMS specification.
Part II

Investigations
Chapter 6

SIP signaling schema

The next thing on the list to investigate is the SIP signaling schema. The purpose is to find the best possible signaling schema. The first task is to define the properties of a good signaling schema and then with that in mind investigate possible solutions. This chapter explains the properties used as guidelines and the choices made during the investigation. It gives insight into the discarded possible solutions and the general idea of the selected schema. The full specification of the signaling schema is given in Appendix C.

6.1 Definition of the objective

How do we define "best possible"? Should it be the simplest or the schema with the most functionality? This choice is not as easy as it might seem at first, but the main objective is clear.

- The signaling schema must not go against the existing standards.

The essence of this demand is clear. The signaling schema created must be compatible with a real IMS network and can not contain elements that can not be handled by the existing standard. This basically limits the options to existing SIP requests as bearers of information. It also demands that the STB has to register with the IMS network in the same manner as any UE to ensure that only authorized subscribers gain access to the network.

- The signaling schema should accommodate for sending information specific to the service provided.

The second demand is also obvious. Without the ability to send specific information, the signaling schema will not prove very useful.

- The signaling schema should make it possible to handle network errors.
This is not as sure as the first two properties and is more of a security issue. We don’t want to have unfinished business hanging around and never being finalized properly.

6.2 The road towards the signaling schema

After setting up the guidelines, it is time to investigate the possible solutions satisfying to these guidelines.

6.2.1 Registering the STB

The first guideline states that to register the STB we have to use the standardized method used in IMS, as described in Section 3.4.1. The problem with this is that registering an STB is not specified in IMS yet and some additional research has to be done into how this should be done in a secure way. However security is not in the scope of this thesis and this is not further contemplated. One final note on registering the STB is that each application will use a public URI as identification within the IMS network and especially towards the AS, therefore all of these application URIs must be implicitly registered when registering the box.

6.2.2 Sending the information

Now the STB has registered and we can start sending information. This will be done by each application and the application has to be authorized in some way. We also need to decide on a suitable bearer of this information. Let us discuss a few possible solutions.

Using event based notification

One possible solution would be to use an event-based schema where the AS subscribes with a SUBSCRIBE to the charging information in the STB and when there is new information the STB sends out a NOTIFY to inform the AS of this. This schema would allow any content to be delivered with the NOTIFY request which is one of the guidelines. It would also allow the AS to respond to the information in the NOTIFY.

There is however one big negative side of the schema and it is that the STB is acting as the slave and the AS as the master. With this schema the STB would only send information if the AS has requested it. Of course it could be argued that the AS will always do that, but keep in mind that there
exists several different applications on the STB and the could be problems with that. Besides, the use of NOTIFY as bearer goes against the concept of events. Events should be used when there is a change in some state and not when the notifier wants to ask for authorization.

Because of these issues the option SUBSCRIBE → NOTIFY is discarded.

Using a special session

One could instead use INVITE to set up a special session with the AS and then use a stream with a specialized protocol. This would indeed provide enough flexibility to ensure that any content can be sent and since we could build the protocol to provide any features we want we could easily handle possible network errors and other problems that could occur. We would also have the benefits of a session that has a clear start and end point with INVITE and BYE respectively which can be used to handle non-finalized data that has not been fully processed at the AS when the session closes.

The possibilities are nearly endless, but is it really necessary to create an entire new stream to handle charging? Probably not. Another aspect against this schema is that it will require upgrades in the current IMS networks so that the functions can identify and allow this new kind of stream. That will take time and cost money and not many operators would be interested.

The idea of a new stream with a new protocol is dismissed, but the idea of setting up a session with INVITE still exists.

MESSAGE requests inside a session

To send information in a session, it is possible to use the request MESSAGE. MESSAGE can take any content and it would not encounter any problems in the IMS network because it is becoming widely used as instant messaging grows in the number of users.

MESSAGE is however intended for short messages sent between to different users chatting with each other and not for sending information from one end to the other. There are no real technical problems with using MESSAGE, but this is not its intended use and it would not look professional to use it.

All in all MESSAGE is a good alternative, but it is not the ideal information bearer.

INFO requests inside a session

There is a more suited alternative that MESSAGE to use: INFO requests. They are designed to send information inside a session and do not affect the
session itself. In this case the session only exists to send INFO requests and the content of them does not affect the session. Of course the content of the INFO requests will affect the services provided, but they are not delivered within the session with the AS.

The possibility to use the BYE request as a final closing point of the charging information exists and provides the ability to handle non-final information.

The only problem we might run into is that the IMS network might not be configured to allow sessions without any streaming media. This could be solved by defining a special content for the INVITE request so that the network can recognize the session as a charging session.

Conclusion

It is clear that INFO is the most suited information bearer of the investigated possibilities and when used inside an INVITE - BYE session error handling can be performed easily.

6.2.3 A new content syntax

We need to invent a syntax for the content sent with the INFO requests. To follow the standard in the best possible way we start in the IMS specifications for charging. There, CDRs are used to send charging information and there is a CDR specific for sending information from the AS. Since the schema’s endpoint in the network is the AS this feels like the most suitable starting point. From this CDR specification, the relevant headers are taken and among those are the header ServiceSpecificData. The necessary charging information that does not have its own header gets an attribute value under ServiceSpecificData header instead, since this is the purpose of the header.

6.2.4 Information states

When deciding how to send information, it is important to make sure the schema can handle services that have gone wrong as well as reserving credits for online charging. Both of these issues are solved with the introduction of a state machine where the charging information has four states.
Requested is when a service first is requested and authorized by the AS.

Charged is when a service is delivered and the charging record is sent to the BD.

Dismissed is when a charging record is deleted because the service was never delivered.

Finalized is the state all charging records that have either been charged or dismissed.

The fourth state is never sent as information, but is more a way of defining what finalized charging information is. Figure 6.1 gives the relationship between the four states. The states are used both in an INFO sent by the STB to inform the AS of the action it wants and by the AS in the response to inform the STB of its decision.

The concept of requesting and reserving credits for prepaid accounts allowed by the states fits well into the concept of ECUR and SCUR \(^1\) in the IMS specification for online charging.

6.2.5 The information to send

For the signaling to be efficient it is important to consider which information is necessary to send. First of all we need to send the state discussed in the previous subsection. It could also be of interest to save timestamps when the service was requested and when the service was delivered. To separate different kinds of services, we introduce a type attribute that indicates whether the service was an event or a session. For the charging to work we need to send the number of units, the type of unit and the price per unit and the

\(^1\)Event/Session Charging with Unit Reservation is described in Section 3.5.2
subscriber probably wants to be able to see a detailed record of the charging so we add an informative text as well. To enable correlation between different INFO request an id is introduced that is unique for each charging record.

We also add restrictions on what information can be changed between different sent INFO requests as listed below.

- The number of units to charge for must never be more that the number of units requested.

- The type of units may not change between INFO requests.

- The price per unit may not change between INFO requests.

All of these restrictions are there to avoid errors from being made when charging. A possible such error that is prevented with these rules is that the subscriber can’t be charged for more credits than was requested and permitted.

6.2.6 Extensions to the schema

The focus has been to create a schema that is light-weigh and easy to use. Sometimes\(^2\) this is not enough and there are a couple of possible extensions that have come up during the work, but were rejected to keep the schema small and flexible as it was set out to be.

When using INVITE-INFO-BYE together with the standardized registration schema already decided upon an opportunity arises. When the AS received the INVITE it knows that the STB is registered with the network and can subscribe with SUBSCRIBE to the registration state of the STB. If the STB deregisters by time-out, the AS will be notified and can do a clean up of non-finalized data.

To further ensure correctness during failure there are two possible ways that could even be combined. First, restrictions can be placed on the STB not to request for services that are too long, but instead chop the request up in smaller part. One example could be to request and charge for five-minute intervals during an online gaming session. The second is to extend the schema with INFO requests going from the AS to ask the STB if it is still delivering. This second way could be especially important during a long event-based service delivery since the AS can assume an active role.

\(^2\) Or probably always…
6.2.7 Summarizing the schema

Now we have a final solution to the SIP signaling schema that can be seen in Appendix C. A few summarizing words are appropriate. The schema has three parts that each fills a role.

- REGISTER is used to authenticate and authorize the STB.

- The INVITE and BYE surround an application’s entire charging information. This provides mechanisms for handling non-finalized data if the application shuts down inappropriately. Together with the registration state of the STB it can also handle network failures.

- The INFO request is used as bearer of charging information. Its content is based on the syntax of the AS CDR specified in IMS.

All of these requests are specified in different RFCs and are well-known which ensures that the solution will work in any IMS network with a dedicated AS.
6.2. THE ROAD TOWARDS THE SIGNALING SCHEMA
Chapter 7

Charging API

It is time to develop a design for the charging API that is to be used by application developers who want to send charging information according to the signaling schema invented. First we need to consider what properties are desirable and then we do the actual design of the API. The resulting interfaces is given in Appendix D as C++ code.

7.1 Properties

To do a good design we have to have predefined properties that the design should fulfill.

- The design should be intuitive and easy to use for the developer implementing charging in an application.

This property does not need motivation. It is pretty obvious that the API should be as easy as it can to use.

- The API should be responsible for registering the STB with the IMS network.

This property comes from the fact that there should be no further adaptations necessary in the STB than implementing the API. The property also leads to the fact that there will be only one API active in the STB to register the STB. That API needs to handle charging information from all applications.

