

# NetLab

## Use Cases for Interconnected Testbeds and Living Labs

Project number: **CP5-018**

Project title: **NETLAB**

Document title: **D2.2 IMS services requirements**

Deliverable type: **Report**

Contractual delivery date: **30/04/2010**

Actual delivery date: **17/05/2010**

Work package: **WP2**

Editor, organization: **Timo Koski, UTU**

Authors, organizations: **Fernando Ortigosa, Ericsson**

**Janne Ylitalo, Octopus Network**

**Andrés Marín López, Davide Proserpio, Fabio Sanvido,  
UC3M**

**Itziar Ormaetxea, SQS**

**Ivan Kotuliak, STUBA**

**Piritta Hakala, UTU**

Reviewers, organizations: **Timo Lahnalampi, Dimes**

**Francisco Rodríguez García, Telefonica**

**Fernando Ortigosa, Ericsson**

**Andrés Marín López, UC3M**

**Itziar Ormaetxea, SQS**

Distribution: **Public**

## **ABSTRACT**

The purpose of this deliverable is to define requirements for IMS services. The requirements are studied from different perspectives. A few IMS services are introduced and propositions to possible future IMS services are given. Service interworking and roaming between IMS testbeds are also covered.

### **Keywords**

IMS services, requirements, testbeds

## **EXECUTIVE SUMMARY**

IMS (IP Multimedia Subsystem) was originally designed for supporting IP-based multimedia services, telephony and messaging. It was one of the first concepts that different standardization institutions agreed on. IMS is the core technology in the NetLab project.

Different kinds of IMS services have different requirements. And the requirements depend on the aspect of the stakeholder, as well. For example, end-users of course experience the services differently than the service providers, and expect different things from the services than the service providers. Different operators also have different requirements for the IMS services. And the used network also creates its own requirements.

Besides these abovementioned, one of the key issues in the NetLab project is the Quality of Service (QoS). This especially concerns the different access-networks used in the IMS domain. Roaming is also very important and should be handled carefully, since the service users may choose to use either their own home network or some visited network to access the IMS services. Security requirements focus on the relationship between the user equipment and the network. IMS achieves security through a registration that provides mutual authentication between the IMS subscriber and his home network.

The IMS testbeds in the NetLab project provide the framework to thoroughly test IMS services in practice and verify that the services meet the requirements. Interoperability of the IMS systems in different locations is very important, and this is researched in the NetLab project.

## Table of contents

<b><u>ABSTRACT.....</u></b>	<b><u>2</u></b>
<b><u>EXECUTIVE SUMMARY.....</u></b>	<b><u>3</u></b>
<b><u>Table of contents.....</u></b>	<b><u>4</u></b>
<b>Document history.....</b>	<b>7</b>
<b><u>1 IMS for dummies.....</u></b>	<b><u>7</u></b>
1.1 Introduction.....	7
1.2 IMS and the market.....	7
1.2.1 Why IMS is needed?.....	7
1.2.2 Advantages of IMS.....	8
1.2.3 Standardization advantage.....	8
1.2.4 Benefits IMS offers.....	8
1.3 IMS components.....	9
1.4 Services.....	10
1.4.1 Voice and video call.....	10
1.4.2 Instant messaging.....	11
1.4.3 Purpose of interconnected IMS networks.....	11
<b><u>2 Requirements for IMS services.....</u></b>	<b><u>12</u></b>
2.1 End-user’s requirements.....	12
2.2 Fixed operator’s requirements.....	12
2.3 Mobile operator’s requirements.....	13
2.4 Fixed-mobile convergent operator’s requirements.....	14
2.5 Network requirements.....	14
2.6 Quality of Service.....	15
2.7 Roaming.....	16
2.8 Security.....	18
<b><u>3 IMS services.....</u></b>	<b><u>21</u></b>
3.1 Multi-platform environment.....	21
3.2 Security services.....	21
3.3 IMS service example: Presence.....	22

[3.3.1 Presence service within the SIP framework.....22](#)

[3.3.2 Presence service within the IMS framework.....23](#)

[3.3.3 Architecture.....23](#)

[3.4 Innovative services.....24](#)

[3.5 IPTV specific services.....26](#)

[3.5.1 Extend voice services into the TV environment.....26](#)

[3.5.2 Extend data services into the TV environment.....27](#)

[3.5.3 Extend TV services into the wireless environment.....27](#)

[3.5.4 Establish consistent user experience across access media and devices.....27](#)

**[4 IMS testbeds.....28](#)**

[4.1 Solution for service interworking and roaming between testbeds.....28](#)

[4.2 Proposal for new authentication mechanisms.....31](#)

**Appendix A: Proposal of enhanced authentication mechanism.....33**

**[1 Overview.....33](#)**

**[2 Architecture of the proposal.....34](#)**

**[3 Implementation details.....35](#)**

[3.1 PANA EAP-TLS Implementation.....35](#)

[3.1.1 System Requirements.....35](#)

[3.1.2 Supported functionality and general description of the existing code.....35](#)

[3.2 OpenIMSCore modifications.....42](#)

[3.2.1 HSS.....42](#)

[3.2.2 SCSCF.....43](#)

[3.3 UCT IMS client modification.....44](#)

**[4 Testing.....46](#)**

[4.1 Architecture.....46](#)

[4.2 Tests.....46](#)

[4.2.1 Send modified register message .....46](#)

[4.2.2 PANA EAP-TLS Authentication.....47](#)

[4.2.3 Generation of security token and PANA security association establishment.....48](#)

[4.2.4 Send the security token to the HSS using Paa.....49](#)

[4.2.5 HSS parses the Paa message and stores the authentication token in the database.....50](#)

[4.2.6 MAA/MAR Exchange.....50](#)

[4.2.7 Complete IMS registration using our protocol.....51](#)

**[5 Future work.....53](#)**



## Document history

Document version #	Date	Remarks
v0.1	31.08.2009	First draft
v0.2	24.09.2009	Some inputs added
v0.3	28.09.2009	More inputs added
v0.4	08.10.2009	More inputs added
v0.5	17.10.2009	More inputs and modifications
v0.6	22.10.2009	Executive summary added
v0.7	11.11.2009	Modifications
v0.8	3.5.2010	Added chapter 1, appendix A. Stilization.
v0.9	7.6.2010	Corrections on chapter 1

## 1 IMS for dummies

### 1.1 Introduction

The purpose of this document is to provide the reader a basic knowledge of IMS functionality and why it is a promising technology for the future mobile services. The document is targeted for a reader who is not a technically oriented person.

The beginning of this document describes the commercial side of IMS, taking into account different user groups involved - these being the end-user, the operator and the content owner. The document describes also general technology behind IMS without going too much into the detail.

### 1.2 IMS and the market

User and enterprise needs will drive multimedia service evolution for both mobile and fixed operators. Users expect to be able to do more with their communications services, for less money, and are showing an interest in services beyond voice. They are attracted by services that offer them access to a wide range of communications information and entertainment services in a user-friendly, cost-effective way. Users also want to be always best connected, i.e. they want access to the services wherever, whenever and however they want. [4]

#### 1.2.1 Why IMS is needed?

Mobile networks are undergoing a major transition in near future. The third generation networks (HSPA/HSPA+) and LTE enables increasingly fast data rates, which opens up new possibilities to

service developers. Today's mobile devices have large, high-resolution displays, multiple cameras for video enabled services and lot of resources for running wide range of applications. [2]

More and more of today's mobile devices are connected to network around the clock. This "always-on, always-connected" principle redefines applications for mobile devices. They're not anymore isolated entities which interact with the mobile user only but peer-to-peer entities which can interact with large user groups for instance. [2]

The largest change in so called "4G" –networks (note that LTE is not 4G) is the transformation from circuit switched to packet switched network. This means that a technology that has been used for decades will be replaced by technology which answers the needs of the future better. All data, including voice will be transferred as packet-switched on top of IP (Internet Protocol). Dialing a number and talking to people will only be a small subset of networking services available to the end-user.

The use of IP in core networks makes possible new service concepts and convergence between fixed and mobile networks. Being access agnostic and supporting both fixed and mobile clients and access, the IMS is a central building block for convergent communications.

### **1.2.2 Advantages of IMS**

It is important to realize that IMS is in fact independent of the access method user connects to the network. Users can connect to the network using mobile device's packet data connection, Wi-Fi or they can use their PC with xDSL. This kind of approach allows users to use the same services not only with their mobile devices but stationary devices too for example set-top boxes.

One of the biggest benefits IMS provides is that it increases overall productivity and effectiveness when it comes to developing new, innovative software to the all-IP network.

### **1.2.3 Standardization advantage**

3GPP released IMS specifications as part of 3GPP release 5 in year 2002. It was intended to be released earlier, in release 4 but 3GPP realized that it could not be completed in time and so they dropped it from release 4. Currently the latest frozen release is Rel. 9 and 3GPP is working on Rel. 10 which is scheduled to be frozen in March 2011. The IMS's used in NetLab project are mainly based on releases 6 and 7. [3]

The standard is not intended to standardize the applications but rather to aid the access of multimedia and voice applications (clients). IMS has a horizontal control layer which isolates access network from the service layer. Using standardized IMS cores ensures that IMS cores from different manufacturers will work together. They can, however, have non-standardized internal interfaces. The standards assure as well backwards compatibility between IMSs based on different releases.

### **1.2.4 Benefits IMS offers**

- Convergence
- Faster time to revenue
- investment protection
- Provides a flexible and scalable architecture
- Service innovation enablement

- New Services can be developed and trialed rapidly
- Open API

#### **1.2.4.1 Operators**

For operators, IMS takes the concept of layered architecture one step further by defining a horizontal architecture, where service enablers and common functions can be reused for multiple applications. The horizontal architecture in IMS also specifies interoperability and roaming, and provides bearer control, charging and security. What is more, it is well integrated with existing voice and data networks, while adopting many of the key benefits of the IT domain. This makes IMS a key enabler for fixed-mobile convergence. [4]

The horizontal architecture of IMS enables operators to move away from vertical ‘stovepipe’ implementations of new services – eliminating the costly and complex traditional network structure of overlapping functionality for charging, presence, group and list management, routing and provisioning. [4]

With an IMS-based network, operators will be able to set up and introduce new converged services much more quickly than before, achieving a faster time to market to meet changing customer demands. New functionalities can be added easily, lowering the cost of new services by avoiding the need to build and run a completely separate, parallel network to support each feature. This opens lucrative opportunities also for other telecoms companies developing products and services for next generation network environment.

In addition to speeding up and simplifying the service creation and delivery process, the reuse of common infrastructure, enablers and competence provided by IMS minimizes OPEX and CAPEX for operators – especially in areas such as service provisioning, O&M, customer care and billing. [4] For these reasons, IMS has become preferred solution for fixed and mobile operators’ multimedia business.

#### **1.2.4.2 Consumers**

As consumers use more and more network bandwidth while using their mobile devices they adapt new services much more easily than before. IMS offers a flexible way to provide new content to the end-user offering innovative, content rich services. Today’s consumers expect new services to offer a seamless experience across multiple access technologies. They want to be able to do the same things with their home computer and with their mobile devices.

IMS also provides end-to-end QoS (Quality of Service) with the underlying access and transport networks. First user’s client negotiates QoS parameters on application level and after that we can assume that the operators negotiate service-level agreement to guarantee sufficient quality level.

### **1.3 IMS components**

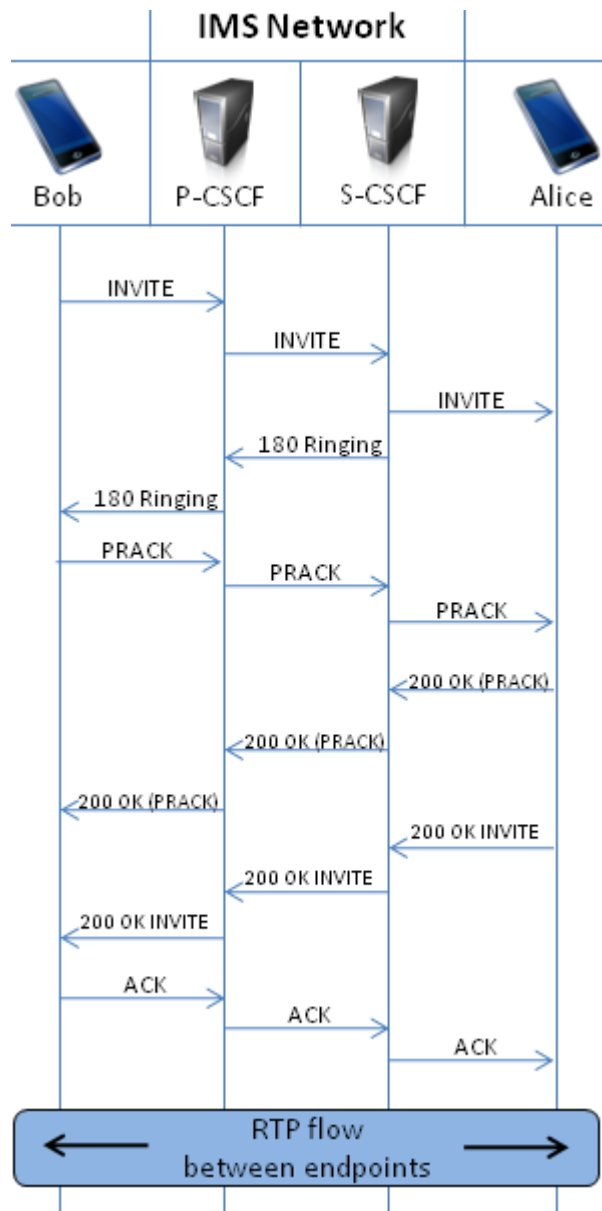
IMS is designed to use layered architecture, this means that transport and bearer services are in separate layers than for example signaling and session management services (see [1]). The IMS core network consists of entities called Call Session Control Function (CSCF). There are three main types of CSCFs: Serving-, Proxy- and Interrogating Call Session Control Functions. They manage the signaling and the media flows in the IMS network. IMS uses Session Initiation Protocol (SIP) as its main signaling protocol. SIP is a text based protocol similar to HTTP which means that the messages are easy to interpret. However IMS introduced few enhancements to the early SIP protocol to fulfill IMS requirements.

## 1.4 Services

The IMS core as it is only provides the very basic services such as voice call, video call, instant messaging and file sharing. More intelligent services can be added by connecting application servers to the IMS using standardized and open ISC interface (between S-CSCF and AS) and configuring appropriate service profile, initial filter criteria and trigger point in home subscriber server. Next few chapters briefly describe the functionality of the basic services.