- The API should encapsulate as much information as necessary to ensure compliance with the SIP signaling schema.

This property provides protection against sending misformed information to the AS and thereby avoiding possible errors arising from that.

53
7.2 Designing the API

This section describes the details of the API and the reasoning behind the choices. Since the API is bound by the signaling schema, the design of the API is done in several steps.

7.2.1 Asynchronous calls

We need to split the API into two parts. This is done to uphold the asynchronous call structure of SIP. Because the SIP signaling could potentially take too long to respond because of network problems it is decided to keep the asynchronous call structure all the way down to the application to prevent temporary annoying locks. These locks cause the viewer to think that the entire STB is locked with the effect that the user restarts the STB and charging information is lost. It is also important that the API can handle several calls from different applications and this is better done with the asynchronous design that handles calls faster.

We have two interfaces and call them \texttt{ICharging} and \texttt{IChargingObserver}. \texttt{ICharging} is used by the application to send charging information and \texttt{IChargingObserver} is used to receive the responses. The \texttt{IChargingObserver} consists of only one function \texttt{OnChargingEvent} and should be implemented by the application to handle the responses on the information sent.

The asynchronous call structure has been refined several times during the thesis when different issues have been discovered. In a first stadium, synchronous calls were considered and rejected on the grounds described above. After that the calls to \texttt{ICharging} was changed and an application id is sent with every call to ensure that the charging API can handle several applications at a time without trouble. Finally the call arguments of \texttt{IChargingObserver} has been remade. The first version used SIP-oriented call arguments that returned SIP status codes to the application. Now the API just sends an event id that identifies the event that just occurred. All of these changes will make application creation more intuitive for a developer without SIP knowledge.

7.2.2 Registering the STB

The registering of the STB should be handled by the charging module once when it starts up. This makes it ideal to use the constructor for this purpose. If the charging module is implemented as a service in the STB there will only be one instance of the charging module and this makes the constructor the ideal way of registering the STB since this function will be called once.
CHAPTER 7. CHARGING API

With this decision it is wise to send a deregister using REGISTER with 
\( \text{expires}=0 \) using the deconstructor when the charging module is closed. During the use of the charging module there will of course be need of a timer of some sort to send reregisters, but this is more of an implementational decision.

7.2.3 Registering the application

Now the STB is registered and we need to handle the registration of the STB that sets up the session using INVITE. This needs to be done with a function call and in the spirit of easy-to-use we call the function \( \text{RegisterApplication} \). To send the closing BYE that deregisters the application we create a similar function \( \text{DeregisterApplication} \).

To enable as much encapsulation as needed, we need to consider what arguments to pass along with the function calls. It is not reasonable to believe that we can do without sending anything since the application needs to identify itself. We could store the entire application URI in the application and send it as an argument, but is it necessary? No, most of the URI is shared with all of the other applications and only a small part identifies the application. So we decide we need to send an id that identifies the application. If this is the same id that is used in the application URI to identify the application in the IMS network is up to the implementor to decide. One could imagine having a mapping inside the charging module between the id and the URI if that is deemed suitable.

7.2.4 Sending charging information

Finally we have reached the information sending part. Here we want as few general functions as possible, but they should be easy to use and encapsulate as much information as possible. We could do with only one function, but it would not be easy to use with a lot of arguments and there would be no encapsulation. We need more than one function, but how many do we need?

Is six functions a good alternative? There are two types of services sessions and events being considered in this thesis and with the information states described in the signaling schema we would need three functions request, charge and dismiss for each type. But six functions would be bad for at least two reasons. First it is not general since there could be a need to differentiate even more types of services in the future and this design would not allow that. Secondly it does not encapsulate all the information that it could encapsulate. The type of service does only need to be given when requesting the service. After that it is not allowed to change anyway.
Is three functions the best option? Yes, because if the charging module keeps a local copy of the information sent to the AS only the RequestService function would contain all information about the service needed for the request such as request timestamp, type of service, an information text, number of units, type of unit and price per unit. The ChargeService function would only contain the identifying id, units used (with a check in the charging module that it does not exceed the authorized number), an updated information text (which is allowed to change) and timestamps for start and end of the service delivery that are required to finalize the record. The DismissService function would only need the id as argument to do its work.

### 7.2.5 Summarizing the API

We now have a complete API that handles all the signaling needed and also fulfills the properties established. The API consists of an interface for handling registration and sending information and a second interface for receiving the answers. Figure 7.1 illustrates the relationship between the two interfaces. The design is simple and yet powerful with five functions plus constructor and deconstructor that have a simple-to-understand purpose and encapsulate as much information as possible for error protection.

![Diagram of Charging API](image)

Figure 7.1: An overview of the charging API
Chapter 8

Use case

This chapter presents a typical use case of a successful charging and all of the steps involved.

1. At this point some kind of service loader decides that TCharging is needed and is initialized with the constructor.

2. The CSCF authorizes and registers the STB and then returns OK.
Figure 8.2: The application registers.

3 The application starts and registers with TCharging who sends an INVITE to the AS to start a charging session.

4 Initial Filter Criteria in the CSCF sends the request to the AS.

5 The charging session gets approval and the application is authorized to charge.
6 The user requests a service and the application sends a request towards the AS via TCharging.

7 Initial Filter Criteria in the CSCF sends the request to the AS.

8 The AS approves the request, creates a CDR and sends an OK back to the application.

9 The application delivers the service.
The service has ended and was delivered. The application notifies the AS with a charge message.

Initial Filter Criteria in the CSCF sends the request to the AS.

The AS correlates the information with the previous request and finalizes the CDR.

The billing domain saves the CDR in the database.
Figure 8.5: The application deregisters.

14 The user shuts down the application and TCharging sends a BYE to end the charging session.

15 Initial Filter Criteria in the CSCF sends the request to the AS.

16 Non-finalized CDRs are handled by the AS.
Some kind of service loader decides that no application wants to charge anything and that TCharging is not needed any more.

TCharging sends a REGISTER with \textit{expires} set to zero and is then destroyed.

The STB is deregistered from the IMS network.
Chapter 9

Demo application

This chapter describes the implementation of the demo application that is used to draw practical experiences from the design. These experiences will be used in the discussion.

9.1 The IMS network

First we need to simulate an IMS network. The process includes setting up the simulation architecture, choosing software to use and describing the implementation of the chosen simulation model.

9.1.1 The simulation architecture

Setting up the architecture was a cutting business. Starting with the IMS network architecture described in Section 3.3, functions not relevant where cut away. To further simplify a number of functions were reduced and simplified and only core functionality was kept. A summary of all simplifications made follows and the resulting architecture can be seen in Figure 9.1.

Interworking functions

The entire demo application will be done without leaving the IPTV network. This of course means that we will never need any interworking with other networks so all of those functions were removed. The functions removed are IBCF, TrGW, BGCF, MGCF, SGW, MGW and SEG.
**Media Resource Function**

Since the demo application only attempts to use IMS for charging, all media control will be handled by using the IPTV functionality and no media handling will be done inside IMS. This means that the MRCF and MRFP are removed.

**UE connection**

The connection will be over a normal computer IP network and QoS is not an issue during the work of this thesis so the UE connection can be simplified. The PDF and PEP are removed as well as the IP-CAN functionality. There is still a connection of course, but it will only be a simple cable and this does not need further discussion.

**User databases**

This demo application is small-scale and only one HSS will be needed to keep records of subscribers and their user profiles. Since we only use one HSS the functionality of the SLF is not needed and removed. It is also reasonable to decide that the HSS could be simulated itself, for example by implementing a Structured Query Language (SQL) database.

**Control functions**

The demo application is small-scale as stated before and there is no need to have more than one S-CSCF and the I-CSCF can be removed. The P-
CSCF can easily be incorporated inside the S-CSCF to avoid unnecessary complexity and we then get a general CSCF that will perform the basic functionality of a SIP proxy and SIP registrar.

**Application server**

Since the STB is acting as a UE, it needs to have a terminating end for all requests it will send concerning charging events. This terminating end is naturally the AS. The ideal choice would be to add a CCF and an OCS in conjunction with an AS and this was first considered as an option. However this would add a lot of complexity when these three work together and it was instead decided to use a single AS. This means that the AS probably need to be completely remade to provide access to the IMS charging functions, but since this is a simulation, we focus on a simple architecture.

**Billing Domain**

A BD is needed primarily to support final storage of the CDRs created in the AS and to hold charging profiles for the subscribers. This means that the functionality could be incorporated inside the AS except for some kind of external final storage solution that can be simulated by an SQL database.

### 9.1.2 Server design

So far we have derived the needed functions. Now it is time to do a design of the server software architecture as a template for those who want to implement either a test environment or a small simulated network. This design can be seen in Figure 9.2.

As can be seen there will be a SIP Servlet for each IMS function in the network: one for the CSCF and one for the AS. The CSCF will act as a P-CSCF and S-CSCF. This means that all incoming requests will traverse the CSCF and be checked for a matching initial filter criteria. If one is found (i.e., the request contains charging information), the request will be proxied to the AS where the information will be handled according to the signaling schema in Chapter 6.

There will also be ordinary Java classes as front-ends to the HSS and the BD to encapsulate the SQL connection. This means of course that a modification would be needed to the CSCF and to the AS if the real HSS and BD would be needed for a full-scale network, but since this is only a simulation it is the front-end and not the inner workings that are important.
9.1.3 Choosing the software

A couple of different ASs were considered and rejected before BEA Weblogic SIP Server was chosen. The motivations for this follows below.