### 1.4.1 Voice and video call

Voice and video calls with IMS aren't from consumer's point of view any different than in circuit switched technologies. However there have been plenty of changes behind the curtains. All session establishment, terminating and overall handling is based on SIP protocol. After initial registration ser can start to make voice and video calls. Figure 1 illustrates an example signaling flow where Bob after initial registration places a call to Alice. The example assumes that Bob's client has its resource ready before sending the initial SIP INVITE to Alice.



**Figure 1: Signalling flow example.**

Before sending the INVITE message, Bob's client determines the set of supported codecs it wishes to use. For example for voice only it lists preferred audio codecs and builds SDP offer request according these codecs. Another possibility is to offer both audio and video. For example Bob might want to use a video codec that supports H.263 and G.711 A-Law audio codec.

P-CSCF is the first contact point to IMS network, it receives the INVITE message and adds itself to Record-Route and Via header to ensure that the subsequent messages gets routed back the same route than the initial INVITE message. Next the message is routed to S-CSCF which performs analysis of the destination address and evaluates the initial filter criteria if there's any and routes the message to Alice via P-CSCF. 180 Ringing mean that Alice has accepted the audio and video codecs Bob suggested and is ready to start the media flow. Bob's client PRACKs the 180 Ringing message and Alice's client OKs the PRACK. When Alice's client finally accepts the call, a confirmation message is generated to the INVITE message and sent back to Bob. Finally Bob's client ACKs the 200 OK for INVITE and RTP stream starts to flow between the clients. It's important to realize that after the call is set up, the media is routed peer-to-peer between the clients and IMS doesn't intervene to it unless some changes to the call are needed.

#### 1.4.2 Instant messaging

Messaging is in its simplicity just about sending a message from user A to user B. In IMS there are two different types of messaging. These are immediate and session-based messaging. Immediate or page-mode messaging uses SIP protocols MESSAGE method to deliver the messages over IMS network. The message can consist of not only text but other multimedia also – such as audio clips or images. The message gets routed through IMS network the same way than the INVITE method in the example in chapter 4.1.

The other type of messaging is called session-based messaging. This type of messaging is actually more like phone call because before any messages can be transferred over the IMS network an end-to-end session is needed to be set up. In session-based messaging the session is negotiated with the use of SIP and SDP just like in IMS phone call. The actual protocol used for conveying the messages is called Message Session Relay Protocol (MSRP).

#### 1.4.3 Purpose of interconnected IMS networks

Experience shows that creating and expanding a mass market requires standards-based solutions that enable interoperability in several dimensions. Terminal-to-terminal interoperability is essential to create convenience and clarity in user's expectations of person-to-person communications services. At the same time, interoperability between operators is necessary to give users the freedom to roam between different networks.[4]

## 2 Requirements for IMS services

IMS services have different kinds of requirements. Requirements can, for example, depend on the different stakeholders or different kinds of services. Here, several IMS service requirements are highlighted from the aspects of the service end-users, different operators and the network. Also, issues concerning Quality of Service (QoS), roaming and security in IMS are covered.

### 2.1 End-user's requirements

End users will have a variety of end devices (STB, PC, handheld, portable devices, home equipment, MP3 players, etc.) with access to several networks. These devices, apart from being able to connect to their respective access networks, must implement a SIP stack through an IMS client.

Desirable service requirements:

- **Valuable service:** The user is interested and willing to pay only for services that are useful and valuable to him/her. The user is not interested in the network infrastructure but only in the services that are useful and usable. IMS as such is not interesting for the user but the IMS services are. Services that are fancy and funny but not really useful for the user will be abandoned in long term.
- **User-friendly service:** The service must be easy to use and the user should intuitively figure out how to use it. For difficult features, there should be instructions that are accessible to the user in a simple way. The service must be implemented for non-technical people.
- **Working at first time:** The service must preferably work for the user at first try because it is very unlikely that the user will try again. The service must therefore be deployed and tested carefully before being offered to the users.
- **Service reliability:** The user must be able to rely on that the service is working when he needs it. This means that the service must work most of the time, preferably 99,99 %.
- **Service robustness:** The service must be robust, i.e. it must continue to work no matter how serious mistake the user has made.
- **Service ubiquity:** The service must be operative at any time and no matter where the user is and what device is being used.
- **Uniform user experience - same service everywhere on any device and on any network:** Nowadays, each user may have several devices of different types and the service must appear more or less the same to the user regardless of the devices. The service must also be delivered continuously when the user is moving between different networks.

### 2.2 Fixed operator's requirements

Operators are moving towards the Next Generation Networks (NGNs), which should consolidate their traditional voice networks and other service delivery networks into a single converged IP-based network. NGNs in Europe are based on IMS and SIP as the session initiation protocol for voice services. The desirable features for services are the following:

- **Ability to offer IP telephony both on fixed and mobile networks:** Although IMS was intended originally for mobile networks it should be usable for both fixed and mobile networks. Fixed

operators view IMS as a technology that enables to expand their activities onto the mobile domain. As MVNOs (mobile virtual network operators) offer mobile subscriptions that uses mostly wireless LAN hotspots and the mobile networks when it is necessary, fixed operators can win back some of the traffic lost to mobile operators in the 90s.

- Incremental service deployment: At the first stage, the focus is on a few successful existing services such as voice, but it must be possible to introduce gradually new advanced services with time.
- Reduction of OPEX/CAPEX: The introduction of IMS is considered by fixed operators as a step to convert their legacy core infrastructure into IP, which should require less investment than traditional telecom equipment. In addition, the operation and management costs should be smaller due to the openness of IP equipment.
- Integration with legacy systems: It must be possible to seamlessly integrate IMS with legacy infrastructure and to maintain the same pre-IMS quality of service.
- Separate charging for connection and service usage: Since fixed operators want to offer their services on the mobile networks, a separate charging for connection and service usage is required. They must also have the possibility to distinguish the services and charge differently.

### 2.3 Mobile operator's requirements

Mobile markets in Europe are saturated with mobile telephony and prices are decreasing. Data services that require much more bandwidth have not been taken off in the market.

- “Transparent IMS”: As observed in the existing IMS commercial deployment, the mobile operator's promotion should be focused on innovative services such as video sharing, presence, messaging, etc. but not on IMS itself. On the contrary, IMS should be in the background as technology enabler.
- Richer communication: To be successful, it is not sufficient to promote IMS only as VoIP service. IMS should be promoted as richer communication that is provided on multiple communication media (voice, video, text, digital content, etc.).
- Incremental service deployment: At the first stage, the focus is on a few new services such as VoIP, PoC and Video sharing, but it must be possible to introduce gradually new advanced services with time.
- Integration with legacy systems: It must be possible to seamlessly integrate IMS with legacy infrastructure and to maintain the same pre-IMS quality of service.
- Integration with non SIP-based applications: For some mobile operators, it is very important that IMS could be easily integrated with non SIP-based applications. This means that proper interfaces must be provided.
- Shorter time to market with IMS: The development of new combinatorial applications, making use of multiple communication channels such as voice, video, text, content, etc., must be simpler and the time-to-market must be reduced considerably.
- Openness: The IMS platform must be open and allow the integration of third party services.
- Service roaming: For mobile operators, it is very important that service roaming is supported and that the user can have access to the IMS services no matter in which network he is located.

- Customization/segmentation: It should be possible for mobile operators to customize and pack the IMS services in various ways to aim at different market segments. A service profile management must be provided to allow easy modification of the services. The management of the User Interface and application flows must be flexible. It should be possible to perform remote update of the User Interface to change the branding and themes.
- Uniform user experience across terminals: It must be possible to ensure a uniform user experience across all terminals. Since mobile terminals come with different functions and capabilities, transcoding and adaptation capabilities must be provided. Both terminal and vendor independency must be supported.

## 2.4 Fixed-mobile convergent operator's requirements

For fixed and mobile operators IMS could be the catalyst for service convergence across wireless and wireline networks. The main requirement is profitability, i.e. to keep the costs down and revenue up in a very competitive market. Profitability is closely linked to the number of subscribers and usage of services.

- Same service both in fixed and mobile networks: For fixed-mobile convergent operators, it is very important to be able to offer the same services both in fixed and mobile networks. The goal is to promote the usage of the operator's network as much as possible. Mobile phones with IMS client must be able to operate in both fixed and mobile networks.
- Richer communication: To be successful, it is not sufficient to promote IMS only as VoIP service. IMS should be promoted as richer communication that is provided on multiple communication media (voice, video, text, digital content, etc.).
- Advanced applications involving both fixed and mobile devices: For convergent operators, it is very important to be able to offer convergent applications that make use of both mobile terminals and fixed devices such as stationary PC, IPTV, set top box, home gateway, etc.
- Service continuity: The services offered must be able to flow seamlessly over heterogeneous devices without interruptions. The user must not have to worry about the change of networks or devices.
- Interoperability: Since communication is universal and users from one operator must be able to interact with users subscribed to other operators, there must therefore be interoperability
  - Between operator's services (presence, community with other operators)
  - Between client frameworks
  - Between devices and device classes
  - Between fixed and mobile devices

## 2.5 Network requirements

The network, as well as the SIP stack in the terminal, must understand what to do with a new session. The session must be routed to the appropriate server; in the device, to the correct application. SIP (through SDP) solely indicates which media is to be used; that is, it does not provide any information about context.

The network must be able to apply the correct policy for the session, for example, by selecting the proper quality bearer.

If the session crosses operator borders, then for charging and settlement purposes, there needs to be some means of identifying the session type. A user might be consuming the resources from a network other than the one with the direct relationship to the service provider. These issues are unavoidable as long as the resources are finite and (relatively) expensive to produce (as is the case in the mobile world for the moment). It should be noticed that alternative business models, such as financing by way of advertising, do not eliminate this need. The key is the network-to-network interface (NNI). In other words, a concept of services needs to be established, which supports a multi-operator cost-and-revenue-sharing business environment without having to renegotiate complex interworking agreements every time a new service is launched.

## 2.6 Quality of Service

One of the challenges with an implementation of end-to-end Quality of Service (QoS) concerns the different access-networks used in the IMS domain. There are numerous of different technologies to implement QoS in existing networks. These could include the following: QoS Differentiation in UMTS, DiffServ in IP, 802.11e in WLAN, ATM QoS mechanisms, MPLS Traffic Class, etc.

In UMTS, the QoS differentiation is not needed in low loaded cells, but is required to ensure the functioning of the application in loaded multiservice scenario. UMTS QoS is divided into four main classes: conversational, streaming, interactive and background. These classes are further divided into three priority categories.

The conversational class is characterized by low end-to-end delay and symmetric or nearly symmetric traffic in person-to-person communications. The traffic categorized to the streaming class is more delay insensitive, but usually requires certain bandwidth to be maintained, like the conversational class. Streaming usually can tolerate some delay, and the packet delay variation can be hidden in the receiver's jitter buffer. Traffic in the interactive class usually requires actions from the end user, and in the background class, the data is not expected to be delivered in a certain time. Additional PDP (Packet Data Protocol) contexts can be opened with different kind of QoS setting associated with them, for example, 1st PDP context for audio and 2nd for video traffic.

This QoS is realized only within UMTS bearer service ending to operator core network gateway, but to be effective, it also needs to be mapped to other networks respectively. There is no one-to-one match of the traffic classification between UMTS and other, regular IP network.

From the edge router forward, Differentiated Services are a useful choice for marking the packets with desired priorities. This can be applied to a fixed line approach also. The classification of the traffic can be made even in an IMS client by setting the DSCP (Differentiated Services Code Point) bits in the IP header. In this case, the DSCP value can be transferred from an IP network to another. Theoretically there can be 26 (=64) different traffic classes, but usually there are four classes defined for Per-Hop Behavior (PHB). These classes consist of the following: Default, Expedited Forwarding (EF), Assured Forwarding (AF) and Class selector for backwards compatibility with IP Precedence (ToS-field in old IP header).

Default class consists of best-effort traffic, so there is no guarantee on delivery or latency. All traffic without DSCP marking will be placed in Default class. Background and Interactive classes in UMTS could most likely be mapped to a Default class in DiffServ. Expedited Forwarding characterizes to low delay, low packet loss and low jitter, which makes this class suitable for voice and video traffic. UMTS Conversational class could be mapped to a DiffServ Expedited Forwarding class. Assured Forwarding can be used to provide an assured delivery of packets. AF is also divided into four subclasses with different priority on how packets will be dropped in congestion state. UMTS Streaming could characterize to AF-class.

In UMTS, when UE starts the IMS-based call with another client, desired codecs and QoS are negotiated for the duration of the call with SDP (Session Description Protocol). The QoS information in SDP-messages is stored in P-CSCF's participating in call setup, and then forwarded to PCRF (Policy and Charging Rules Function) that forms IP QoS authorization data (also known as policy control rules). These rules include maximum bandwidth and QoS class for the negotiated codec. PCRF delivers this information to GGSN, which enforces the policy during PDP context activations. Depending on the bearer requirements, multiple PDP contexts can be opened for different kind of traffic with different QoS settings.

In order to receive Differentiated Services from one's ISP, the SLA (Service Level Agreement) usually needs to be created. When using dynamic SLAs, the resources need to be reserved with a signalling protocol, like RSVP (Resource reservation protocol). The successful implementation of an end-to-end QoS needs a support from every network the packet traverses and the mapping of QoS classes needs to be performed between networks.