**Oracle M2CE**

+ It can work as a SIP Proxy/Registrar with minimal configuration.

+ The server is carrier-grade.

+ Fully compliant with SIP Servlet.

+ Integration with Eclipse Integrated Development Environment (IDE)

+ There is knowledge on M2CE at Attentec so help could be received by coworkers.

- The installation process is tedious since M2CE depends on several other applications.

- A license is required.

- No Diameter support.
SIPMethod AS from Micromethod

+ Easy installation.
+ Easy development with specialized IDE.
+ Fully compliant with SIP Servlet.

- A license is required.
- No Diameter support.

Jiplet

+ Free open source.

- No specialized IDE.
- Requires additional software.
- Not compliant with SIP Servlet.
- No Diameter support.

BEA Weblogic SIP Server 2.2

Finally the choice fell on BEA Weblogic SIP Server 2.2, primarily because it is a full-scale application server fully compliant with SIP Servlet. BEA also has the generosity to offer a six-month trial which is perfect for doing a thesis. The list of advantages and disadvantages are

+ Six-month trial available.
+ Fully compliant with SIP Servlet.
+ Development can be done with small modifications to Eclipse IDE.
+ Extensive documentation.

- No built-in SIP Proxy/Registrar.

- No Cx Application Diameter support.

As can be seen in the list BEA’s SIP Server does not have a built-in SIP Proxy/Registrar so a separate CSCF has to be created to accommodate for that need.
MySQL

The HSS will be constructed in a MySQL database and communication will be through SQL. There will be an additional database in MySQL for the BD accessible for the AS via SQL as well.

9.1.4 Experiences from the implementation

The server is implemented according to the design in Section 9.1 and only handles the four requests REGISTER, INVITE, BYE and INFO that are needed in the test application. It is a Java-based application server with the two SIP Servlets for the CSCF and the AS. The implementation of the server was a straight-forward business and no problems with the design were found. No IMS compliance problems with the SIP signaling were discovered either during the implementation of the test application.

9.2 The Set-Top Box

The client is implemented in C++ and follows the API set up in Chapter 7. Since no design or software choices were made during the implementation this part is briefly described.

9.2.1 Client design

The part of the client that contains the charging module consists of one class named TCharging that implements the interface ICharging. TCharging acts as a front end to a SIP stack called reSIProcate. ReSIProcate is an open source project that among other SIP projects develops a SIP stack in C++ which is fully compliant with the RFCs. The SIP stack has been precompiled against the STB software, which simplifies implementation significantly and this stack was choosen without further investigation.

There is also a test application that is just an application that enables streaming media to be viewed. It consists of several classes that fulfill different necessary parts, but the TMainWinow class is the class that makes charging calls to TCharging. It is also the class receiving the answers since it implements the IChargingObserver interface.
9.2.2 Experiences from the implementation

If the implementation of the server was trouble-free, this can not be said for the client. Several redesigns of the charging API was done before the final solution given in Chapter 7 was determined. All of the changes were to the call structure of the two interfaces `ICharging` and `IChargingObserver` to make application development more intuitive and to encapsulate the SIP signaling as much as possible. With the final design, the developer does not necessarily need to know that it is using SIP to send charging information.

The first choice was to use asynchronous calls and to have an observer class listening for responses from the charging API. The cause for this was discussed in Section 7.2.1 and the main reason was latency when signaling over a network.

After that, the arguments of the calls both to `ICharging` and `IChargingObserver` were changed to make the API feel more application-oriented and less SIP-oriented. These changes are described more in section 7.2.1.

Finally, it can be concluded that implementing a test application was very useful as several small issues with the API design were found and corrected, which amounted to a better design.
Part III

Conclusions
Chapter 10

Results and discussion

This chapter presents the results of the thesis and discusses them.

10.1 Results

The problem this thesis tries to answer is whether or not there is an efficient way to use IMS to charge for IPTV services. That question is answered in the affirmative. This thesis provides its results in the form of a signaling schema in Appendix C and a design for a charging API in Appendix D. These are the results of a thorough investigation and have also been tested during the implementation of a demo application and improved according to the experiences from the demo application implementation.

10.2 Discussion

We have now come to the point when it is time to discuss the solution proposed in the investigation.

10.2.1 SIP signaling schema

If we start by looking at the signaling, the problem described was to find an efficient signaling schema that enables sending charging information. Such a schema has been devised through a thorough investigation and it is efficient in several ways. First of all, it does not flood the network with messages. Instead it only takes seven requests to register the STB, invite the AS to a charging session, send the information necessary and finally close the session and deregiser. Seven messages are not many and the number of messages per charged service shrinks if the application charges for several services during
a session. The schema is also small and flexible and can be extended with several fail-safe mechanisms for detecting network or STB problems from the server side. The STB registration state can be monitored by the AS by subscription. To provide fail-safes against service delivery failure it is possible to place restrictions within the signaling schema on how much time can pass between a charging request and a finalization of the CDR as well as to extend the schema with INFO requests going from the AS basically asking the STB if it is still delivering the service. The final conclusion is that the schema presented by this thesis can be considered efficient, flexible and extendable.

Let us now turn the discussion towards IMS specifications and see if the schema can be used without compliance problems with existing parts. The schema consists of only well-known requests REGISTER, INVITE, BYE and INFO which will be recognised by the functions in IMS. This is a first and very important step towards integration into IMS. There are however some possible problems that could arise when using the schema. For the registration of the STB, authentication and authorization might be difficult since there is no ISIM card in the STB. It is not hard to do authentication and authorization per se, but it needs to be secure enough. Since security is not within the scope of this thesis this issue is not further dwelled upon. As for the charging session, the main issue to consider whether the current network accepts sessions that do not contain any multimedia. One solution to this problem is to use a specially defined SDP content describing the charging session. To summarize the compliance, there are some issues, but they can be solved.

### 10.2.2 Charging API

If we now instead concentrate on the charging API provided by the thesis, we can discuss how well it handles the signaling schema and if it meets the demands of being easy to use by the developer. The most important characteristic of the API is that it should encapsulate information so that the developer can make less mistakes, but not so that the developer gets handicapped. The API has been both investigated and tested during the implementation of a demo application to ensure the right mixture of encapsulation and power towards the developer. The API is based on asynchronous calls to handle the latency effects of sending SIP messages over a network and this also accommodates for the possibility of serveral applications using the same charging module as calls will be fast.
Chapter 11

For further studies

Before the results of this thesis can be used in a customer network, there are some aspects that need further investigation. Many of these aspects concern either security for a third party or security against mistakes. Finally last-minute thoughts are briefly mentioned at the end.

11.1 STB authentication and authorization

With the scheme in this thesis, the STB is required to register with the network, but there are no directions on how to do this. Although Release 7 from 3GPP will include a new access method, this is not yet fully specified and a method for authentication and authorization might have to be invented and thoroughly analyzed for security holes.

11.2 Protecting the content

In this thesis, all content is sent in plain text which of course cannot be accepted in a customer network. With the current solution it is possible to insert a server between the STB and the IMS network that reads and/or changes all outgoing messages. This could be used to charge some other customer’s account or change the amount to charge. One solution that will probably cover most needs is using secure SIP with TLS to encrypt the messages.

To provide security against changing messages, some kind of signature should be investigated and to protect the content from inappropriate viewing, the content or the entire message should be encrypted. It should also be investigated how thorough this encryption and signature should be since it will take time to perform.
11.3 Detecting blocked messages

A firewall could be installed in between the STB and the network, which would allow REGISTER and INVITE to be sent, but would block and respond to all INFO messages. The application could then believe that it has been authorized to deliver a service it has not which of course is not desirable.

11.4 Protection against charging frenzy

As long as all applications are developed by trusted suppliers no problems should occur, but in the future it is likely that untrusted third-party suppliers will be developing more applications and also applications accessible through a browser. Then concerns should arise around those applications charging through this system. There are no checking for authenticity of the charging messages as long as the subscriber is allowed to use the application. This means that the application can charge any amount without the user getting any service.

Perhaps this could be solved partly with demands on the delivering server to report information on what services are used, but this needs further study.

11.5 Finally, last minute thoughts

During the last couple of weeks of the thesis I have received input from others that gave some interesting points and ideas. This input came to late to effect the results of the thesis and I will therefore only briefly mention them here.

The use of INFO as information bearer could mean some trouble in some IMS networks since not all networks allow UEs to send INFO requests. The solution to this problem is a request not investigated during the work of the thesis, namely UPDATE. If UPDATE is to be used the results of the thesis would still be the same and all usages of INFO would just be switched to UPDATE. The idea of using SUBSCRIBE and NOTIFY could also be considered as an option despite the argumentation against it in Section 6.2.2, since the AS would have more control.

A design issue to consider with the charging API is whether to have one function for each possible event. This could make the design easier to work with as a developer, but it will lose in generality. Since the two possible designs have different properties and consequences it would be wise to consider both of them when choosing a design.
Part IV

Appendices
Appendix A

Security

This appendix is dedicated to the security mechanisms in IMS not central to the thesis.