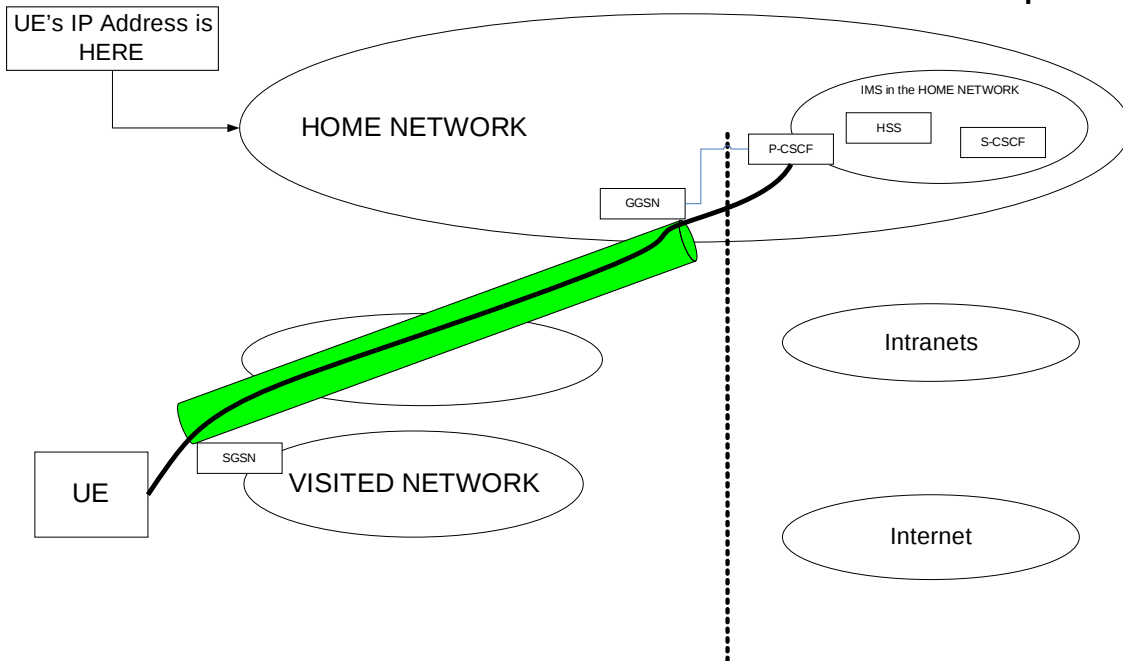
## 2.7 Roaming

In order to get IMS services successfully implemented, and more over in such an infrastructure like the one deployed within NETLAB, roaming is one of the major issues. If roaming is not properly handled, the implementation of the platform could be hindered.

In principle, IMS roaming means that Visited PLMN network resources are used to connect to an IMS core network residing in Home PLMN or Visited PLMN. In a roaming scenario, only P-CSCF could be located in the visited network (P-CSCF and GGSN are always in the same network); all other IMS core components are located, always, at the home network. Thus, the user is always registered with the S-CSCF of the home network for the purpose of charging and interfaces to the HSS, used for service provisioning needs.

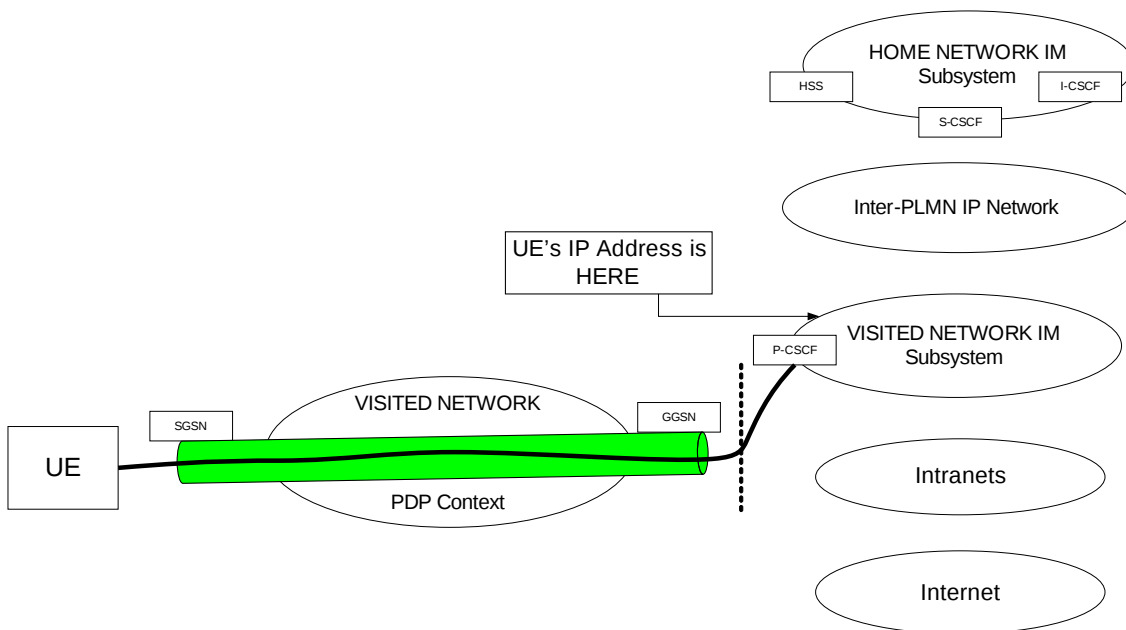
When an IMS user is located at his home GPRS/IMS network, he will normally access IMS services through a home network Access Point (GGSN). In a roaming scenario, the subscriber either uses a visited network Access Point or a home network Access Point. The home network can restrict the use of visited network Access Point.

In the usual scenario, the user would choose a home network Access Point to access IMS services (Figure 2). The visited network does not even have to have an IMS domain in this case.



**Figure 2: User selects home network**

In the other scenario, the roaming subscriber uses the visited IMS network (Figure ). In this case the IP address assigned to the IMS terminal is from the visited network-addressing domain. I-CSCF is a contact point between visited and home networks. In Figure 3, I-CSCF is only used in the home network. Thus, P-CSCF of the visited network discusses directly with I-CSCF of the home network.



**Figure 3: User selects visited network**

**Testing**

Even if the scenario, where the GGSN is in the home network, is easier to implement and is the preferred model, for the basic services to test in NETLAB both scenarios should be considered.

The interconnected architecture developed in NETLAB should provide users with roaming service capabilities, and interoperability capabilities in any case.

In the first phase, basic services will be tested in roaming scenarios:

- Registration in visited network
- Roaming call:
  - Call from a visited network to a user in a home network
  - Call from a visited network to a user in the same network
  - Call from a visited network to a user in a third network
- Register in presence service as a visitor
- View IPTV service from home network while roaming

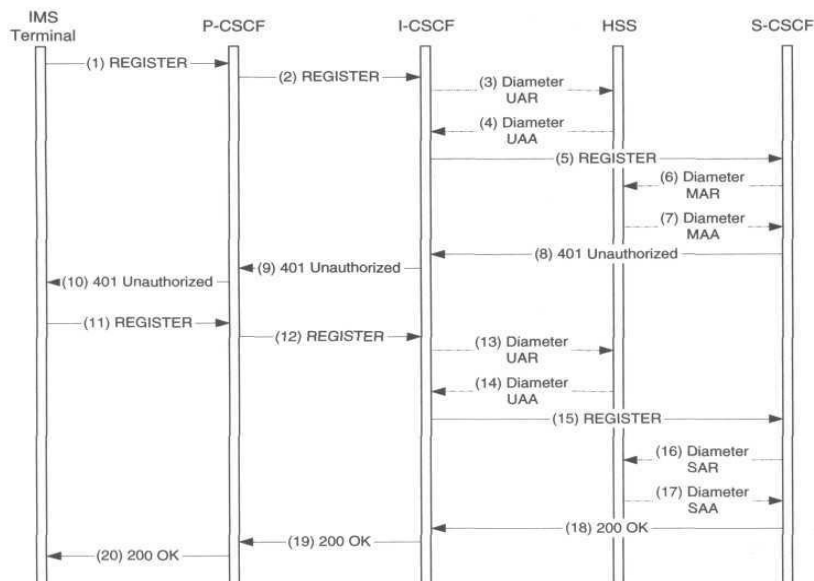
In all these basic test cases, both scenarios should be taken into account:

1. User uses the visited network's Access Point
2. User uses his home network's Access Point

When the roaming service within the NETLAB infrastructure is successfully proven for the basic cases, then the service should be tested for selected use cases, in which more complex application services will be involved.

## **2.8 Security**

Security requirements in an intra-domain connection focus on the relationship between the user equipment and the network. IMS achieves security through a registration that provides mutual authentication between the IMS subscriber and his home network. The authentication mechanism uses AKAv1-MD5 algorithm and SIP protocol to transport security parameters. For example, keys generated by the algorithm summarize a successful IMS registration (Figure 4).



**Figure 4: Successful IMS registration flow messages with no errors**

When S-CSCF receives user's register message, it can send a request to HSS for an Authentication Vector (AV), which is used for authenticating and agreeing a key with the user. An AV is composed of a random number RAND, an expected response XRES, a cipher key CK, an integrity key IK and an authentication token AUTN. CK and IK are used by UE and P-CSCF to establish secure IPsec/IKE associations, while RAND and AUTN are sent to the terminal. UE uses AUTN to authenticate the home network, computes a response and sends it back to P-CSCF. Upon receiving the UE response, S-CSCF checks it against XRES. If they match, then S-CSCF can authenticate the user. Using AV components, IMS registration is accomplished with the following steps:

- Integrity protection of SIP messages due to the security associations established between UE and P-CSCF
- User authenticates the network
- The network authenticates the user

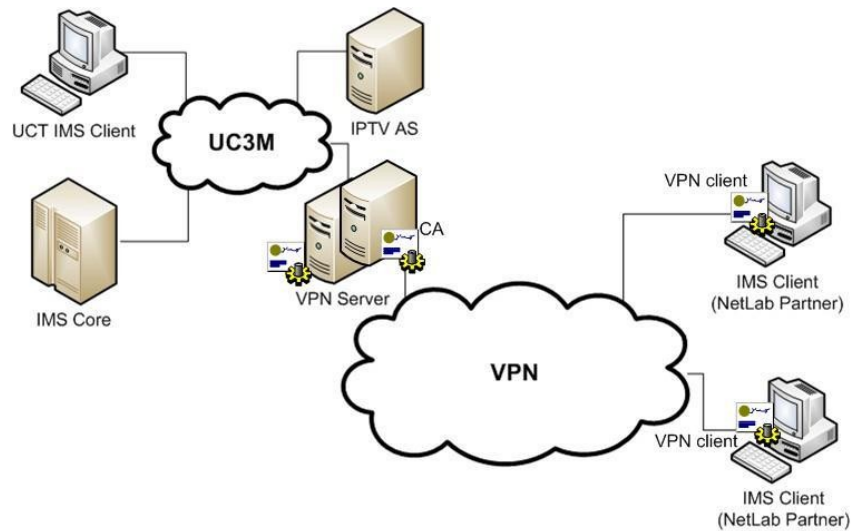
If the policy in P-CSCF requires confidentiality protection over the connection with UE, IPsec-ESP should be used. In this case UE sends the security parameter index (SPI) values and a pair of ports selected for the communication together with the REGISTER message to P-CSCF. Upon P-CSCF receives CK and IK from the S-CSCF, it selects the encryption algorithm from its own priority list and sends the SPIs and assigned ports to UE. UE then selects from the received list the first integrity and encryption algorithm combination which it also supports. Then two pairs of unidirectional security associations (SAs) are established between UE and P-CSCF. One SA pair is for traffic between a client port at UE and a server port at P-CSCF and the other SA pair is for traffic between a client port at P-CSCF and a server port at UE.

In roaming scenarios IMS registration process is essentially the same. UE sends a SIP REGISTER message to P-CSCF which is situated in the visited network. Visited P-CSCF is able to resolve partner operator domain due to established roaming agreement, so it forwards UE's message to home network I-CSCF and the authentication process follow as in the intra-domain scenario.

In order to provide integrity and confidentiality inside their networks, operators can use TLS (Transport Layer Security), which is supported by SIP proxies. TLS can also be used instead of or on top of IPsec to protect communications between CSCFs located in different partner networks.

**Network**

The NetLab architecture includes a Virtual Private Network (VPN), which has been implemented to solve NAT server problems. These kinds of servers, that are present between the core IMS network and the partners’ clients, affect the routing of SIP messages by modifying their headers. Figure shows the VPN server hosted by UC3M and some partners connected to it through VPN clients creating tunnelled connections. All IMS-related traffic is routed throughout these tunnels.



**Figure 5. VPN network deployment**

OpenVPN has been used to manage network interconnections. This software allows clients to authenticate themselves using certificates, username/password combination or a pre-shared secret. Certificate option was used in this network. OpenVPN server uses OpenSSL library to create a Certification Authority, which is used to issue a signed authentication certificate for each client containing client data and a public key. Every certificate is stored and passed to the client in PKCS#12 format together with the associated private key. By this way each client can authenticate itself with a signed certificate during the connection establishment. Once the VPN server has verified client certificate, it assigns an IP address in the VPN address network (10.8.0.0/24). Thus, IMS registration can take place as in direct connection with the P-CSCF.

### 3 IMS services

IMS services can be offered in a multi-platform environment. In the NetLab project, this environment is realized by using Symbian and J2ME. Security services are introduced here briefly, and as one example of an IMS service the Presence service is presented. Also, under discussion are new innovative services that could be implemented with IMS in the future.

#### 3.1 Multi-platform environment

The focus in the multi-platform environment is on testing compatibility between different IMS APIs and the IMS core in the registration process. Two IMS clients have been developed using two different programming environments: Symbian and J2ME.

*Symbian client:* Symbian version 2 and S60 third edition (feature pack 1) software are used. Test part is implemented using Nokia N95 mobile phone. To perform successful registration with the IMS core, some adaptations are made due to some differences compared with the standard SIP REGISTER message that implied errors in the registration phase. For example, the first header of the SIP REGISTER message, "request-line", includes Proxy port number along with the realm causing a strange behaviour in the IMS core. Another problem is a malformed *Authorization* header, that prevents the IMS core to correctly parse the SIP REGISTER message.

*J2ME client:* some Java programming environment, like SDK S60 3rd FP1 MIDP, and "Wireless Toolkit" are used. Test part is deployed using Nokia E61 and Nokia N95 mobile phones. As for the Symbian client, the SIP REGISTER message shows some differences compared with the standard message. Though, the adaptation used for Symbian client allowed the client to authenticate with the IMS core. Furthermore, J2ME uses SipConnectionNotifier class to manage SIP signalling. This API allow client to work with dedicated or shared mode connection. In the dedicated mode the application manages the registration process, while in the shared mode the system manages the whole registration process. In this test, neither Nokia E61 nor Nokia N95 allowed to use the dedicated mode.

#### 3.2 Security services

The IMS security deals with access security and network security. The access security defines how the users can securely access to IMS networks. The network security deals with protecting the traffic among network nodes (belonging to the same or different domains).