A.1 IPsec

Internet Protocol security (IPsec) is a protocol for securing IP packets, specified in RFC 2401 [22]. IPsec can provide security like message confidentiality and message integrity. This is done using a few components of which the most important are described in the subsections below. [16]

A.1.1 Security Associations

A Security Association (SA) defined in RFC 2401 [22] is set up between two entities that want some kind of security between them. The SA are one-way traffic only and if two-way traffic is needed two SAs have to be set up. The SAs are comprised of the three important parts described below. [16]

Security Parameter Index (SPI) is used to identify the SA.

Destination IP address identifies the receiver of the secure information.

Security protocol specifies which protocol to use. IMS has specified Encapsulated Security Payload (ESP), described in Section A.1.3.

In IMS, SAs are established between the P-CSCF and the UE so that information can travel securely between the user and the network. How this is done is illustrated in Section B.1.1. Inside the network the traffic is not secured with IPsec since this is considered a protected domain with no physical access for attackers. [14; 16]

79
A.1.2 IKE

Internet Key Exchange (IKE) defined in RFC 2409 [24] is used to exchange keys securely with the Ephemeral Diffie-Hellman key exchange algorithm when setting up SAs. IKE is used in conjunction with the Internet Security Association and Key Management Protocol (ISAKMP) framework for setting up SAs. This is done in two steps. First an ISAKMP SA is established through which information is sent. After that, other SAs are created to be used for the real communication. [14; 16]

A.1.3 ESP

To make IP packets secure, IMS uses Encapsulated Security Payload (ESP) defined in RFC 2406 [23] in the SAs that are established. ESP provides integrity of the content of the IP packet including the ESP header and encryption for the data protected by ESP but not the ESP header. [14; 16]

A.2 SIP Security Mechanism Agreement

In order to agree on the security mechanism that is to be used between the UE and the P-CSCF the SIP Security Mechanism Agreement, defined in RFC 3329 [35], is used. This is a mechanism that requires that two messages are sent. [16]

The first message is the first REGISTER that the UE sends. In it, the UE specifies which mechanisms it supports and the parameters that goes with them. The P-CSCF removes and saves this information before sending the REGISTER to the S-CSCF. When the response comes from the S-CSCF the P-CSCF adds information about its supported mechanisms with priorities. After sending the 401 to the UE the P-CSCF can start using the security mechanisms since the UE is forced to choose the highest prioritized mechanism that both support. An example of this procedure is shown in Appendix B.1.1 [16]
A.3 Interoperator security

So far this chapter has discussed the security between the UE and the P-CSCF. This is enough for a single operator without roaming agreements with other operators, but if the communication is to be sent between operators it needs to be secure. This is accomplished by the SEG. The SEG resides on the border to other networks and makes sure security policies are enforced. Encryption of traffic between different SEGs over the Za interface is mandatory, while encryption from inside the network to the SEG is optional. [16]

The Za interface uses IPsec to set up SAa between different SEGs and the procedure is similar to that between a UE and its P-CSCF. Two IPsec with ESP SAs are set up to handle two-way traffic as well as an additional ISAKMP SA that stays active at all time to handle key management. [16]
Appendix B

IMS Examples

This appendix will show some basic examples of how messages flow in standard IMS and how the different parts interact during the most common transactions: registration and session initiation. Both sections start with a very simple example followed by a few subsections with extra features.

B.1 Registration examples

Figure [B.1] is an overview of how a basic successful registration is accomplished. When the I-CSCF receives a REGISTER request, it sends an UAR Diameter request to the HSS. By doing this, it lets the HSS determine if this is a valid subscriber to the network and answer with the appropriate S-CSCF. If the user already has an allocated S-CSCF the HSS will answer with that S-CSCF, otherwise it will allocate a S-CSCF and answers with the newly allocated S-CSCF. The I-CSCF then forwards the REGISTER to the S-CSCF. [14]

Upon receiving the REGISTER request from the I-CSCF, the S-CSCF sends an MAR request to the HSS requesting the AV from the HSS. These vectors contain a RAND that is sent back to the user in a 401 Unauthorized response. [14]

The user formulates a RES to the RAND presented by the S-CSCF and sends a new REGISTER request containing the answer. The I-CSCF receives the REGISTER from the P-CSCF and again sends an UAR to the HSS to find the appropriate S-CSCF to forward it to. When the REGISTER eventually

---

1Diameter requests UAR, MAR and SAR with their respective answers UAA, MAA and SAA are parts of the Cx and Dx Diameter Applications described in TSs 29.228 [3] and 29.229 [4].

2The AV, RAND, RES and XRES are described in Section 3.4.1.
Figure B.1: A basic registration. [14]

Figure B.2: Setting up security between the UE and the P-CSCF. [16]
reaches the S-CSCF, it verifies that the UE’s RES is the same as the XRES and registers the user. [14]

The S-CSCF notifies the HSS with the SAR request and asks for the user profile for the newly registered user, which is sent back in the SAA response. [14]

### B.1.1 Setting up security

During the REGISTER request, the UE and the P-CSCF set up SAs to be used between them in all communication until the UE is deregistered. Figure [B.2] shows how this is done. The flow is a part of Figure [B.1] namely the part between the UE and the P-CSCF. The I-CSCF is hinted just to show which specific headers and parameters are sent to the I-CSCF. Parameters and headers are written under the line of each message and are of course not the only headers sent, but the most important in this context.

In the first REGISTER sent by the UE, the UE includes the headers Require: sec-agree and Proxy-Require: sec-agree to show that it will only connect if security is provided. The header Security-Client is included to indicate which security mechanisms it can support and the parameter for each mechanism. This is part of the SIP Security Mechanism Agreement and IPsec SA set up discussed in Section [A.2] and [A.1], respectively. [16]

These headers are saved and removed by the P-CSCF before it sends the REGISTER to the I-CSCF with the parameter integrity-protected = "no" in the Authorization header. [16]

Eventually a 401 Unauthorized response containing the RAND will arrive to the P-CSCF. The response will also contain the IK and CK to be used when setting up the SAs. The P-CSCF removes the IK and CK and finally inserts the header Security-Server where it specifies which security mechanisms it can provide and the parameters and priority of them. [16]

When the P-CSCF has sent the 401 to the UE, it sets up the SAs and will from now on only accept requests if they come through the SAs. The UE does the same thing when it receives the 401 and is informed of the security mechanism to use. It calculates the IK and CK from the shared secret K and can now start using the SAs. [16]

Through the SAs, the UE sends the next REGISTER that contains the RES to the RAND. The UE also includes the Require: sec-agree, Proxy-Require: sec-agree, Security-Client and Security-Verify headers, where the last is a duplicate of the Security-Server it received in the 401. [16]

The P-CSCF removes everything but the RES and adds the integrity-protected = "yes" in the Authorization header before sending it on to the I-CSCF. [16]
Figure B.3: The UE’s and P-CSCF’s subscriptions to the UE’s registration state. [16]

Figure B.4: Deregistering from the network. [16]
B.1.2 Subscription to registration state

When the UE has successfully registered to the network and received the final 200 OK, it will send a SUBSCRIBE to the S-CSCF to indicate that it wants to know if the registration state changes. The P-CSCF also subscribes to this information. Since these two SUBSCRIBE requests are two different dialogs there will be two 200 OK and two NOTIFY sent at once from the S-CSCF. The NOTIFY requests will inform the UE and the P-CSCF of the current registration state. Figure B.3 that illustrates this flow should follow after the last 200 OK in Figure B.1. [16]

B.1.3 Deregistration

Deregistration can be done either by the UE (eg. hanging up the phone) or the network (eg. user no longer a paying customer). Figure B.4 shows what a UE-initiated deregister looks like. Observe that it is the REGISTER request with the parameter expires=0 in the Contact header that is used for deregistering. In a network-initiated deregister, the S-CSCF only sends out the NOTIFY requests to inform the UE and the P-CSCF that the user has been deregistered. [16]

B.2 Session initiation examples

How a simple session initiation is accomplished can be seen in Figure B.5. The UE sends an INVITE request with another UE as recipient. It will traverse its own P-CSCF and S-CSCF. At the S-CSCF, the initial filters will be checked before the request is sent onwards to an I-CSCF. The I-CSCF will find out in which S-CSCF the terminating UE is registered. When this S-CSCF receives the INVITE, it will check its initial filters for a match. After all filters have been handled the INVITE will traverse a P-CSCF and finally arrive at the terminating UE. [16]

At every leg of the journey a 100 Trying response will be sent back to prevent the sender from resending again and again. The reason is that it will take some time before a real provisional response is sent back. The 183 Session Progress is sent back from the terminating UE to the originating UE to inform the originating UE that the INVITE has been received and that the dialog is underway. [16]

Finally the terminating UE starts to let the user know it is being contacted (eg. ringing³). After the UE has informed the user, it will send back a 180

³At least it will do something, although it will most probably not sound anything like
Figure B.5: A basic session initiation. [16]
Ringing to the originating UE. [16]

If the terminating user answers the UE will send a 200 OK to let the originating UE know the session has been set up. Since the 200 is the final response to an INVITE, the originating UE must send an ACK request to acknowledge that the session has been initiated. The ACK is the only request not requiring a response and the UEs can now start sending media. Also note that the ACK does not traverse the I-CSCF since this is a new request and the UE’s S-CSCF now know the routing information to bypass the I-CSCF. [16]

B.2.1 Media negotiation

Figure [B.5] should be extended with Figure [B.6] from the 183 session progress and downwards to allow media negotiation to take place.