Access security requires mutual authentication between the subscriber and the network, and this is achieved via a security association between the user equipment (UE) and the Proxy-CSCF of the IMS network (either in the home or visited network). The security association follows the standard "Security Mechanism Agreement for the Session Initiation Protocol (SIP)" 4.2. Both client initiated and server initiated associations are allowed, together with different security mechanisms which are negotiated between the parties 4.2. The negotiated mechanisms can be: tls, digest, ipsec-ike, ipsec-man and ipsec-3gpp, though the ipsec-3gpp is preferred. The mutual authentication is achieved via the authentication and key agreement (AKA) (see 4.2). AKA is a challenging protocol, where the authentication centre in the serving network (the HSS in IMS) challenges the subscriber and passes the expected response (XRES) and a message authentication code (MAC) to the entry point in the network (the Proxy-CSCF in IMS). The P-CSCF sends the challenge to the UE, so that the UE can check that the other side has access to a long-lived secret stored both in the UE and in the HSS, which is required to generate RES and MAC, together with the challenge used by the HSS. There are several considerations on how this challenge is to be derived, reused, and personalized for different users at the HSS (as explained in 4.2) to avoid attacks, specially man-in-the-middle (MITM) attacks. Authorization occurs as registered users demand services: Serving-CSCF is in charge of that, together with the HSS, which logs user access to subscribed services for billing purposes. Besides

authentication and authorization, IMS offers confidentiality and integrity protection, by means of the security association between the UE and P-CSCF. In case IPsec is used, both nodes will share two keys for ciphering and for signing messages. Both mechanisms follow the standard "IP Encapsulating Security Payload (ESP)" 4.2. To achieve access security a secure storage is required at the UE. Ideally this should be provided by a smartcard at the UE running the IP-Multimedia Services Identity Module application (ISIM). The storage will include the private identity, one or more public identities (privacy protection means at IMS), the home network domain name, the authentication key, and some algorithms required by AKA and to ensure MITM protection in AKA, i.e. sequence number checking. Network security follows from using IPsec among the nodes with a variety of mechanisms, at least Triple-DES and HMAC-MD5 for the intra-domain case, and DES for the inter-domain case. The use of TLS is also supported.

The security services provided by network domain security in the IPsec profile are: data integrity, data origin authentication, anti-replay protection, confidentiality (optional), and limited protection against traffic flow analysis when confidentiality is applied. In the TLS profile only data integrity, data origin authentication and anti-replay protection is required.

### **3.3 IMS service example: Presence**

The presence service is one of the key services that the next-generation architectures, like the IMS, provide. Before the IMS networks were built and deployed, the presence service existed in ordinary Session Initiation Protocol (SIP) networks already. The presence service has been deployed for both legacy SIP and IMS networks in the NetLab project.

The presence service requires two main components:

- A SIP server to manage all presence publications, subscriptions and notifications
- An XML Document Management Server (XDMS) to manage authorization rules

These two components must be integrated to achieve a functional presence management service.

#### **3.3.1 Presence service within the SIP framework**

In the SIP framework, the proxy server passes presence related messages (SIP SUBSCRIBE and SIP PUBLISH) to the presence server.

In contrast to the presence server, which is usually reachable for the users only through the SIP core, the Extensible Configuration Access Protocol (XCAP) server advertises its interface directly towards the User Equipment (UE). The XCAP protocol allows clients to read, write and modify XML data on the server. In combination with a presence server, the XCAP server manages data, like buddy lists, presence policy and presence information and makes it accessible through the XCAP interface to the presence clients.

The presence server will store its data in its own database. Although it could be integrated with the SIP server database, this solution increases the independence from the SIP core and presence domain.

By default, the presence server does not require to authenticate the SIP messages. It can trust all the messages it receives from the SIP proxy, which is responsible for authentication of the messages. However, it is important that the XDMS has access to the authentication information (database) to authenticate direct user requests via HTTPS.

### **3.3.2 Presence service within the IMS framework**

The IMS follows the approach of a layered architecture. Each deployed service will be offered by one common transport layer, providing IP access to various fixed and mobile devices. This layer provides transparent access to all kinds of devices and allows the connection to the next layer: the control layer. The IMS interaction with services is triggered at that layer by the Serving-CSCF. The network operator has to set up Initial Filter Criteria (IFC) at the S-CSCF. Based on these criteria the controller can determine which messages it has to forward to which service.

All services reside on the third layer: the application layer. The presence management service, as one application on this layer, will store its data locally. The presence data of all IMS users will be stored in this database as well.

The XDMS is a separate component on the application layer. It will share the same database with the presence service. The XDMS, however, requires some authentication information of the IMS domain as well. As the IMS user data will be stored in a different format than the SIP user data, the presence server database must contain all credentials (SIP and IMS). In a future set-up, the XDMS will communicate with the HSS for authentication to allow the user to have a single username/password combination.

### **3.3.3 Architecture**

#### **3.3.3.1 SIP server**

For the presence server itself, the open-source SIP server implementation OpenSIPS version 1.5 is deployed. The open-source software is a fork of the well-known OpenSER. A MySQL server is used as the user profile storage.

The following changes in the OpenSIPS configuration are made:

- Handle subscriptions and publications
- "User authentication" via P-Asserted-Identity headers (for SIP and IMS to simplify the configuration)
- Accept only trusted sources as final packet source

The user authentication in the deployed IMS is made against the user database of the HSS. The presence server, however, has its own application database and hence a separate user and password management would be required, if the UE should be challenged. As the usage of two separate databases does not scale and more importantly, the IMS guidelines recommend another approach, the deployed solution makes use of the private extension headers (esp. P-Asserted-Identity).

The privacy header of the SUBSCRIBE and PUBLISH messages are checked by the presence server. Only if the P-Asserted-Identity header has been added by the SIP server and the S-CSCF, and the message has been received from this trusted node, the SIP message is processed. The privacy header username and domain are checked and treated as if those parameters would have been successfully challenged. The P-Asserted-Identity URI must match the SIP From: URI.

#### **3.3.3.2 XCAP server**

At the current state, the presence server would be able to handle all SIP subscriptions and publications properly. OpenSIPS allows using the presence server in a so-called "forced-authorization" mode. In this mode, all subscriptions are considered to be authorized. This is of course not intended: In real-

world scenarios, the users are responsible to authorize their presence states. The defined standard for those authorizations is XCAP.

The deployed XCAP server/XDMS is OpenXCAP from AG Projects. The software is open-source as well. It has been designed to work together with OpenSIPS. This is important because the XCAP server must communicate with the presence server.

The OpenXCAP server is an application written in Python and using most of the currently available standards. The Application Usage IDs (AUID), that the server supports, are:

- pidf-manipulation
- xcap-caps
- resource-lists
- pres-rules, org.openmobilealliance.pres-rules
- rls-services
- watchers (proprietary)

OpenXCAP uses the management interface from OpenSIPS to authorize users. The user authorization state is written in the database of OpenSIPS. For authorization, the presence server database is accessed directly. The XCAP data is also stored in the database. For authorization, OpenXCAP must modify the parameter according to which OpenSIPS decides whether the NOTIFY body will be filled with presence information or not. Instead of accessing the database directly for this purpose, the XCAP server uses the XML Remote Procedure Calls (XMLRPC) interface to authorize the users. The XCAP documents for these purposes, however, are stored via normal MySQL access in the database.

The SIP interface of OpenSIPS is exposed towards the SIP proxy/S-CSCF, and the interface of OpenXCAP is exposed towards the proxy server. It is responsible to use Network Address Translation (NAT) to make the respective port available throughout the internet. The DNS name for the XCAP server points towards the proxy from external requests and towards the XCAP server for internal requests.

### **3.4 Innovative services**

IMS has a flexible, modular, cost-effective infrastructure which can differentiate service offerings from competitors' non-IMS offerings by creating revenue-generating multimedia services that were previously cost-prohibitive or simply not possible. This allows to significantly ramp up the business flexibility and velocity—a bankable advantage in today's hyper-competitive environment.

IMS brings service creation into the pervasive realm of IT tools. IMS represents the main tool to create new innovative services. Voice and text message services have been traditionally the killer applications for operators. However, multimedia communication services are emerging rapidly. With the introduction of IMS, a lot of multimedia services are enabled, such as multimedia conferencing, multimedia messaging and Multimedia Ring Back Tone (MRBT).

Let us go through different services which could be provided by IMS operators:

- Voice telephony as the main application

## NETLAB – D2.2 IMS services requirements

This includes handing over voice calls between networks as the user roams out of coverage of a network. Furthermore, advanced IMS solutions will enable handovers of voice calls to a 2G network if no high speed 3G network is available anymore. The IMS enables video calls with the advantage over current 3G circuit switched mobile video calls, that the video stream can be added or dropped at any time during the session.

- Presence and instant messaging
- Multimedia sharing

Multimedia sharing includes photo, audio and video sharing. It is a new multimedia service that enriches 3G mobile users' communication by allowing them to share a photo, an audio clip, or a live camera view or a video clip while speaking on their mobile. It provides the users with an instant way of adding a visual element into a phone conversation. During the conversation, sharing can be initiated or terminated either by the calling party or called party. When the conversation is over, this feature is closed as well.

- Multimedia conference

Multimedia conference can be divided into three kinds of conferences: audio conference, video conference and data conference.

Data Conference is composed of electronic whiteboard, application sharing and file transfer in order to assist collaborative work.

Some of the abundant features supported by Multimedia Conference are listed below:

- Conference can be an instant conference or a reserved one. Notification of a conference can be sent to the members through SMS or voice call.
  - Conference chairman can manage the conference in real-time on a GUI interface.
  - Sub-conferences can be created within a conference and participants can be moved from and into a sub-conference by the chairman.
  - Conference members' rights can be modified. For instance, speaking rights can be withdrawn or granted to a member.
  - Conference can be set to be recorded at the time of reservation. Later it can be listened by dialling specific service code, conference number, and entering password.
  - Voice and video session conferencing with three or more parties.
- Session transfer
- A session can be moved from one device to another while it is ongoing. A video call, for example, might be accepted on a mobile device but transferred to the home entertainment system when the user arrives at home. Transferring the session also implies a modification of the session parameters. While a low resolution video stream is used for a mobile device, the resolution can be increased for the big screen of the home entertainment system if this is supported by the device at the other end.
- Identity management

*One identity / telephone number for all devices of a user.* A session is delivered to all or some devices based on their capability. A video call would only be delivered to the user's registered devices that are capable of receiving video. Sessions can also be automatically modified if devices do not support video.

## NETLAB – D2.2 IMS services requirements

*Use of several user identities per device.* This allows only using a single device or a single set of devices to be reached by friends and business partners. With user profiles in the network, incoming session requests can be managed on a per user identity basis. This way, business calls could be automatically redirected to the voice mail system at certain times or to a colleague while the user is on vacation while private session requests are still connected.

- Live audio and videocasts of events

The difference to current solutions is the integrated adaptation to the capabilities of the device.

- Push message & video services

These could include, for example, sending messages to subscribers when their favourite football team has scored a goal, when something exciting has happened during a sports event, and so on.

- Calendar synchronization among all IMS devices.
- Unified voice and video mail

Because all devices used by a person are subscribed to the same IMS account.

### 3.5 IPTV specific services

The initial opportunities for service convergence through IMS include adding voice and data functionality to IPTV video and extending IPTV functionality to mobile devices. Beyond this, there is the possibility of extending core service enablers to create a unified user experience across devices. Here are described some typical opportunities that are possible with currently available IP and IMS technologies, and a set-top-box or card for the user's television.

#### 3.5.1 Extend voice services into the TV environment

While watching TV, the user receives an online prompt showing the caller ID of an incoming call. The phone does not ring (a user-selectable setting), so nobody else in the household is disturbed. The subscriber can choose to either:

- Accept the call — The phone rings and the call can be answered on the home phone or a speakerphone associated with the TV.
- Reject the call — The phone never rings and the call is discarded.
- Forward to voice mail — The caller can leave a voice mail message, and the subscriber sees a message-waiting indicator on the TV screen. The subscriber does not have to get up to see who is calling, which is particularly welcome if the calling party is not of interest. Who wants to interrupt a favourite TV show to take a message for another household member, when voice mail can do just as well or better? The service also supports click-to-call capability, whereby a subscriber can place a call using the remote control, either from an address book or a list of received calls. In the future, subscribers will also be able to place and receive video calls using a TV-mounted webcam.

### **3.5.2 Extend data services into the TV environment**

Data integration can also take several forms. For example, subscribers could exchange instant messages (IMs) with others on their personal buddy lists, while watching a TV show. Thanks to “presence” capabilities, the system knows whether the subscriber’s buddy is watching the IPTV service, and if so, sets up the IM connection for them to share a back-and-forth chat while watching TV. The two subscribers can then share the viewing experience, trading comments about what they’re watching, even though they are in different places.

Picture-sharing is another popular option. Suppose you want to share digital photos with a distant friend or colleague who does not have a PC or Internet connection. You could send the photos to the service provider’s photo exchange service, which in turn uploads them to the recipient’s set-top-box. That person receives a notification that pictures are available for viewing, and selects “slideshow” on the TV remote control to view the photographs on the TV screen.

### **3.5.3 Extend TV services into the wireless environment**

A mobile device can become an extension of the IPTV service — both to control and to view video content. For example, suppose a change in your flight schedule will cause you to miss tonight’s episode of your favourite TV show. You can use your PC, PDA, cell phone or other wireless device to pull up the TV schedule, select the episode and send a command to the digital video recording (DVR) service to record this episode. Later, you can use your video-enabled mobile device to call up the DVR menu, select the pre-recorded program and watch it wherever you are.

Conversely, you could begin watching the TV show at home and then switch seamlessly to the mobile video device to watch the rest of the program on your trip. The user interface will be the same on the mobile device as on the home IPTV set, making navigation familiar and convenient.

### **3.5.4 Establish consistent user experience across access media and devices**

The unified service experience is made possible by core service enablers, such as:

- Presence — Subscribers can see if someone on their buddy list is online, whether that person is on a mobile device, a PC or watching their IPTV.
- Network buddy list — This is the subscriber’s circle of friends for sharing interactive services. The buddy list would be the same on the PC, mobile device or IPTV.
- Single sign-on — Users can log onto a service using one device and their sessions can be continued in another device without having to sign in again. For example, when a subscriber signs on and accesses IPTV from the TV set, there is no need to sign in again when switching over to watch the rest of the show on a mobile device. The service is seamless between devices.