The first INVITE (not shown in Figure [B.6]) contains the first SDP offer with the media that the originating UE wants the session to have and what codecs the UE supports. In the 183 Session Progress, the terminating UE will respond with the media and codecs that it supports. Note that the user is not informed of the session yet. First the media has to be negotiated. [16]

The originating UE now chooses one of the codecs that both support for each media stream it has requested. If a media stream has no mutually supported codecs it will be dropped. The UE then sends a PRACK request that acknowledges the provisional 183 response. This PRACK includes the final offer with at most one\(^4\) codec per media stream and SDP attributes that indicate that no reservation has been made although it will be required. Immediately after sending the PRACK the UE will start reserving resources for the streams. [16]

When the terminating UE receives the PRACK, it will respond with a 200 OK that includes a copy of the agreed codecs and SDP attributes indicating that no reservation has been made. When the 200 OK is sent, the UE will start reserving resources. [16]

When both the reservation and the 200 is received by the originating UE, it will send an UPDATE request where it indicates that reservation is complete. The UPDATE will be answered by the terminating UE with a 200 OK and when its resource reservation is made as well the user will be notified and a 180 Ringing will be sent back to the originating UE. Since reliable provisional responses are used, the 180 Ringing must be answered with a PRACK request which generates a 200 OK response from the terminating UE. [16]

---

4 Zero if the media stream is to be dropped.
Figure B.6: Negotiating media and reserving resources. [16]
Finally the user answers and after the 200 OK response and the ACK request the session can be started. Note that only the responses (provisional and final) that belong to the first INVITE traverse the I-CSCF. All other requests and responses skip this the I-CSCF they follow the routing information received in the 183 Session Progress. [16]

**B.3 Terminating a session**

Termination a session is as easy as sending a BYE which is then responded to with a 200 OK by the other side. This is shown in Figure B.7.
Appendix C

SIP signaling schema

This appendix gives detailed information on the SIP signaling schema designed and used in this thesis. The parts included in this specification are the mandatory signaling and the mandatory content of messages and the description is for a general implementation.

C.1 General description

There are four request types used REGISTER to register and authorize the UE, INVITE and BYE to control a charging session and INFO (specified in [26]) to send charging information.

The registration involves the IMS authorization schema with two REGISTER requests sent, where the second REGISTER contains the answer to the random challenge sent by the S-CSCF in a 401 Unauthorized as a response to the first REGISTER.

When an application starts, it sends an INVITE to the AS responsible for charging. The AS checks if the subscriber is authorized to use the application and in that case starts an charging session with the UE.

After the application is authorized and the charging session is set up it can send charging information to the AS. The information is always sent as the content of two INFO requests where the first always has the charging state Requested and the second can have either a Charged or Dismissed charging state. The schema handles two types of services Event or Session and also accommodates both online and offline charging. The type of service is included in the information sent, while information regarding the type of account resides in the AS. Because of the content syntax specified we define a new extension name *ue-charging* that we can add to the *Require* header field when sending INFO requests to the AS.
C.2 Definitions

Some words and concept will benefit from a formal description and these apply throughout this appendix.

**Charging dialog** is all messages from the first INFO request to the 200 OK returned when the charging information is either stored in the BD or has been dismissed.

**Charging message** is a message sent with charging information. It can be either an INFO request or a response from the AS.

**Charging record** is a saved record of the information sent by the UE to the AS. It can be of an intermediate form when the charging state is Requested and of a finalized form when the charging state is Charged and the record is saved to a long-term storage.

**Charging session** is the session established with the INVITE between the UE and the AS for the sole purpose of sending charging information.

**Charging state** is the current state of the charging dialog. The three first states are Requested, Charged and Dismissed. A state change occurs when the UE sends an INFO request with a new state attribute. Finalized is the fourth state that symbolizes that the charging record is finalized and this is when the state is either Charged or Dismissed.

**Offline charging** is when the subscriber receives an invoice at the end of each billing period.

**Online charging** is when the subscriber buys credits in advance that can be used to access services.

C.3 Content of messages

C.3.1 Content of REGISTER requests

In accordance with IMS standards, the REGISTER requests has no content.

C.3.2 Content of INVITE and BYE requests

Neither the INVITE nor the BYE has any specified content. This is because the dedicated AS knows this is a charging session and that no media session is set up inside this session. A special content could be required for the network to accept the session, but this is outside the scope of this thesis.
C.3.3 Content of INFO requests

The content of the INFO requests is comprised of four lines using the general syntax `<header>`: `<value>` where the last line’s `<value>` is made up of seven attributes with the syntax `<attribute>=<value>;`. The headers are all defined in the ASRecord CDR definition in [8] which forms the basis for the syntax. The full syntax of the content is shown below.

ServiceRequestTimeStamp: `<timestamp>`
ServiceDeliveryStartTimeStamp: `<timestamp>`
ServiceDeliveryEndTimeStamp: `<timestamp>`
ServiceSpecificData: state=<int>;
    type=<int>;
    info=<string>;
    units=<int>;
    unittype=<int>;
    unitprice=<float>;
    cdrid=<int>;

The `ServiceSpecificData` may also be written in a single line or any number of lines as long as each attribute is separated by a semicolon. All possible syntaxes must be supported by the AS. Each of the types is defined below and some have different meaning depending on the others.

`timestamp` is defined by [8] and has the syntax `YYMMDDhhmmsShmm` where the five last characters contain the timezone offset from Coordinated Universal Time (UTC) and S can have the values + or -. The three header fields indicate when the request for the service was made, when the service started and when the service ended. The first is mandatory in a service request (INFO with state=1) and the last two are mandatory when the service is charged (INFO with state=2).

`state` indicates if the information sent is to dismiss a previous request (=0), request a service (=1) or charge a service (=2). This attribute is mandatory.

`type` indicates if the service is an event (=0) or a session (=1). This attribute is mandatory and cannot change during a charging dialog. New additional types can be defined if such a need is identified.

`info` is a description of the service offered (eg. the title of a movie viewed using VoD). This attribute is typically the part that makes sense to the subscriber and must provide useful information. It is mandatory, but
can change during a dialog and the last given value must be what is stored by the AS.

**units** is set to the number of units requested or used by the service. It has different meaning for different states. It is mandatory for `state=1` and `state=2` and has the following meanings for different combinations.

INFO with `state=0` No meaning.

200 OK for `state=0` No meaning.

INFO with `state=1` The number of units requested by the UE.

200 OK for `state=1` The number of units the AS has granted.

INFO with `state=2` The number of units used by the service. This value must never be more than the returned granted number in the first 200 OK.

200 OK for `state=2` The number of units that was charged for the service.

**unittype** is a description of the type of units used (eg. seconds or pieces). This attribute is mandatory and must not change during a charging dialog.

**unitprice** is the price (in the currency used by the system) per unit. This attribute is mandatory and must not change during a charging dialog.

**cdrid** is decided by the AS and returned in the 200 OK for `state=1` and is used to correlate information sent in different requests with an existing intermediate charging record. This attribute is mandatory, except for the INFO with `state=1`, and must not change after being set.

## C.4 URIs

There are four URIs of importance and while it is possible to give them any values, it is recommended that they follow certain guidelines.

**Private URI** does not really need a special value, but since this is an IMS simulation a Private URI must be registered to the subscriber. In a real IMS network the Private URI would be the one assigned to the subscriber and a possible syntax is `<subscriberid>@<domain>`.

**Public URI** is a general public URI used when authorizing the subscriber with the network. It is typically used to reflect the UE’s identity and a possible syntax is `<ueid>@<domain>`.
Application URI is the public URI of a certain application on a specific UE. It is used to identify both the UE and the application used. There is one application URI for each application on the UE and the preferred syntax is `<appid>.<ueid>@<domain>.

Charging URI is used as request URI when sending requests containing charging information and should be consistent. This is to ensure that the Initial Filter Criteria in the S-CSCF can detect the request as charging information and route it to the correct AS. One possible syntax is `charging@<domain>`.

C.5 SIP signaling

Figure [C.1] shows the signaling involved in a successful charging. This signaling can be divided into two parts: Registration and Charging.

C.5.1 Registration

The registration follows the IMS specifications and if used in a real IMS network this part would be identical to the signaling there. The important part is the authorization which is handled with two REGISTER requests and the Authorization header described in [30]. The REGISTER requests are sent by the STB with that STB’s Public URI. The essential part of the messages is shown in figures [C.2 to C.5]. These should not be seen as complete messages as they are examples that demonstrate the most important and relevant parts. If in doubt, IMS specifications are to be used.

C.5.2 Charging

The essential part of the signaling schema involves charging. All charging is done in a two-step manner. First the service is requested and then it is charged or dismissed. The typical messages involved are shown in Figures [C.6 to C.11]. Since the CSCF more or less acts as a proxy and just adds and removes Via and Route headers, the presence of the CSCF is only hinted to illustrate that messages traverse it. The figures only show the parts important to charging and additional header fields are necessary to follow IMS specifications.

The first request sent is always the INVITE seen in Figure [C.6] and its purpose is to set up a charging session between `appid.userid@domain.com` and `charging@domain.com`. During this process, the use of the application is authorized.
Figure C.1: The successful charging of a service
REGISTER sip:domain.com SIP/2.0
Via: SIP/2.0/UDP [192.168.1.1]
Max-Forwards: 10
To: <sip:ueid@domain.com>
From: <sip:ueid@domain.com>
Contact: <sip:ueid@192.168.1.1:5060>; expires=3600
Call-ID: ndjdnH84nf43H09unf4K4i6eD
CSeq: 1 REGISTER
Authorization: Digest username="private@domain.com",
    realm="domain.com",
    nonce="",
    uri="sip:domain.com",
    response=""
Require: sec-agree
Content-Length: 0

Figure C.2: (1) REGISTER

SIP/2.0 401 Unauthorized
Via: SIP/2.0/UDP [192.168.1.1]
To: <sip:ueid@domain.com>
From: <sip:ueid@domain.com>
Call-ID: ndjdnH84nf43H09unf4K4i6eD
CSeq: 1 REGISTER
WWW-Authenticate: Digest realm="domain.com",
    nonce="vf7di5gh4fhfg5f3r3d",
    algorithm=AKAv1-MD5
Content-Length: 0

Figure C.3: (2) 401 Unauthorized
REGISTER sip:domain.com SIP/2.0
Via: SIP/2.0/UDP [192.168.1.1]
Max-Forwards: 10
To: <sip:uid@domain.com>
From: <sip:uid@domain.com>
Contact: <sip:uid@192.168.1.1:5060>; expires=3600
Call-ID: ndjdH43H09uf4K4i6eD
CSeq: 2 REGISTER
Authorization: Digest username="private@domain.com",
    realm="domain.com",
nonce="vf7di5gh4fhfg5f3r3d",
algorithm=AKAv1-MD5,
    uri="sip:domain.com",
    response="f02nd37n484c5nsd"

Require: sec-agree
Content-Length: 0

Figure C.4: (3) REGISTER

SIP/2.0 200 OK
Via: SIP/2.0/UDP [192.168.1.1]
To: <sip:uid@domain.com>
From: <sip:uid@domain.com>
Contact: <sip:uid@192.168.1.1:5060>; expires=3600
Call-ID: ndjdH43H09uf4K4i6eD
CSeq: 2 REGISTER
Content-Length: 0

Figure C.5: (4) 200 OK
INVITE sip:charging@domain.com SIP/2.0
Via: SIP/2.0/UDP [192.168.1.1]
Route: <sip:cscf@domain.com>
To: <sip:charging@domain.com>
From: <sip:appid.ueid@domain.com>
CSeq: 3 INVITE
Require: ue-charging
Content-Length: 0

Figure C.6: (5) INVITE

INFO sip:charging@domain.com SIP/2.0
Via: SIP/2.0/UDP [192.168.1.1]
Route: <sip:cscf@domain.com>
To: <sip:charging@domain.com>
From: <sip:appid.ueid@domain.com>
CSeq: 4 INFO
Require: ue-charging
Content-Length: 277

ServiceRequestTimeStamp: 061002192000+0200
ServiceDeliveryStartTimeStamp:
ServiceDeliveryEndTimeStamp:
ServiceSpecificData: state=1; type=0;
           info=Video on Demand: Titanic;
           units=2; unittype=pcs; unitprice=6950;
           cdrid=

Figure C.7: (7) INFO
SIP/2.0 200 OK
Via: SIP/2.0/UDP [192.168.1.1]
To: <sip:charging@domain.com>
From: <sip:appid.ueid@domain.com>
CSeq: 4 INFO
Content-Length: 287

ServiceRequestTimeStamp: 061002192000+0200
ServiceDeliveryStartTimeStamp:
ServiceDeliveryEndTimeStamp:
ServiceSpecificData: state=1; type=0;
    info=Video on Demand: Titanic;
    units=2; unittype=pcs; unitprice=6950;
    cdrid=3759365936;

Figure C.8: (8) 200 OK

INFO sip:charging@domain.com SIP/2.0
Via: SIP/2.0/UDP [192.168.1.1]
Route: <sip:cscf@domain.com>
To: <sip:charging@domain.com>
From: <sip:appid.ueid@domain.com>
CSeq: 5 INFO
Require: ue-charging
Content-Length: 303

ServiceRequestTimeStamp:
ServiceDeliveryStartTimeStamp: 061002192010+0200
ServiceDeliveryEndTimeStamp: 061003024000+0200
ServiceSpecificData: state=2; type=0;
    info=Video on Demand: Titanic;
    units=2; unittype=pcs; unitprice=6950;
    cdrid=3759365936;

Figure C.9: (9) INFO
SIP/2.0 200 OK
Via: SIP/2.0/UDP [192.168.1.1]
To: <sip:charging@domain.com>
From: <sip:appid.ueid@domain.com>
Call-ID: ndjdmH84nf43H09unf4K4i6eD
CSeq: 5 INFO
Content-Length: 321

ServiceRequestTimeStamp: 061002192000+0200
ServiceDeliveryStartTimeStamp: 061002192010+0200
ServiceDeliveryEndTimeStamp: 061003024000+0200
ServiceSpecificData: state=2; type=0;
    info=Video on Demand: Titanic;
    units=2; unittype=pcs; unitprice=6950;
    cdrid=3759365936;

Figure C.10: (10) 200 OK

BYE sip:charging@domain.com SIP/2.0
Via: SIP/2.0/UDP [192.168.1.1]
Route: <sip:cscf@domain.com>
To: <sip:charging@domain.com>
From: <sip:appid.ueid@domain.com>
CSeq: 6 BYE
Require: ue-charging
Content-Length: 0

Figure C.11: (11) BYE
Figure C.7 shows a typical first request of service. Note in the SIP header that the message is routed via the CSCF and that the request URI is the charging URI decided for the domain. Also note that the From header field gives enough information to identify both the UE and the application used to provide the service. Finally, the Require header field shows that we demand that the AS can handle the content properly. In the content, the ServiceRequestTimeStamp is set and from the ServiceSpecificData it can be deduced that an event (type=0) is requested (state=1). The event requested is watching Titanic (info=Video on Demand: Titanic) a maximum of two times (units=2, unittype=pcs) at the cost of 69.50 SEK per time (unitprice=6950). Note that no cdrid exists as this is the first message in the charging dialog.

In the response (Figure C.8) from the AS, all data is replicated to show what has been granted by the AS.

The second INFO (Figure C.9) from the UE says that the service was provided to the subscriber and should be charged (state=2). The movie was obviously so good that it was seen twice (units=2) starting in the evening at 7:20 pm on Oct 2 and ending in the night at 2:40 am.

The final response from the AS (Figure C.10) includes all the data collected throughout the charging dialog in the content.

When the application has delivered all of its services, it needs to send a BYE (Figure C.11) to close the charging session.

C.6 Possible error responses

The schema allows the AS to send a couple of error messages back to the UE if something goes wrong during the charging dialog. An error response always contains a copy of the original content for the UE to analyze.

400 denotes Bad Request and means that something is wrong with the request. If a mandatory part is missing in the content the returned response must be

- 400 Bad Content: Missing <header/attribute>

and if a part does not follow the specifications a

- 400 Bad Content: Bad <header/attribute>

must be returned.
401 denotes Unauthorized and is used when a UE tries to send INFO messages outside a charging session. The AS must respond with a

- 401 Unauthorized: No session established

402 denotes Payment Required and if the charging indicates an online charging account and there is not enough credits to permit the requested service the response must be

- 402 Not enough credits

403 denotes Forbidden and if the supplied application URI in the From header field cannot be associated with a charging profile during charging session setup the response must be

- 403 Forbidden: No charging profile found

480 denotes Temporarily Unavailable and is returned if the service currently cannot perform the function required. If database connections are lost the response must be

- 480 No database connection
Appendix D

Charging API

This appendix contains the C++ code that specifies the two interfaces in the charging API. For information on the files’ intended usage see Chapter [7].

D.1 ICharging.h

This is the interface implemented by the charging module and used by applications to send charging information.

```cpp
#ifndef ICHARGING_H
#define ICHARGING_H

#include <string>

class ICharging {
public:
    ICharging() {}

    virtual ~ICharging() {} = 0;

    virtual void RegisterApplication(int appID) = 0;
    virtual void DeregisterApplication(int appID) = 0;
    virtual void RequestService(int appID,
                                std::string requestTimestamp,
                                int type,
                                std::string info,
```
int units,
int unittype,
double unitprice) = 0;
virtual void ChargeService(
    int appID,
    std::string deliveryStartTimestamp,
    std::string deliveryEndTimestamp,
    int cdrID,
    std::string info
    int units) = 0;

virtual void DismissService(
    int appID,
    int cdrID) = 0;
};

#endif

Constructor
Registers the STB with IMS AKA procedure.

Destructor
Deregisters the STB by sending REGISTER with expires set to zero.

RegisterApplication
 Registers the application by initiating the charging session with INVITE.
appId  Unique identifier to know which application made the call.

DeregisterApplication
Deregisters the application by sending BYE to end the charging session.
appId  Unique identifier to know which application made the call.

RequestService
Requests the delivery of a service.
appId  Unique identifier to know which application made the call.
requestTimestamp  The time the service was requested.
**type**  The type of service requested (eg. session or event).

**info**  A short description of the service.

**units**  Number of units to reserve.

**unitype**  The type of the units.

**unitprice**  The price per unit.

**ChargeService**

Tells the AS to charge for the service delivered.

**appId**  Unique identifier to know which application made the call.

**deliveryStartTimestamp**  The time the service delivery started.

**deliveryEndTimestamp**  The time the service delivery ended.

**cdrID**  The id received when the service was requested.

**info**  A short description of the service (can be changed).

**units**  The number of units used.

**DismissService**

Tells the AS to dismiss the service that was requested, but not delivered.