## **4 IMS testbeds**

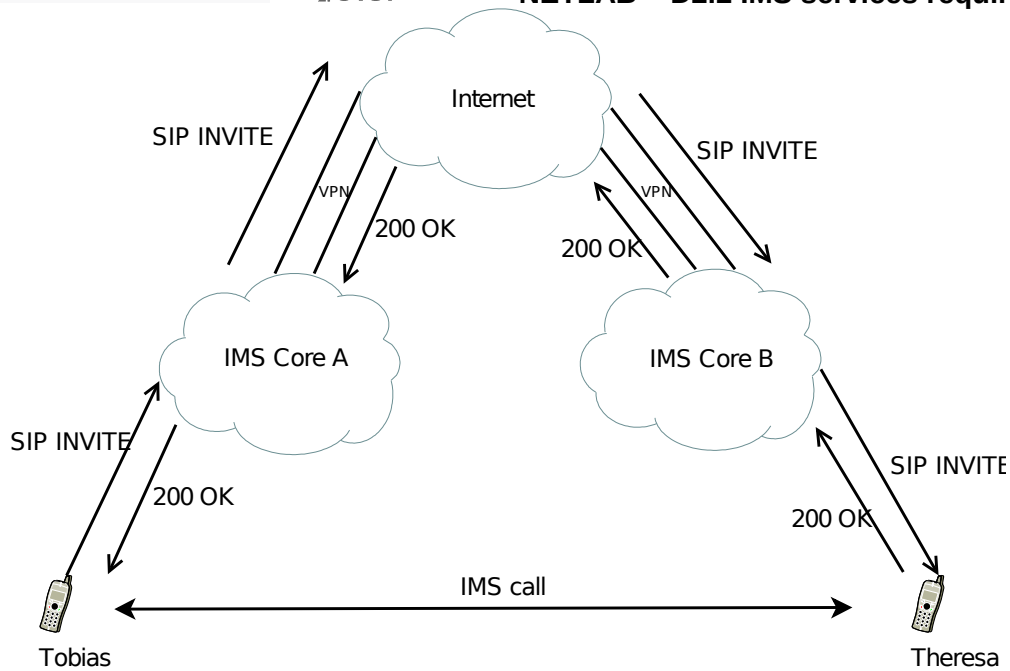
The IMS architecture is designed to deliver applications to subscribers on any access network through a common IMS core network. IMS's major promise, and the network operators' major goal, is the access agnostic, service agnostic, and location agnostic core network. Interoperability is a major factor in achieving this goal. In the NetLab project the interoperability of the different IMS cores and services is researched. Some of the interoperability issues between different IMS testbeds will be discussed on high level. As some problems have been discovered, a proposal for new mechanisms is also considered.

### **4.1 Solution for service interworking and roaming between testbeds**

The interconnection of IMS cores will be implemented in two phases in the NetLab architecture. First, IMS cores will be connected through the public Internet, using OpenVPN. In the second phase, NetLab partners will use GEANT network for higher bandwidth. After basic connectivity between IMS cores is established and proved functional, there arises the question of service interworking between these IMS cores. No matter how the IMS cores are interconnected, DNS (Domain Name System) acts a great role when it comes to routing traffic between IMS cores.

Let's consider the following scenario. Two IMS cores are connected using a VPN through the public Internet as shown in Figure . There are two IMS subscribers, Tobias and Theresa. Theresa is using her home IMS network and Tobias is using another IMS network. Tobias wants to place a call for Theresa who is registered at her home IMS network. Tobias uses his IMS client software to send an SIP INVITE request to her address: `theresa@home.com`. There are three steps in the call flow:

1. IMS routing of initial SIP INVITE
2. IMS routing of first response to the INVITE
3. Audio path setup



**Figure 6: IMS connection**

When the call is made, SIP INVITE first arrives to the originating P-CSCF (see Figure 6), which notices that the called address is `theresa@home.com` and does not belong to the call originating network. It queries DNS to resolve the IP address of the terminating network I-CSCF, which is the entry point to the network Theresa is using. After the IP address of the terminating I-CSCF is resolved, SIP INVITE is forwarded to this node. Next, the S-CSCF and P-CSCF that Theresa is currently using needs to be identified. This is done with the help of HSS. These addresses were saved to HSS when Theresa registered her IMS client. After IP addresses of S-CSCF and P-CSCF are fetched from HSS, the SIP INVITE message can be routed to Theresa and the call is established. It should be noted that this signalling example is greatly simplified. For example, all HSS related DIAMETER signalling is excluded in this example.

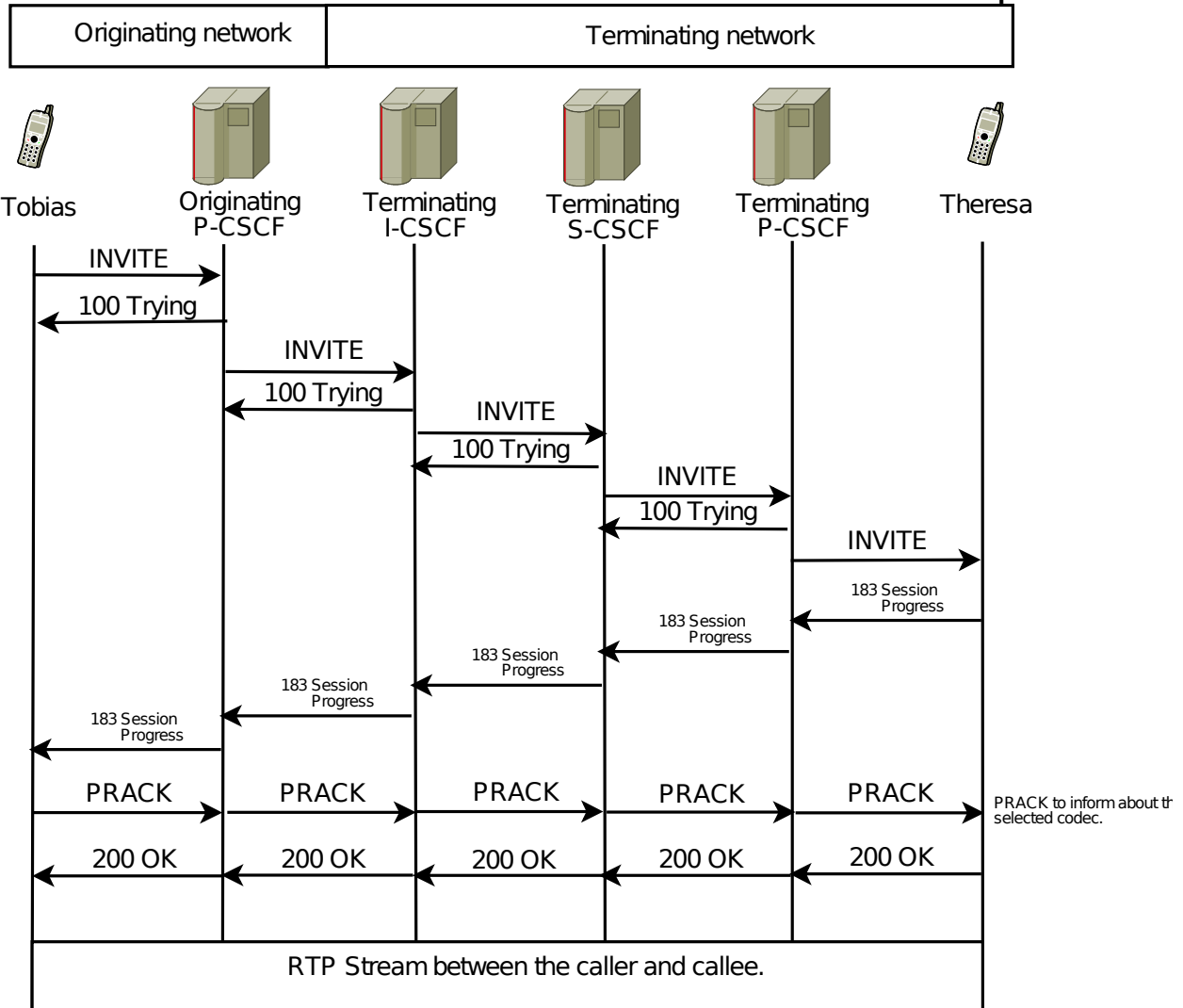


Figure 7: IMS call

As mentioned earlier, DNS acts a great role when it comes to routing SIP messages. User equipment and CSCFs cannot route directly using FQDN's (Fully Qualified Domain Name), instead they need to use DNS to resolve the relevant IP addresses. When IMS client is turned on, it registers to the network and uses DNS to find IP addresses relevant for its use. There are three types of queries that UEs and CSCFs mainly use: NAPTR, SRV and A. The NAPTR (Naming Authority Pointer Resolving) query is used to find out more information about the domain where the message should be routed to. For example:

*DNS query – NAPTR*

*Name: visited.com*

DNS would respond with a list of services in the domain:

*DNS response – home.com*

*IN NAPTR 100 10 "s" "E2U+sip" " sip\_udp.visited.com*

*IN NAPTR 100 10 "s" "E2T+sip" " sip\_tcp.visited.com*

Let's assume Tobias' IMS client decides to use UDP as transport protocol. It needs to resolve the indicated service name further via DNS and this is done by sending DNS SRV (service) query to the local DNS server. For example:

*DNS query – SRV*

*Name: \_sip.\_udp.visited.com*

DNS server returns host names of all SIP servers in visited.com domain that support desired service.

*DNS response - \_sip.\_udp.visited.com*

*IN SRV 0 0 5060 pscsf.visited.fi*

Now the IMS client knows FQDN of the P-CSCF and its port. Finally, IMS client needs to find out the IP address of the P-CSCF in use. This is done by making A query (AAAA query if IPv6 is in use) to the local DNS server. For example:

*DNS query – A*

*Name: pscsf.visited.com*

DNS server would respond as follows:

*DNS response – pscsf.visited.com*

*IN A 10.1.10.1*

## **4.2 Proposal for new authentication mechanisms**

The authentication mechanism has been identified as a critical task in the IMS standard. Standard registration implies two authentication processes: one with the access network and other with IMS network. As 3GPP enforces some requirements for access network authentication that involves the use of AKA, the final registration process yields to 4 Round Trip Time (2 RTT for AKA with access network and 2 RTT for IMS registration). This double authentication leads to a very time-consuming overhead, especially when the user is roaming and messages are forwarded from the visited network to the home network. Moreover, the adoption of AKA requires implementing EAP-AKA (Extensible Authentication Protocol Method for [UMTS](#) Authentication and Key Agreement) for every access network technology. A solution to overcome these problems has been tried to find.

The idea behind the solution is to reuse legacy authentication protocols intended for other purposes, creating a tunnel that authenticates the Network Access Server (NAS) before starting IMS authentication mechanism. Once the NAS is correctly authenticated, it will forward authentication messages to a back-end authentication service (typically HSS). In this way, an authentication protocol inside the tunnel can be reused in a secure fashion since it is executed over a tunnel between client and NAS. Furthermore, this tunnelled protocol helps to avoid the problem of having implemented authentication protocols in L2, by providing an alternative transport over a higher protocol (Protocol for carrying Authentication for Network Access, PANA).

The objectives of the proposed registration framework are to reduce the registration time, provide a general authentication mechanism for any upcoming technology, prevent attacks, fulfil the 3GPP requirements and maintain backwards compatibility. Two scenarios are defined where the protocol could be used. These scenarios are presented in Appendix A.

## References

- [1] IP Multimedia Subsystem – Wikipedia, the free encyclopedia. Available: [http://en.wikipedia.org/wiki/IP\\_Multimedia\\_Subsystem](http://en.wikipedia.org/wiki/IP_Multimedia_Subsystem). Search date: 15.4.2010.
- [2] Poikselkä, Miikka – Mayer, Georg 2009. The IMS: IP Multimedia Concepts and Services, Third Edition. A John Wiley and Sons, Ltd., Publication.
- [3] 3GPP Releases. Available: <http://www.3gpp.org/releases>. Search date: 15.4.2010.
- [4] Ericsson AB, 2004. IMS – IP Multimedia Subsystem – The value of using the IMS architecture.
- [5] IETF, RFC 3329: “Security Mechanism Agreement for the Session Initiation Protocol (SIP)”, 2003
- [6] IETF, RFC 3310: “Hypertext Transfer Protocol (HTTP) Digest Authentication Using Authentication and Key Agreement (AKA)”, 2002
- [7] IETF, RFC 2406: "IP Encapsulating Security Payload (ESP)", 1998

## Appendix A: Proposal of enhanced authentication mechanism

### 1 Overview

The following sections summarize the work done in NetLab related to improving IMS registration authentication mechanism. We start by (1) a brief description of the standard authentication and (2) the present proposal stating the improvement to the process.

1. Standard IMS registration is based on HTTP Digest Authentication [1], using AKAv1-MD5 [2] as algorithm, which requires exchanging four messages (2 Round Trip Time – RTT) between the subscriber, at the visited network, and the subscriber's home network. Before that the user has to register with the access network in order to obtain connectivity. Both access network authentication and IMS registration involve an exchange of challenge-response messages. This double authentication process leads to a very time consuming overhead, especially when the user connects from a visited network (Roaming scenario).
2. To overcome the problem, we present a general IMS registration protocol that accelerates the process, while authenticating the user, the core and the access network, thus preventing man-in-the-middle (MiTM) attacks by cryptographically relating access network authentication to IMS registration [3].

The main idea is to use a token to relate both processes in order to save up to 3RTTs. Once the user gains access to the network, both user and NAS derive a token. The token is sent to the HSS by the NAS. The token is used in the initial registration sent by the user to the IMS network. Besides the secret shared among HSS and user, the S-CSCF checks if the user token and the HSS token are equal to authenticates user and access network.

## 2 Architecture of the proposal

IP Multimedia Subsystem is based on a common control plane using the Session Initiation Protocol (SIP) [4] over internet protocol (IP). One of the core elements of the control layer of IMS is the called Call Session Control Function (CSCF) Server. IMS defines three CSCF: a Proxy-CSCF, a SIP proxy in charge of redirecting the messages to the correct Home Domain of the UE. A Serving CSCF in charge of authenticating the users and providing service to them and an Interrogating-CSCF, in charge of locating the S-CSCF in the Home Domain which manages the registration process. Another important element is the Home Subscriber Server (HSS) that stores the user profile of each end user. This profile can include user’s IP address, telephone records, buddy list and so on.

In order to allow the easy deployment of the IMS enhanced authentication protocol we use the same framework elements of IMS core network.

The access network authentication is performed using the Protocol for carrying Authentication for Network Access (PANA) [5] in conjunction with Extensible Authentication Protocol (EAP) that derives keys using the Transport Layer Security method (TLS) [6][7] to protect the channel. The election of the PANA protocol is due to its properties: it supports various authentication methods, it is suitable for roaming users, and it is independent from the link layer mechanisms.

Moreover we use Keying Material Exporters for Transport Layer Security [8] in order to generate a cryptographic proof of the message exchange. Using this proof is possible to relate access network authentication with IMS registration.

The following figure depicts our proposal architecture.