**appId**  Unique identifier to know which application made the call.

**cdrID**  The id received when the service was requested.

### D.2  IChargingObserver.h

This is the interface implemented by the application and used by the charging module to notify the application about the response.

```cpp
#ifndef ICHARGINGOBSERVER_H
#define ICHARGINGOBSERVER_H

#include <string>

class IChargingObserver {
```
public:
  virtual void OnChargingEvent(
    const int& eventID,
    const int& cdrID) = 0;
};

#endif

OnChargingEvent

Called when an event has occurred in the charging module.

eventID Identifies which event that has occurred.

cdrID Returned when any of the three service-functions were responsible for the event returned.
# Acronyms

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3G</td>
<td>3rd Generation</td>
<td>1</td>
</tr>
<tr>
<td>3GPP</td>
<td>3rd Generation Partnership Project</td>
<td>1</td>
</tr>
<tr>
<td>3GPP2</td>
<td>3rd Generation Partnership Project 2</td>
<td>17</td>
</tr>
<tr>
<td>AAA</td>
<td>Authentication, Authorization and Accounting</td>
<td>20</td>
</tr>
<tr>
<td>AKA</td>
<td>Authentication and Key Agreement</td>
<td>29</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
<td>2</td>
</tr>
<tr>
<td>ARPU</td>
<td>Average Return Per User</td>
<td>1</td>
</tr>
<tr>
<td>AS</td>
<td>Application Server</td>
<td>25</td>
</tr>
<tr>
<td>AUTN</td>
<td>network authentication token</td>
<td>29</td>
</tr>
<tr>
<td>AV</td>
<td>Authentication Vector</td>
<td>29</td>
</tr>
<tr>
<td>AVP</td>
<td>Attribute Value Pair</td>
<td>20</td>
</tr>
<tr>
<td>AVP</td>
<td>Audio/Video Profile</td>
<td>21</td>
</tr>
<tr>
<td>B2B</td>
<td>Back-to-Back</td>
<td>25</td>
</tr>
<tr>
<td>BD</td>
<td>Billing Domain</td>
<td>30</td>
</tr>
<tr>
<td>BGCF</td>
<td>Breakout Gateway Control Function</td>
<td>26</td>
</tr>
<tr>
<td>CCF</td>
<td>Charging Collection Function</td>
<td>30</td>
</tr>
<tr>
<td>CDMA</td>
<td>Code Division Multiple Access</td>
<td>17</td>
</tr>
<tr>
<td>CDR</td>
<td>Charging Data Record</td>
<td>30</td>
</tr>
<tr>
<td>CK</td>
<td>cipher key</td>
<td>29</td>
</tr>
<tr>
<td>COPS</td>
<td>Common Open Policy Service</td>
<td>21</td>
</tr>
<tr>
<td>CSCEF</td>
<td>Call/Session Control Function</td>
<td>24</td>
</tr>
<tr>
<td>CTF</td>
<td>Charging Trigger Function</td>
<td>30</td>
</tr>
<tr>
<td>DVR</td>
<td>Digital Video Recorder</td>
<td>35</td>
</tr>
<tr>
<td>ECUR</td>
<td>Event Charging with Unit Reservation</td>
<td>32</td>
</tr>
<tr>
<td>Notation</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---</td>
</tr>
<tr>
<td>ESP</td>
<td>Encapsulated Security Payload</td>
<td>80</td>
</tr>
<tr>
<td>FTP</td>
<td>File Transfer Protocol</td>
<td>9</td>
</tr>
<tr>
<td>GGSN</td>
<td>Gateway GPRS Serving Node</td>
<td>26</td>
</tr>
<tr>
<td>GPRS</td>
<td>General Packet Radio Service.</td>
<td>18</td>
</tr>
<tr>
<td>GSM</td>
<td>Global System for Mobile Communication</td>
<td>17</td>
</tr>
<tr>
<td>HSS</td>
<td>Home Subscriber Server</td>
<td>25</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hypertext Transfer Protocol</td>
<td>9</td>
</tr>
<tr>
<td>I-CSCF</td>
<td>Interrogating-CSCF</td>
<td>24</td>
</tr>
<tr>
<td>IBCF</td>
<td>Interconnection Border Control Function</td>
<td>27</td>
</tr>
<tr>
<td>IDE</td>
<td>Integrated Development Environment</td>
<td>66</td>
</tr>
<tr>
<td>IEC</td>
<td>Immediate Event Charging</td>
<td>32</td>
</tr>
<tr>
<td>IETF</td>
<td>Internet Engineering Task Force</td>
<td>9</td>
</tr>
<tr>
<td>IGMP</td>
<td>Internet Group Management Protocol</td>
<td>36</td>
</tr>
<tr>
<td>IK</td>
<td>integrity key</td>
<td>29</td>
</tr>
<tr>
<td>IKE</td>
<td>Internet Key Exchange</td>
<td>79</td>
</tr>
<tr>
<td>IM</td>
<td>Instant Messaging</td>
<td>34</td>
</tr>
<tr>
<td>IM-SSF</td>
<td>IP Multimedia Service Switching Function</td>
<td>25</td>
</tr>
<tr>
<td>IMS</td>
<td>IP Multimedia Subsystem</td>
<td>1</td>
</tr>
<tr>
<td>IMS-GWF</td>
<td>IMS GateWay Function</td>
<td>32</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
<td>1</td>
</tr>
<tr>
<td>IP-CAN</td>
<td>IP Connectivity Access Network</td>
<td>22</td>
</tr>
<tr>
<td>IPsec</td>
<td>Internet Protocol security</td>
<td>79</td>
</tr>
<tr>
<td>IPTV</td>
<td>Internet Protocol Television</td>
<td>1</td>
</tr>
<tr>
<td>ISAKMP</td>
<td>Internet Security Association and Key Management Protocol</td>
<td>79</td>
</tr>
<tr>
<td>ISP</td>
<td>Internet Service Provider</td>
<td>20</td>
</tr>
<tr>
<td>ITU</td>
<td>International Telecommunication Union</td>
<td>17</td>
</tr>
<tr>
<td>K</td>
<td>shared secret</td>
<td>29</td>
</tr>
<tr>
<td>MEGACO</td>
<td>Media Gateway Control Protocol</td>
<td>22</td>
</tr>
<tr>
<td>MGCF</td>
<td>Media Gateway Control Function</td>
<td>26</td>
</tr>
<tr>
<td>MGW</td>
<td>Media GateWay</td>
<td>27</td>
</tr>
<tr>
<td>MLD</td>
<td>Multicast Listener Discovery</td>
<td>36</td>
</tr>
<tr>
<td>MRF</td>
<td>Media Resource Function</td>
<td>26</td>
</tr>
<tr>
<td>MRFC</td>
<td>Media Resource Function Controller</td>
<td>26</td>
</tr>
<tr>
<td>Notation</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>MRFP</td>
<td>Media Resource Function Processor</td>
<td>26</td>
</tr>
<tr>
<td>NAPT-PT</td>
<td>Network Address Port Translator-Protocol Translator</td>
<td>27</td>
</tr>
<tr>
<td>OCS</td>
<td>Online Charging System</td>
<td>32</td>
</tr>
<tr>
<td>OMA</td>
<td>Open Mobile Alliance</td>
<td>18</td>
</tr>
<tr>
<td>OSA-SCS</td>
<td>Open Service Access-Service Capability Server</td>
<td>25</td>
</tr>
<tr>
<td>P-CSCF</td>
<td>Proxy-CSCF</td>
<td>24</td>
</tr>
<tr>
<td>PDF</td>
<td>Policy Decision Function</td>
<td>26</td>
</tr>
<tr>
<td>PDP</td>
<td>Policy Decision Point</td>
<td>21</td>
</tr>
<tr>
<td>PEP</td>
<td>Policy Enforcement Point</td>
<td>21</td>
</tr>
<tr>
<td>PIM</td>
<td>Protocol Independent Multicast</td>
<td>36</td>
</tr>
<tr>
<td>PSTN</td>
<td>Public Switched Telephone Network</td>
<td>12</td>
</tr>
<tr>
<td>PVR</td>
<td>Personal Video Recorder</td>
<td>35</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
<td>19</td>
</tr>
<tr>
<td>RADIUS</td>
<td>Remote Authentication Dial In User Service</td>
<td>20</td>
</tr>
<tr>
<td>RAND</td>
<td>random challenge</td>
<td>29</td>
</tr>
<tr>
<td>RES</td>
<td>result</td>
<td>29</td>
</tr>
<tr>
<td>RFC</td>
<td>Request For Comments</td>
<td>9</td>
</tr>
<tr>
<td>RTCP</td>
<td>RTP Control Protocol</td>
<td>21</td>
</tr>
<tr>
<td>RTP</td>
<td>Real Time Protocol</td>
<td>20</td>
</tr>
<tr>
<td>RTSP</td>
<td>Real Time Streaming Protocol</td>
<td>36</td>
</tr>
<tr>
<td>S-CSCF</td>
<td>Serving-CSCF</td>
<td>24</td>
</tr>
<tr>
<td>SA</td>
<td>Security Association</td>
<td>79</td>
</tr>
<tr>
<td>SCUR</td>
<td>Session Charging with Unit Reservation</td>
<td>32</td>
</tr>
<tr>
<td>SDP</td>
<td>Session Description Protocol</td>
<td>13</td>
</tr>
<tr>
<td>SEG</td>
<td>Security Gateway</td>
<td>26</td>
</tr>
<tr>
<td>SGW</td>
<td>Signaling GateWay</td>
<td>27</td>
</tr>
<tr>
<td>SIP</td>
<td>Session Initiation Protocol</td>
<td>9</td>
</tr>
<tr>
<td>SIPs</td>
<td>Secure SIP</td>
<td>10</td>
</tr>
<tr>
<td>SLF</td>
<td>Subscription Locator Function</td>
<td>25</td>
</tr>
<tr>
<td>SMTp</td>
<td>Simple Mail Transfer Protocol</td>
<td>9</td>
</tr>
<tr>
<td>SPI</td>
<td>Security Parameter Index</td>
<td>79</td>
</tr>
<tr>
<td>SQL</td>
<td>Structured Query Language</td>
<td>64</td>
</tr>
<tr>
<td>SQN</td>
<td>sequence number</td>
<td>29</td>
</tr>
<tr>
<td>Notation</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>----------</td>
<td>--------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>STB</td>
<td>Set-Top Box</td>
<td>2</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol</td>
<td>20</td>
</tr>
<tr>
<td>TD-CDMA</td>
<td>Time Division Code Division Multiple Access</td>
<td>17</td>
</tr>
<tr>
<td>TLS</td>
<td>Transport Layer Security</td>
<td>10</td>
</tr>
<tr>
<td>TR</td>
<td>Technical Report</td>
<td>18</td>
</tr>
<tr>
<td>TrGW</td>
<td>Transition GateWay</td>
<td>27</td>
</tr>
<tr>
<td>TS</td>
<td>Technical Specification</td>
<td>18</td>
</tr>
<tr>
<td>UA</td>
<td>User Agent</td>
<td>10</td>
</tr>
<tr>
<td>UAC</td>
<td>User Agent Client</td>
<td>10</td>
</tr>
<tr>
<td>UAS</td>
<td>User Agent Server</td>
<td>10</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
<td>15</td>
</tr>
<tr>
<td>UE</td>
<td>User Equipment</td>
<td>24</td>
</tr>
<tr>
<td>UMTS</td>
<td>Universal Mobile Telecommunications System</td>
<td>1</td>
</tr>
<tr>
<td>URI</td>
<td>Unified Resource Identifier</td>
<td>10</td>
</tr>
<tr>
<td>UTC</td>
<td>Coordinated Universal Time</td>
<td>95</td>
</tr>
<tr>
<td>VoD</td>
<td>Video on Demand</td>
<td>34</td>
</tr>
<tr>
<td>VoIP</td>
<td>Voice over IP</td>
<td>19</td>
</tr>
<tr>
<td>WCDMA</td>
<td>Wide-band Code Division Multiple Access</td>
<td>17</td>
</tr>
<tr>
<td>XCAP</td>
<td>XML Configuration Access Protocol</td>
<td>21</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
<td>21</td>
</tr>
<tr>
<td>XRES</td>
<td>expected response</td>
<td>29</td>
</tr>
</tbody>
</table>
Bibliography


[6] 3GPP. TS 32.296 - Telecommunication management; Charging management; Online Charging System (OCS); Applications and interfaces (Release 6), 2006. Version 6.2.0.

[7] 3GPP. TS 32.297 - Telecommunication management; Charging management; Charging Data Record (CDR) file format and transfer (Release 6), 2006. Version 6.1.0.

[8] 3GPP. TS 32.298 - Telecommunication management; Charging management; Charging Data Record (CDR) parameter description (Release 6), 2006. Version 6.4.1.


Index

3G, 1
3GPP, 1, 17
   TR, 18
   TS, 18
3GPP2, 17

AAA, 20, 29
AKA, 29
   AUTN, 29
   CK, 29
   IK, 29
   K, 29
   RAND, 29
RES, 30
SQN, 29
XRES, 29

API, 2
ARPU, 1
AV, 29

B2B, 25

CDMA, 18
charging API, 53, 69, 73, 76

GPRS, 18, 19
GSM, 17

IDE, 66
IETF, 9, 18
IM, 34
IMS, 1, 17, 33, 41, 46, 53, 63, 73, 75
   charging

CDR, 30, 48, 51, 65
CTF, 30
ECUR, 32, 49
IEC, 32
SCUR, 32, 49

functions

AS, 25, 30, 46, 50, 65, 68, 73
BD, 30, 65
BGCF, 26, 30, 63
CCF, 30, 41, 65
CSCF, 24, 65, 68
CTF, 41
GGSN, 26
HSS, 25, 29, 64
IBCF, 27, 63
IMS-GWF, 32
IP-CAN, 22, 64
MGCF, 27, 63
MGW, 27, 63
MRF, 26
OCS, 32, 41, 65
PDF, 26, 64
PEP, 26, 64
SEG, 26, 63
SGW, 27, 63
SLF, 25, 29, 64
TrGW, 27, 63
UE, 24, 29, 41, 65, 79, 80

interfaces

Bi, 31
Bo, 32
Cx, 28
Dx, 28
Gm, 28
Go, 28
Gq, 28
ISC, 28, 32
Mg, 28
Mi, 28
Mj, 28
Mn, 28
Mp, 28
Mr, 28
Mw, 28
Mx, 28
Rf, 31
Ro, 32
Sh, 28
Ut, 28
Za, 28, 81
Zb, 28

IP, 1, 17

IPTV, 1, 33, 41, 73
Broadcast, 37
DVR, 35
entities
Application server, 37
Gateway, 39
Local VoD server, 39
Router, 39
STB, 2, 39, 41, 46, 50, 53, 54, 65, 73, 75
Streaming server, 37
VoD server, 37
Multicast, 37
PVR, 35
Unicast, 37
VoD, 34
ISP, 20
ITU, 18

NAPT-PT, 27

OMA, 18

protocols
COPS, 21
PDP, 21, 26
PEP, 21
Diameter, 20, 31, 32
AVP, 20
FTP, 9, 31, 32
HTTP, 9
IGMP, 36
IPsec, 79
ESP, 79, 80
IKE, 80
ISAKMP, 80
SA, 79
SPI, 79
MEGACO, 22
MLD, 36
PIM, 36
RADIUS, 20
RTP, 21
AVP, 21
RTCP, 21
RTSP, 36
SDP, 13
SMTP, 9
TCP, 20
TLS, 10, 75
UDP, 13, 21
XCAP, 22
XML, 22

PSTN, 12
QoS, 15, 19, 34, 39, 64
RFC, 9

signaling schema, 45, 53, 55, 65, 73
SIP, 9, 41, 45, 54
entities
proxy, 65
INDEX

Proxy servers, 11
Redirect servers, 11
Registrar, 10
registrar, 65
UA, 10
requests
ACK, 9
BYE, 9, 47, 51, 68, 74
CANCEL, 9
INFO, 15, 47, 50, 51, 68, 74, 76
INVITE, 9, 47, 51, 55, 68, 74, 76
MESSAGE, 47
NOTIFY, 15, 46, 76
OPTIONS, 9
PRACK, 15
REGISTER, 9, 29, 51, 55, 68, 74, 76, 80
SUBSCRIBE, 15, 46, 50, 76
UPDATE, 76
UAC, 10
UAS, 10
SIPS, 10
SQL, 64, 65

TD-CDMA, 17

UMTS, 1, 18
URI, 10
UTC, 95

VoIP, 19

WCDMA, 17
## Abstract

With the standardization of IP Multimedia Subsystem (IMS), the telecommunications scene becomes more and more converged and in the future we will most likely access our services from all kinds of devices and link them together. One important future access method that has so far been left out of the standardization is television. There is a need for Internet Protocol Television (IPTV) to work together with IMS and this thesis focuses on one aspect of that convergence, namely charging.

The problem explored in this thesis is if there is an efficient way of charging for IPTV services while taking advantage of the IMS charging functionality and this is done for two aspects of the problem. First, the possibility of an efficient Session Initiation Protocol (SIP) signaling schema is investigated and then a good charging Application Programming Interface (API) to be used when developing applications is investigated. The findings of these two investigations are then tested and improved during the implementation of a demo application.

This thesis delivers specifications for a signaling schema that enables a Set-Top Box (STB) to pass charging information to an IMS network via INFO requests inside a special charging session. The schema is small and extendable to ensure that it can be modified further on if necessary. The thesis also delivers an encapsulating and intuitive charging API to be used by developers who want to charge for their services.
Copyright

Svenska

Detta dokument hålls tillgängligt på Internet - eller dess framtida ersättare - under 25 år från publiceringsdatum under förutsättning att inga extraordinära omständigheter uppstår.

Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för ickekommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns det lösningar av teknisk och administrativ art.

Uphovsmannens ideella rätt innefattar rätt att bli nämd som upphovsmann i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart.

För ytterligare information om Linköping University Electronic Press se förlagets hemsida http://www.ep.liu.se/

English

The publishers will keep this document online on the Internet - or its possible replacement - for a period of 25 years from the date of publication barring exceptional circumstances.

The online availability of the document implies a permanent permission for anyone to read, to download, to print out single copies for your own use and to use it unchanged for any non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional on the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility.

According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement.

For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its WWW home page: http://www.ep.liu.se/

© Stefan Östergaard
Linköping, December 12, 2006