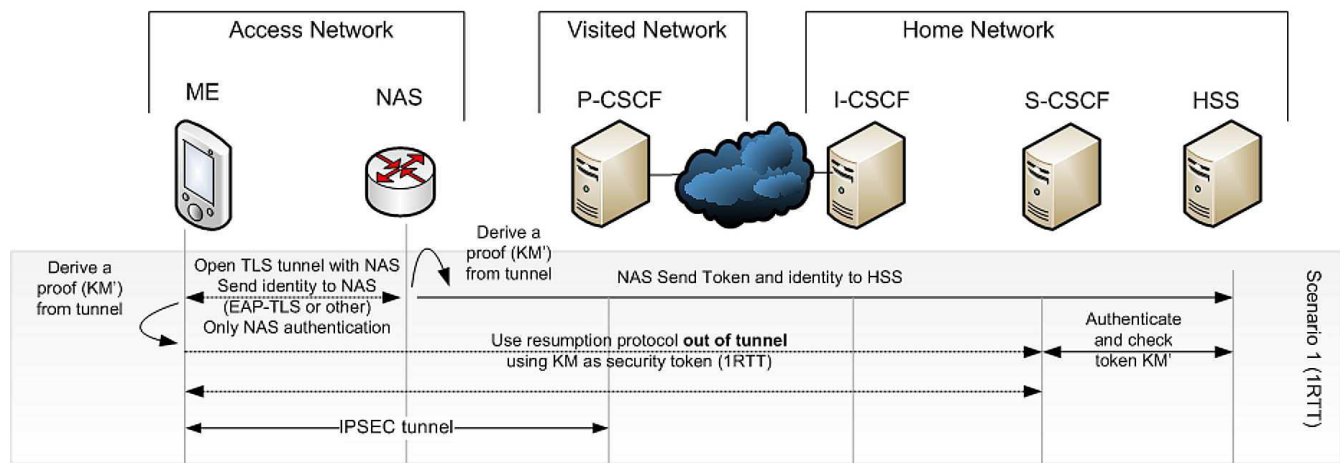


Figure 1: Proposed protocol with cryptographic relation to access network authentication

### 3 Implementation details

We used an open source implementation of IMS core called OpenIMSCore (FOKUS Institute, German) to simulate an IMS network.

The IMS client used is the UCT IMS Client 1.0.12 (University of Cape Town, South Africa).

Finally, to deploy the proposed protocol, we had to implement the PANA EAP-TLS protocol to use in the access network authentication. Thus the three base elements of the PANA protocol architecture had to be defined: the PANA client (Pac), the PANA Authentication Agent (Paa) and The Authentication, Authorization and Accounting Server (AAA).

Next sections describe the decision and modification we have taken in order to implement our proposal including some details of the PANA EAP-TLS software we used.

#### 3.1 PANA EAP-TLS Implementation

After dealing with other implementations, we ended with a BSD open source implementation of the protocol [10] written in C. The software implements the Pac, Paa and a local eap-tls or AAA server and it supports only IP version 6. EAP-TLS part was implemented using OpenSSL (> 0.9.7c).

##### 3.1.1 System Requirements

FreeBSD 4.9 or more for the Pana server.

FreeBSD 4.9 or more, Linux for the Pana client.

IPv4 is not supported, IPv6 only.

Libraries to install:

- ISC library, only on the server. (Clients tree management)
- XML2 library. (to parse the configuration file)
- Pthread library
- OpenSSL (> 0.9.7c)
- ip6fw utility, on the server.(if you want to configure a firewall)
- curl library

##### 3.1.2 Supported functionality and general description of the existing code

This version of the software does not support all options of PANA and EAP-TLS protocol. The based documents used for this implementation are draft-ietf-pana-pana-06.txt and RFC 2716.

The following features of the PANA protocol are not supported:

- Sending EAP-Payload in PSR/PSA messages

- Mode without security association. i.e. After each new authentication, a new PANA SA is established between the PAC and the PAA
- Separated authentication NAP-ISP
- Mobility and context transfer
- Authentication without "Piggy-backing"

Concerning EAP-TLS protocol, only the required (minimal set) EAP messages are implemented:

- EAP-Request
- EAP-Response
- EAP-Success
- EAP-Failure

The configuration of the implementation is done using XML files. In this way is possible to set IP direction, listening port and TLS parameters in order to perform a TLS authentication. In the Paa there is also the diameter server field that indicates the eap-tls server to use for the authentication or, in case of use the local server, the local socket to use for the communication.

The code is structured as follow:

### ***Pac***

It uses the PANA protocol over UDP to communicate with the Paa.

Usage: panac -p pana\_configuration\_file -e eap\_configuration\_file

Main functions are:

```
int load_configuration (char * file)
```

To parse the XML configuration file corresponding to a client configuration and to set up the variables used by the pana client. It returns 0 if there aren't errors.

```
int read_pana_hdr (struct pana_hdr *h, char * buffer)
```

To read the buffer and create a pana header fields. It returns 0 if there aren't errors

```
int raw_to_pana (char * buffer)
```

```
int pana_to_raw (struct message *msg, char * buffet)
```

The first one converts the buffer received in a pana message and the second one converts a pana message in a buffer. They return 0 if there aren't errors.

```
int process_PSR(struct message *psr)
```

```
int process_PAR(struct message * par)
```

```
int process_PBR(struct message *pbr)
```

```
int process_PER(struct message *per)
int process_PTR(struct message *ptr)
int process_PTA(struct message *pta)
int process_PPR(struct message *ppr)
int process_PSR(struct message *psr)
int process_PPA(struct message *ppa)
int process_PRAA(struct message *praa).
```

Functions for message parsing a message and sending the corresponding answer or Pana error message.

EAP-TLS functions are defined in order to create the EAP payload to carry out the TLS authentication.

Most important functions are:

```
int load_eap_configuration(char*)
```

Function that loads and parse the XML eap configuration file.

```
int eap_build_auth_response(EAP_DS *)
```

Function that builds the response frame for the authentication.

```
Struct key_t* SSL_tls1_key_extractor_EAP(SSL * s)
```

Function to generate the proof of eap-tls authentication (extracting it from the TLS master key).

### ***Paa***

The main role of the Paa is translating the pana messages into diameter messages, and forwarding them to the EAP-TLS server and the other way round. Therefore it is a gateway, capable of understanding both diameter and pana protocols. Paa is threaded: threads are created to handle pana socket and pana messages, to handle aaa socket and aaa messages (when external AAA is used), and to process eap request (when local AAA server is used).

Usage: pana\_server -v (verbose) -f configuration\_file

Main functions are:

```
int load_configuration (char * file)
```

XML configuration file parsing function.

```
void * PANA_message_handler(void * param)
```

This function allows to accept connections from the pana client and to handle the pana signaling flow.

```
void * process_pana_message(void * param)
```

Function that processes the pana messages received.

```
void * AAA_message_handler(void * param)
```

Function that deals with diameter messages received from AAA server.

```
void * process_AAA_message(void * param)
```

Function that processes the diameter messages when external AAA is used.

```
void * EAP_message_handler(void * param)
```

Function that deals with diameter messages when the local eap-tls server is used.

```
void * process_EAP_request(void * param)
```

Function that processes the diameter messages when a local socket is used.

### ***Eap tls server***

It is the server in charge of authenticates the client. It is a multi-process module and there is one opened socket per client.

Usage: eap\_server -f configuration\_file.

Main functions are:

```
void eaptls_operation(EAP_TLS_PACKET*, eaptls_status_t, EAP_HANDLER*)
```

It processes the TLS incoming data.

```
int load_configuration (char * file)
```

Function in charge of parsing the XML configuration file.

```
int eaptls_start(EAP_DS * eap_ds)
```

To compose an eap-tls start message.

```
int eaptls_success(EAP_DS * eap_ds)
```

To compose an eap-tls success message.

```
int eaptls_fail(EAP_DS * eap_ds)
```

To composes an eap-tls fail message.

```
int eaptls_request(EAP_DS * eap_ds, tls_session_t * ssn)
```

To compose an eap-tls request message.

```
int eaptls_send_ack(EAP_DS * eap_ds)
```

To composes an eap-tls ack message.

Moreover there are TLS specific functions to handle the TLS handshake and to generate the proof when authentication succeeds.

The most important functions are:

```
tls_session_t * new_tls_session(eap_tls_t *eaptls)
```

It creates a new tls session.

```
int tls_handshake_rcv(tls_session_t *ssn)
```

It handles the received data during the handshake (dirty to clean data)

```
int tls_handshake_send(tls_session_t *ssn);
```

It makes data ready to be send (clean to dirty data)

```
struct key_t * SSL_tls1_key_extractor_EAP(SSL *s);
```

Function in charge of generating the proof of eap-tls authentication.

### 3.1.3 PANA software modifications

Our proposal is intended to work on Linux platform (Ubuntu 8.04) and IPv4. Therefore we made some changes to the code to provide the requested functionality. More changes were done once the code fulfilled the requirements due to some implementation error that avoided the correct compilation of the code.

Once the code compiled we first test the implementation in order to verify the correct signaling message flow among parts.

Following picture depicts signaling flow between PaC, Paa and AAA.

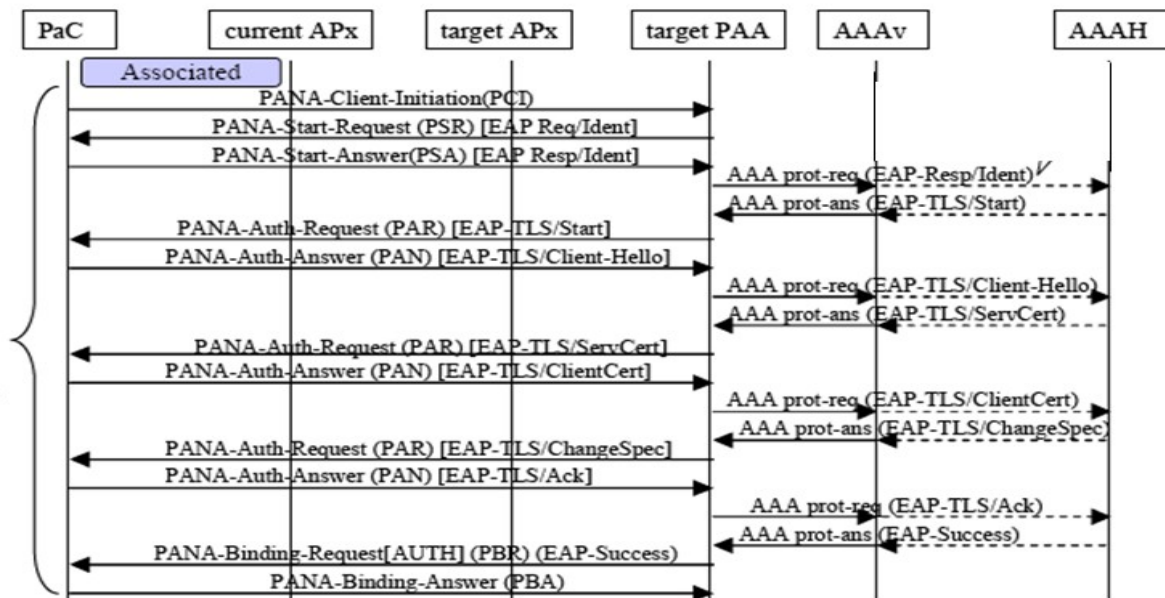


Figure 2: PANA EAP-TLS message flow

Once we verified that the code worked correctly we started to modify it in order to implement the proposal.

### ***Pac modifications***

The pana client has to work in conjunction with the IMS client in order to perform the access network authentication using PANA EAP-TLS. Hence it has to be integrated in every IMS client to allow this functionality. We decided to compile the Pac as a dynamic library; in this way we make easy the integration with any client implementation that would use our protocol.

The modifications required are the following.

Change the main function.

```
char * main_client(int argc, char argv[], char * user_id)
```

*Argc* and *argv[]* are used to get the XML configuration files as parameters. The new parameter *user\_id* is the IMS private identity of the user that will be authenticated.

The function returns the proof extracted (as a string) when the authentication ends, namely after the client processes the PBR message (PANA\_BIND state).

Change the function to derive the token after the authentication. Three new functions have been implemented to extract the key material.

```
struct key_t * SSL_tls1_key_extractor_EAP(SSL *s);  
  
void tls1_PRf_EAP(const EVP_MD *md5, const EVP_MD *sha1,  
unsigned char *label, int label_len, const unsigned char *sec, int  
slen, unsigned char * out1, unsigned char *out2, int olen);  
  
void tls1_P_hash_EAP(const EVP_MD *evp_md, const unsigned char*  
secret, int secret_len, const unsigned char *seed, int seed_len,  
unsigned char *out, int out_len);
```

The function *SSL\_tls1\_key\_extractor\_EAP* implements the Keying Material Exporters for Transport Layer Security. It extracts the connection related random material from the SSL object (client and server random numbers and the master secret) and defines the label and context ASCII strings used in the computation of the cryptographic proof of eap-tls authentication. UC3M defines label and context as two fixed strings:

1. Label: "EXPERIMENTAL\_IMS\_RESUMPTION"
2. Context: "CONTEXT-ASSOCIATION"

Label value begins with "EXPERIMENTAL" for private use without IANA registration.

*SSL\_tls1\_key\_extractor\_EAP* composes label and context in one string element that *tls1\_PRf\_EAP* and *tls1\_P\_hash\_EAP* functions use to compute the proof value. The derivation of the proof is based on the use of the HMAC set of functions, which are keyed hash functions used for message authentication, provided by the OpenSSL library. The combination of md5 and sha1 hash functions in the use of HMAC set provide the univocally association between the EAP-TLS authentication and the cryptographic proof used in the IMS authentication.

Change the makefile in order to create a dynamic library.

### ***Paa modifications***

Paa has to be able to connect with the HSS of the IMS core in order to send to it the token extracted and the identity related to the proof.

Therefore we modified the XML configuration file to add a new field indicating the HSS servername to send the token:

```
<hss_server>HSS Server Name</hss_server>
```

We also defined new functions:

```
void * AAA_message_handler_HSS(void *)  
  
int start_attendant_HSS()
```

It returns the socket identifier of the connection Paa – HSS.

The first function is called when the binding is performed among the Pac and the Paa, namely when the Paa has received the PBA message from the client.

The call is as follows:

```
pthread_create(&AAA_handler_thread_HSS, NULL, AAA_message_handler_HSS,  
(void*)msg);
```

In this way we use a new thread to handle the communication with the HSS.

Within this function, a call to the *start\_attendant\_HSS* is done, in order to open a socket among the Paa and the HSS. The socket is used to exchange the capability request and an answer message (CER and CEA) to allow a diameter communication. To carry out a successful exchange-capability a new Attribute – Value Pair (AVP) was defined using a new function:

```
struct AAA_avp* build_vendor_specific_application_id_avp(int  
vendor_Id, int auth_application_Id)
```

This AVP was added to the CER message sent by the Paa.

Once the connection was established (the socket identifier returned is greater than zero), next step was to create a diameter message which includes the extracted token and the identity of the authenticated user.

The build \_Token function:

```
struct diameter_message * build_Token(struct client* client)
```

is in charge of creating a new diameter EAP message (number 987) with the username AVP and the new token AVP ( AVP number 100). The second AVP was created using the general function:

```
void create_AAA_AVP(struct AAA_avp** avp, int type, uint8_t flags,  
void *value, int size)
```

that allows the creation of non-standard AVPs.

### ***Eap-tls server modifications.***

We have modified the function to extract the proof after the authentication in the same way we did for the pana client.

## 3.2 OpenIMSCore modifications

The Open IMS Core is an Open Source implementation of IMS Call Session Control Functions (CSCFs) and a lightweight Home Subscriber Server (HSS), which together form the core elements of all IMS/NGN architectures as specified today within 3GPP, 3GPP2 and ETSI TISPAN. The four components are all based upon Open Source software (the SIP Express Router (SER) and MySQL).

As described in [3], a second use case was proposed. It defines a second way to use our proposal where the HSS acts as authenticator for the access network. Furthermore we want to deploy an external AAA to perform the PANA EAP-TLS authentication; in this way AAA y Paa are not forced to work in the same machine and further scenarios could be defined.

Hence we have provided an eap-tls interface to the HSS. Moreover changes were done in order to include the access network token to the information returned to the SCSCF in order to deploy our protocol.

### 3.2.1 HSS

We have defined a new TLS package and within this the main class *DER* to perform the eap-tls authentication.

The principal function used by the *DER* class is:

```
addEapPayloadTls(DiameterMessage response, DiameterMessage request,  
Session session)
```

defined in the *UtilAVP* class.

The first parameter is the message that contains the answer to the received message, the request parameter is the received message and the last parameter is used to handle the database session. The function is in charge of reads and parses the eap payload of the diameter message received and creates the new eap payload to add to the response.

Parse operation is done using some more function that helps to correctly analyze the eap payload.

In order to perform the TLS handshake we use an external TLS server written in C that communicates with the HSS (the HSS doesn't have TLS capabilities, and rapid prototyping is the reason why we deployed our prototype with an external server).

The class in charge of the communication with this server is the *javaClient* class. Once the eap payload is parsed we extract the TLS message and we send it to the TLS server that parses and creates the corresponding response.

Moreover we have to provide the functions to parse the diameter message containing the token extracted and to store the information in the database.

The class *SKEY* defines functions to parse the message. We get the username and the token related to it with the functions:

```
public static string getUsername(DiameterMessage message)  
  
public static string getProofKey(DiameterMessage message)
```

The data are stored and retrieved using a new table named *eap\_tls*:

Impi (private identity)	proof

The classes to manage it are: *eap\_tls* and *EAPTLS\_DAO*.

The first class is in charge of defines the object and the method to set or get information from the table. The second class defines methods that implement the required queries.

We have defined three functions:

```
public static void insert(Session session, String proof, String impi)
```

Function in charge of fill the table.

```
public static String getProofByIdentity(Session session, String impi)
```

This function is in charge of retrieve the proof for a specific private identity.

```
public static void updateproof(Session, String proof, String impi)
```

This function is in charge of update the proof for a specific private identity.

With these changes we were able to perform a PANA EAP-TLS authentication using the HSS as AAA. Moreover with little changes in the code we made possible to use the HSS for both the use case 2 and use case 1.

To deploy the protocol the HSS has to be able to send in the Multimedia Authentication Answer (MAA), along with the authentication vector (AV), the token related with the identity to authenticate in the IMS network.

Hence the MAR class was modified in order to add another field to the AV included in the MAA message. Further details on the MAR class modifications can be provided upon request.

### 3.2.2 SCSCF

The SCSCF is the entity in charge of retrieves the AV from the HSS to authenticate the IMS user. It create a Multimedia Authentication Message (MAR) asking for the AV and creates a challenge in response to the first IMS register message (standard protocol).

To implement our proposal we defined a new module called “dumb\_mod”, used in the scscf.

The function in charge of handle our authentication is:

```
static int Dumb_cmd_2(struct sip_msg *rq, char * p1, char * p2)
```

The first parameter is the SIP message to check. The others two parameter are unused. The function gets the register message and checks the authenticated identity body (aib) [10]. If the verification was successful, it calls the function:

```
int S_challenge (struct sip_msg * msg, char * str1, char * str2)
```

The first parameter is the sip message, the second is the realm and the third is unused.

The function creates a challenge to the register message and it returns a success for our authentication (value -2) if the function:

```
int pack_challenge (struct sip_msg, str realm, auth_vector * av)
```

returns success for our authentication (value 2).

The parameters of *pack\_challenge* are the SIP message, the realm and the authentication vector. This function is in charge of adding the WWW-authenticate parameter header for the challenge. It was modified in order to compare the nonce received from the IMS client (that contains the token extracted after the pana authentication) with the token received from the HSS (in the MAA message). If there is a match it returns 2 (1 on standard registration success and 0 on error).

Finally if the *S\_challenge* returns success for our authentication, the *Dumb\_cmd\_2* returns 1 (success), -1 on error. If it fails a standard registration is carry out.

*Dumb\_cmd\_2* is used in the *scscf.cfg* file, a configuration script that defines the Serving behavior.

### 3.3 UCT IMS client modification

In order to implement our proposal we had to use the dynamic library of the pana client in the IMS client to perform the access network authentication before the IMS authentication and add the returned value of the function *main\_client* and the aib body to the SIP REGISTER message.

Therefore we modified the makefile in order to use the dynamic library of the pana client. After that, the *main\_client* function was added to the function:

```
Gint ims_send_register()
```

The value returned by the pana function was added to the nonce field of the SIP REGISTER message.

The AIB body was created using OpenSSL software (version 0.98): we created the *sipfrag* part of the body including the WWW-authenticate parameter header and the header filed:

- Contact
- Call\_id
- From
- Cseq header

A timestamp (date and hour) was also added.

We put this part in a file and we passed it to an OpenSSL command in order to generate the required body:

```
openssl smime2 -sign -signer certs/ca.cer -inkey keys/ca.key -in sipfrag.txt -out register_body.txt -passin file:passwd.txt
```

*smime2* is a command created by us started from the original one *smime*. Some changes we did in order to obtain a well formed aib body.

To add the aib part to the REGISTER message we used the function:

```
osip_message_set_body(reg, buffer_AIB, strlen(buffer_AIB));
```

The *buffer\_AIB* is the body created with OpenSSL.

## 4 Testing

In this section we depict the different test phases carried out in order to validate the proposal. On one hand we need to enable IMS core/client architecture to send and receive modified messages in the registration process and secondly we need to establish a PANA association between PaC and PAA applications. Finally we tested the integration of all the components of the proposed IMS registration.

### 4.1 Architecture

The currently architecture is a simplified one:

1. All the element of the IMS core works on one machine:
  - Proxy – CSCF: sip port 4060
  - Interrogating – CSCF: sip port 5060, diameter port = 3869
  - Serving – CSCF sip port 6060, diameter port = 3870
  - Home Subscriber Server (HSS): diameter port = 3868

The domain name used is “carissimi.gast.it.uc3m.es”.

Access network elements also run in the same machine:

2. PANA listening port are 3001 for the Paa and 3002 for the PaC
3. The domain associated with the Paa is “localdomain2.net”
4. The external AAA listening port is 3885 with the server name “hss2.carissimi.gast.it.uc3m.es”

### 4.2 Tests

#### 4.2.1 Send modified register message

- **Description**

The first phase of the testing was defining the modified SIP REGISTER message

In this test, it was checked the capability of the UCT client to create and send a SIP REGISTER message that includes a well formed S/MIME body.

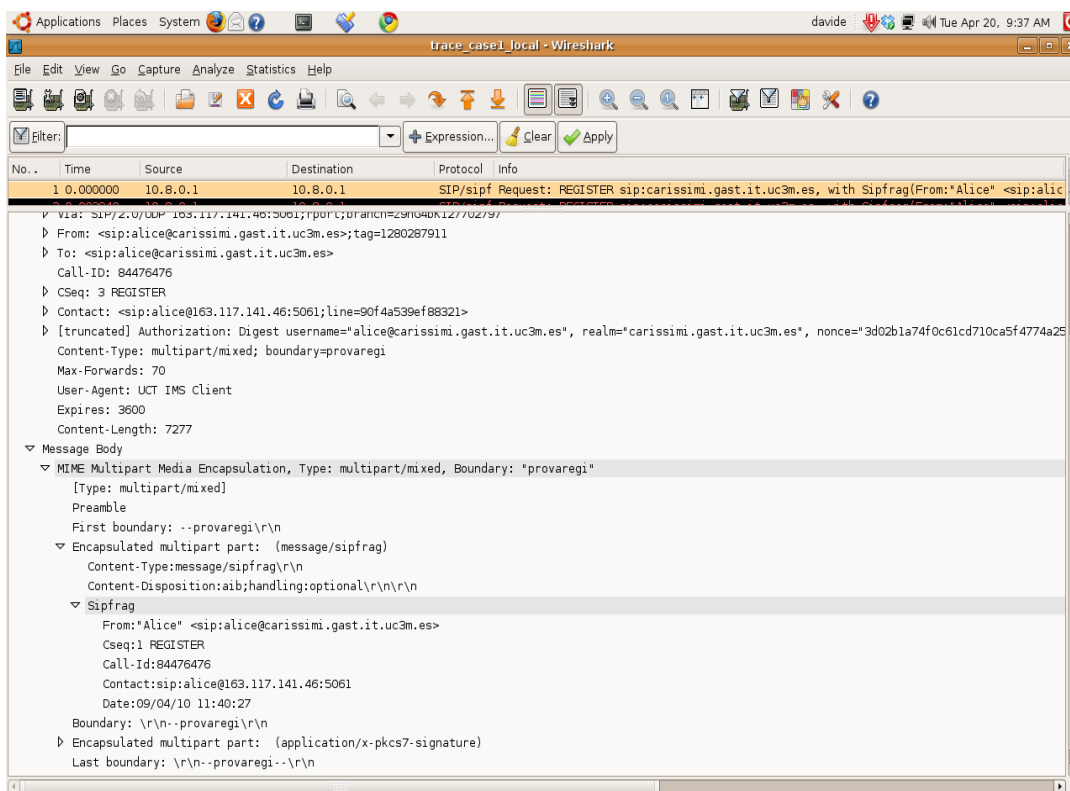
- **Tools used**

UCT IMS client, OpenIMSCore, OpenSSL and Wireshark (packet analyzer)

- **Results**

We were able to create and send the modified message. Using Wireshark we also checked that the message was well formed.

Following picture depicts the modified SIP REGISTER message we have created using UCT IMS client.



**Figure 3: SIP Register message with the S/MIME body containing both sipfrag part and pkcs7 part**

#### 4.2.2 PANA EAP-TLS Authentication

- **Description**

In this test, it was checked the capability of the PaC application to authenticate itself through the Paa with the local EAP-TLS (AAA) server. Tests include authentication (eap-tls tunnel open) and termination (eap-tls tunnel closed).

- **Tools used**

PaC, Paa and EAP-TLS server applications to perform the access network authentication and Wireshark.

- **Results**

The result of this test was the successful establishment a cryptographically protected tunnel between the PaC and AAA using EAP-TLS as authentication mechanism.

The tunnel was successfully closed by client exchanging a PANA termination request and answer messages (PTR, PTA).

The same test has been done using the HSS as authenticator.

Following picture show the debug of the pana client (left) and the local eap-tls server (right). We can appreciate the authentication of the user (after the PBA message) and termination (PTR/PTA messages).

```

PDI enviado panac_message.c, line 343.....
>>> PSR : <Nonce> <Cookie>
<<< PSA : <Cookie> <Nonce>
>>> PAR : <Session-Id> <EAP-Payload>
<<< PAN : <Session-Id> <EAP_Payload>
>>> PAR : <Session-Id> <EAP-Payload>
<<< PAN : <Session-Id> <EAP_Payload>
>>> PAR : <Session-Id> <EAP-Payload>
<<< PAN : <Session-Id> <EAP_Payload>
>>> PAR : <Session-Id> <EAP-Payload>
<<< PAN : <Session-Id> <EAP_Payload>
>>> PAR : <Session-Id> <EAP-Payload>
<<< PAN : <Session-Id> <EAP_Payload>
>>> PBR : <Session-Id> <Session-Lifetime> <Result-Code=Authenticated> <E/
<<< PBA : <Session-Id> <Result-code> <Key-Id> <MAC>
elapsed time : 0s 61993u

<<< PTR : <Session-Id> <Term-Cause=1> <MAC>
>>> PTA : <Session-Id> <MAC>
BYE !!!!!
davide@davide-netlab:~/PANA_C/client-pana-eaptls/src$ █

EAP server certificate : /home/davide/Desktop/openssl-1.0.7-i/libdiametereap/tes
Clients CA certificate : /home/davide/Desktop/openssl-1.0.7-i/libdiametereap/tes
Server Private Key file : /home/davide/Desktop/openssl-1.0.7-i/libdiametereap/tes
Diffie-Hellmann file   : /home/davide/Desktop/openssl-1.0.7-i/libdiametereap/tes
Random generation file : /home/davide/Desktop/openssl-1.0.7-i/libdiametereap/tes
Client CA cert path    : /home/davide/Desktop/openssl-1.0.7-i/libdiametereap/tes
Server cert path       : /home/davide/Desktop/openssl-1.0.7-i/libdiametereap/tes
Packet chunk size      : 2000
Authentication type    : eap_local
SCVP server             : (null)
Listening socket file   : /home/davide/PANA_C/server-pana-eaptls/localsockfile_AAA
-----
eap.c: initiate_eap_session
initiate eap session --cont--
initiate eap session --cont22--
initiate eap session --cont333--
received eap identity answer!!
in eap_validation
in eap_validation
in eap_validation
in eap_validation
We have authenticated bob@carissimi.gast.it.uc3m.es
Closing session of bob@carissimi.gast.it.uc3m.es ...
█

```

Figure 4 – PaC (above) authentication and termination with the local eap-tls server (below)

#### 4.2.3 Generation of security token and PANA security association establishment

- **Description**

In this part we tried out the TLS extractor function we defined. After a PANA EAP-TLS successful authentication both Pac and EAP-TLS server extract a proof.

The EAP-TLS server sends its key material to the Paa in order to create a PANA association with the PaC.

- **Tools used**

PaC, PAA and EAP-TLS server applications to perform the access network authentication and Wireshark.

- **Results**

In this test, PaC and PAA application were able to establish a PANA association after EAP-TLS authentication was performed.

The same test has been done using the HSS as authenticator.

#### 4.2.4 Send the security token to the HSS using Paa

- **Description**

We checked the proper functioning of the modified Paa. As we explained earlier the Paa has to open a connection with the HSS to send it the token retrieved from the EAP-TLS authentication.

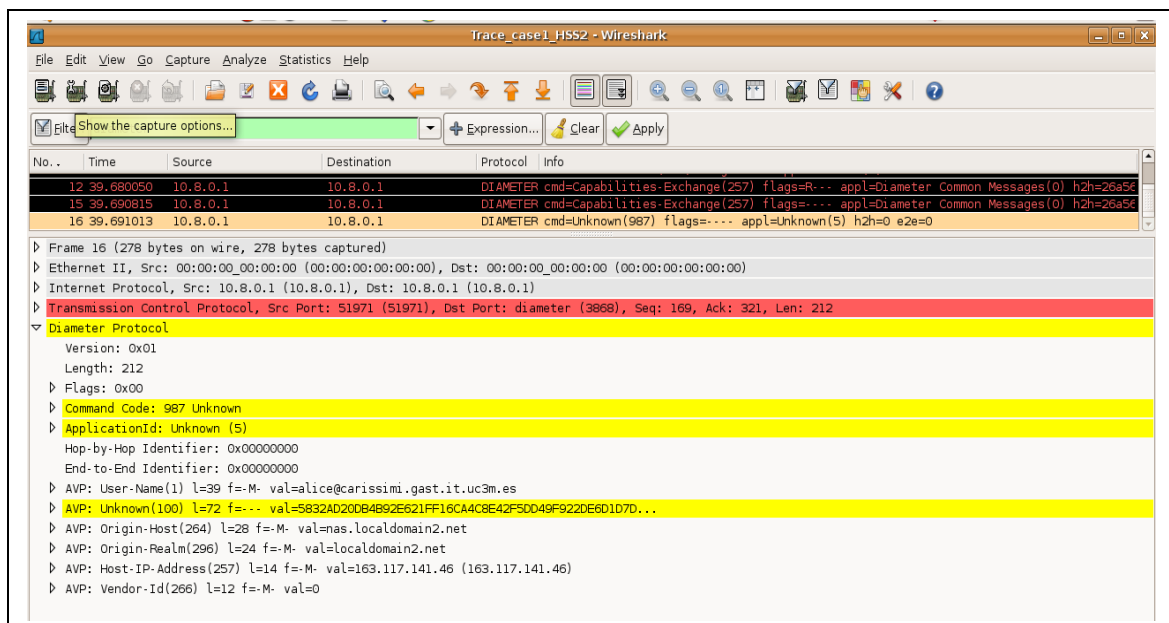
- **Tools used**

PaC, PAA and EAP-TLS server applications to perform the access network authentication, the HSS and Wireshark.

- **Results**

After the PANA EAP-TLS authentication the Paa was able to connect with the HSS and sent to it the 987 diameter message containing the identity of the user authenticated and the extracted token.

Following picture is a Wireshark capture that shows the message exchange required in order to send the security token from the Paa to the HSS.



Picture 5 – Paa connects with the HSS (257 messages) and send to it the token (987 message) and user identity

#### 4.2.5 HSS parses the Paa message and stores the authentication token in the database

- **Description**

After the reception of the diameter message containing the security token, the HSS should parse it, extract the proof and store it in the eap\_tls table of the database.

- **Tools used**

PaC, PAA and EAP-TLS server applications to perform the access network authentication, the HSS and Wireshark.

- **Results**

The result showed the correctly parsing of the message. Store the extracted token in the database was also achieved.

Following picture shows the eap\_tls table where the HSS stores the identity and the token of the authenticated user.

```
mysql> use hss_db
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from eap_tls
-> ;
+-----+-----+-----+
| id | impi | proof |
+-----+-----+-----+
| 7 | alice@carissimi.gast.it.uc3m.es | b24465437d2cccfa754f77647721c9891f953a63d1f81afc4559e41b5158b60d9d3c7dfbb17576018fbd6626ff69416798d39893700b553e9d1acf1785c297f |
| 5 | bob@carissimi.gast.it.uc3m.es | da3c79f888d87018ac166e253fe42cade85560694f61961ccbe8c1acf25b46a144eb36771cf66dfd17a4203a7019ae0649650e58f1d9574efb8f42d9ab991dd6 |
| 9 | davide@carissimi.gast.it.uc3m.es | 82260cccd6873092bfc4de865fe7bad81c02ddf5b7a4af4ec6f2eb7996a389c31dbfbef020f1315f86860df50247afbe9c8cbeae3f3b89b2bb86372f47399c51 |
+-----+-----+-----+
3 rows in set (0.00 sec)
```

Figure 6 – eap\_tls table where HSS stores the token(proof) and identity of the user (impi)

#### 4.2.6 MAA/MAR Exchange

- **Description**

In this test, it was checked the message exchange among the SCSCF and the HSS in order to download the authentication vector used to authenticate the IMS user.

- **Tools used**

UCT IMS client, OpenIMSCore, OpenSSL and Wireshark.

- **Results**

After the SIP REGISTER was received the SCSCF was able to download the modified authentication vector that also contains the security token retrieved from the access network authentication.

Following picture shows the SCSCF log where the nonce from the client is retrieved along with the token received from the HSS.

```

Apr 20 11:42:42 davide-netlab /opt/OpenIMSCore/ser_ims/ser[7090]: ERR:S-CSCF:pack_challenge (out of
if): nonce
received: : fcc05f1115188c5c34f81ec4d47fb5ff67b72773309220f1e4285917150d64ed22f5eddb49ccb6509d5d32a98f3d5
8c9cf4f772e1bb6241c36f3759ac4e40525",
.....
.....
.....
.....
Apr 20 11:42:42 davide-netlab /opt/OpenIMSCore/ser_ims/ser[7090]: ERR:S-
CSCF:pack_challenge: token extracted:
fcc05f1115188c5c34f81ec4d47fb5ff67b72773309220f1e4285917150d64ed22f5eddb49ccb6509d5d32a98f3d58c9cf4f772
e1bb6241c36f3759ac4e40525
    
```

**Figure 7 – SCSCF log**

**4.2.7 Complete IMS registration using our protocol**

- **Description**

The last part was to test our proposal. Therefore a complete IMS registration was performed.

- **Tools used**

We used the modified UCT IMS Client that includes the pana client, the Paa, the EAP-TLS server, OpenIMSCore and Wireshark

- **Results**

The result of this test was a successful IMS registration using our protocol.

The Paa application sends the token extracted from the EAP-TLS authentication to the HSS over diameter protocol while the IMS client includes in the SIP REGISTER message the proof retrieved from the access network authentication.

The S-CSCF checks the sign and the token with the information received from the HSS and a 200OK message is sent to the client.

The same test has been done using the HSS as authenticator.

Following Wireshark capture shows the whole authentication process (access network authentication and IMS registration) using our protocol. Access network authentication is done using the external AAA.

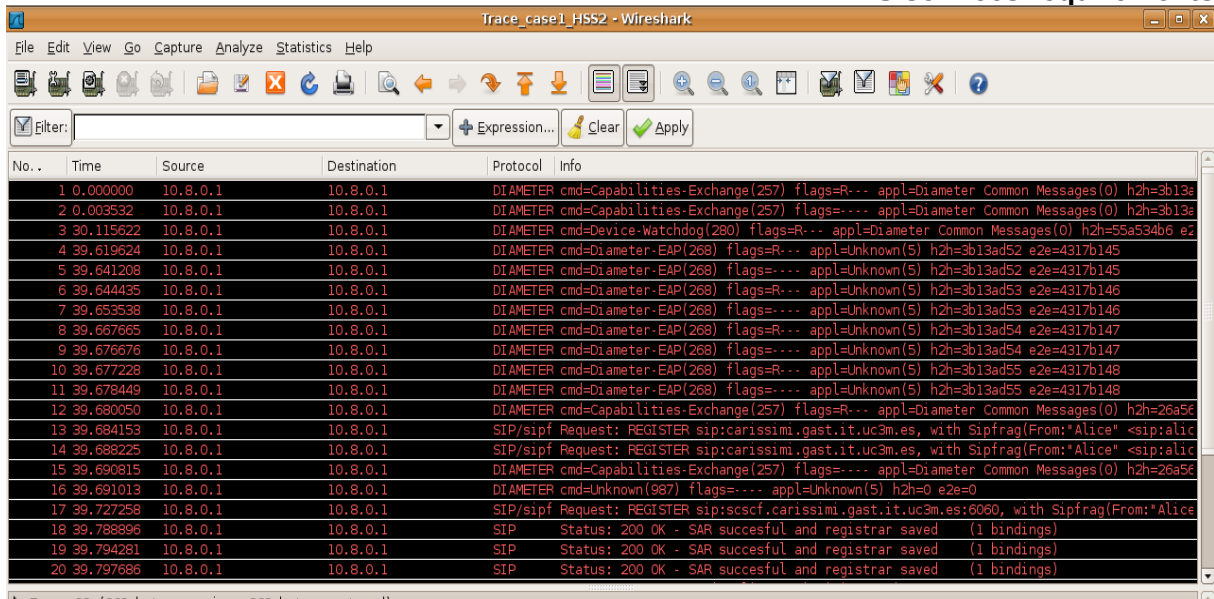


Figure 8 – IMS authentication using our protocol and the external AAA

Following picture shows the UCT IMS Client authentication using our protocol. As we see after the SIP REGISTER message a positive 200OK answer is received. The 401 challenge message isn't required.

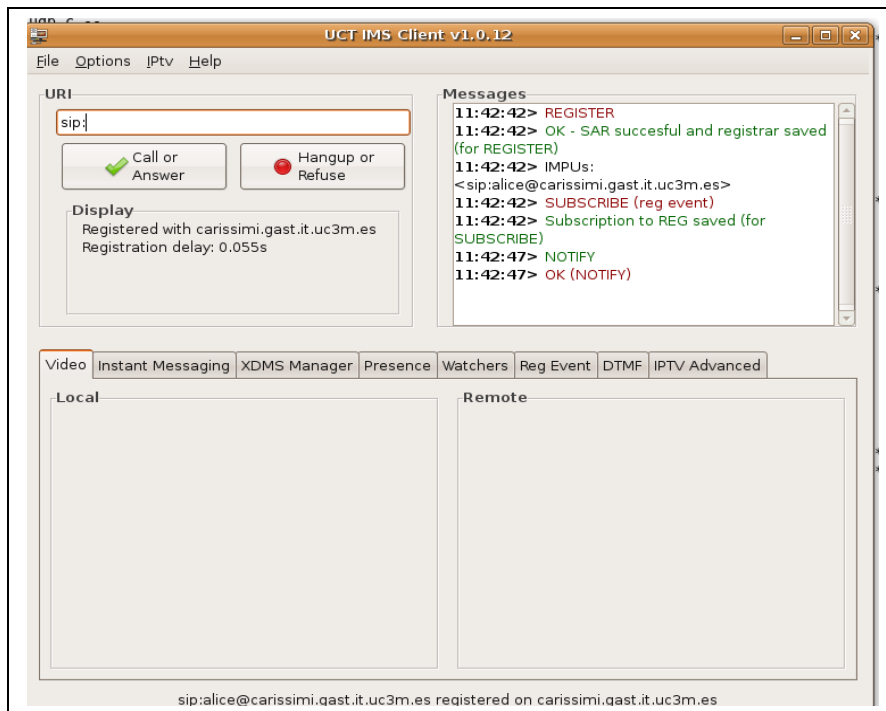


Figure 9 – UCT Client one pass registration

## 5 Future work

In order to prove the benefits of our proposal one future work will be test our protocol in terms of time consuming with respect the standard IMS registration. We will simulate a roaming scenario for IMS in order to verify the goodness of our prototype. Moreover this kind of test could give us some more details about the stability and resource consuming of our implementation.

Another important task would be trying out the prototype with different IMS client in order to verify the compatibility and possible incompatibility. We could verify how to create a modified REGISTER message with others implementations and integrate the PaC into them.

Porting the PaC to a mobile device and use it with a mobile IMS client could be another part of our future works.

Further works could include the implementation of a class that defines the required TLS function for the HSS in order to completely integrate the EAP-TLS module. We could use PureTLS implementation (a free Java-only implementation of the SSLv3 and TLSv1 (RFC2246) protocols) for this scope.

Moreover we're thinking on how we can exploit the tunnel created in the access network authentication in new scenarios where concept of digital identity is used. The idea is to use TLS layers [12] to “upgrade and downgrade” the level of authentication sending attributes over the tunnel in order to gain access to third party services.

## **Bibliography**

1. **Rescorla, E.** *Keying material exporters for transport layer security (tls)*. Technical Report draft-ietf-tls-extractor-05.txt, IETF (2009) <http://tools.ietf.org/html/draft-ietf-tls-extractor-05>.
2. **Peterson, J.** *Session initiation protocol (sip) authenticated identity body (aib) format*. Technical Report RFC3893, IETF (2004) <http://www.ietf.org/rfc/rfc3893.txt>.
3. **Dierks, T., Allen, C.** *The transport layer security (TLS) version 1.0*. Technical Report RFC2246, IETF (1999) <http://www.ietf.org/rfc/rfc2246.txt>.
4. **Ohba, Y., Baba, S.** *Pana over tls*. Technical report, IETF (2002).
5. *Protocol for Carrying Authentication for Network Access (PANA) Framework*, RFC 5193.
6. **Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., Schooler, E.** *Sip: Session initiation protocol*. Technical Report RFC3261, IETF (2002).
7. **Daniel Díaz Sanchez, Davide Proserpio, Andrés Marín López, Florina Almenárez-Mendoza and Peter Weik.** *A general IMS registration protocol for wireless networks interworking*.
8. *Technical report, 3GPP, Third Generation Partnership Project, Technical Specification Group Services and Systems Aspects, 3G Security, Security Architecture, Technical Specification 3G TS 33.102, V3.7.0 (2000)*.
9. **Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A., Stewart, L.** *Http authentication: Basic and digest access authentication*. Technical Report RFC2617, IETF (1999) <http://www.ietf.org/rfc/rfc2617.txt>.
10. *Pana Eap-tls implementation*: <http://www-lor.int-evry.fr/~maknavic/VERICERT/SP3/>.
11. **Daniel Díaz Sánchez, Andrés Marín y Florina Almenarez, Celeste Campo, Alberto Cortés and Carlos García-Rubio.** *Trust Negotiation Protocol Support for secure mobile network service deployment*.