

Universidade de Aveiro

# Comunicações Móveis

## Projeto IMS



João Gameiro (93097), Pedro Abreu (93240)

Departamento de Electrónica, Telecomunicações e Informática

3 de fevereiro de 2022

# Conteúdo

<b>Lista de Figuras</b>	<b>ii</b>
<b>Glossário</b>	<b>iii</b>
<b>1 Resumo</b>	<b>1</b>
<b>2 Introdução</b>	<b>2</b>
<b>3 Implementação</b>	<b>5</b>
3.1 Kamailio . . . . .	5
3.2 OpenIMSCore . . . . .	5
3.2.1 Instalação de pré-requisitos . . . . .	5
3.2.2 Configuração da Máquina Virtual . . . . .	10
3.2.3 Chamada SIP . . . . .	11
<b>4 Conclusão</b>	<b>16</b>
<b>Bibliografia</b>	<b>17</b>

# Lista de Figuras

2.1	Arquitectura conceptual do IMS . . . . .	2
2.2	Arquitectura geral do core IMS . . . . .	3
3.1	Bases de dados que devem estar contidas no mysql . . . . .	7
3.2	Conteúdo do ficheiro /etc/bind/named.conf.local . . . . .	8
3.3	Componentes da esquerda para a direita de cima para baixo (pcscf, scscf, icscf e hss) . . . . .	9
3.4	Interface Web para gestão do HSS . . . . .	10
3.5	Configuração do adaptador de rede da Máquina Virtual . . . . .	11
3.6	Credenciais da alice . . . . .	12
3.7	Credenciais do bob . . . . .	13
3.8	Troca de pacotes relativas a uma tentativa de autenticação . . . . .	13
3.10	Troca de pacotes relativas ao término da chamada . . . . .	15

# Glossário

**3GPP** 3rd Generation Partnership Project

**CSCF** Call Session Control Function

**DNS** Domain Name System

**GCC** GNU Compiler Collection

**GPRS** General Packet Radio Service

**HSS** Home Subscriber Server

**I-CSCF** Interrogating - Call Session Control Function

**IMS** IP Multimedia SubSystem

**P-CSCF** Proxy - Call Session Control Function

**QoS** Quality of Service

**RTCP** RTP Control Protocol

**RTP** Real-Time Transport Protocol

**SQL** Structured Query Language

**SIP** Session Initiation Protocol

**S-CSCF** Serving - Call Session Control Function

**UMTS** Universal Mobile Telecommunications System

# Capítulo 1

## Resumo

O presente relatório visa descrever a implementação do projeto sobre IP Multimedia SubSystem (IMS) proposto no âmbito da unidade curricular de Comunicações Móveis do curso de Engenharia de Computadores e Telemática da Universidade de Aveiro.

O principal objetivo deste projeto era aprofundar e estudar o ecossistema IMS, através da instanciação dos seus componentes e análise dos mesmos em operação para vários tipos de serviços como por exemplo voz e multimédia.

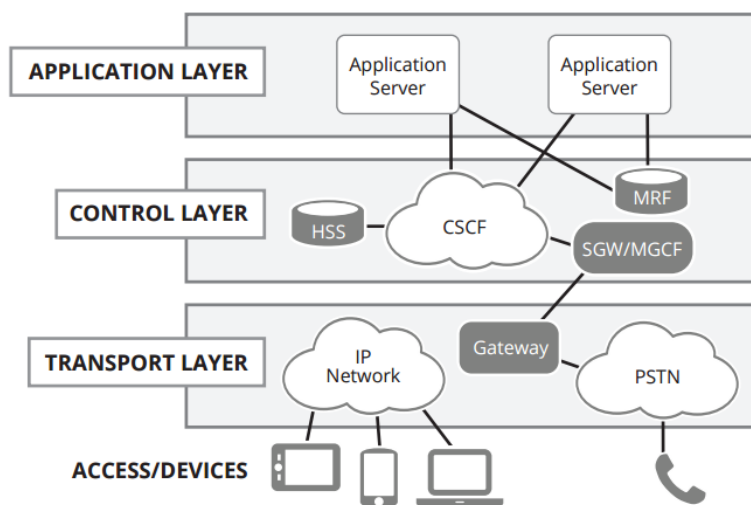
Este relatório está fundamentalmente dividido em 3 capítulos, sendo que no primeiro (Capítulo 2) será apresentada uma introdução teórica ao IMS, no segundo (Capítulo 3) uma descrição da implementação dos componentes IMS e finalmente no 3º e último capítulo (Capítulo 4) a consequente conclusão do projeto.

## Capítulo 2

# Introdução

[1]

O IMS é um dos projectos de maior sucesso do 3rd Generation Partnership Project (3GPP). Foi originalmente desenvolvido para permitir a redes Universal Mobile Telecommunications System (UMTS) a possibilidade de transportarem tráfego multimédia a utilizadores *mobile*. No fundo o IMS oferece uma plataforma comum que fornece o controlo de sessões multimédia, suporte para implementação de serviços e um acesso independente da tecnologia a ser utilizada pelos utilizadores.



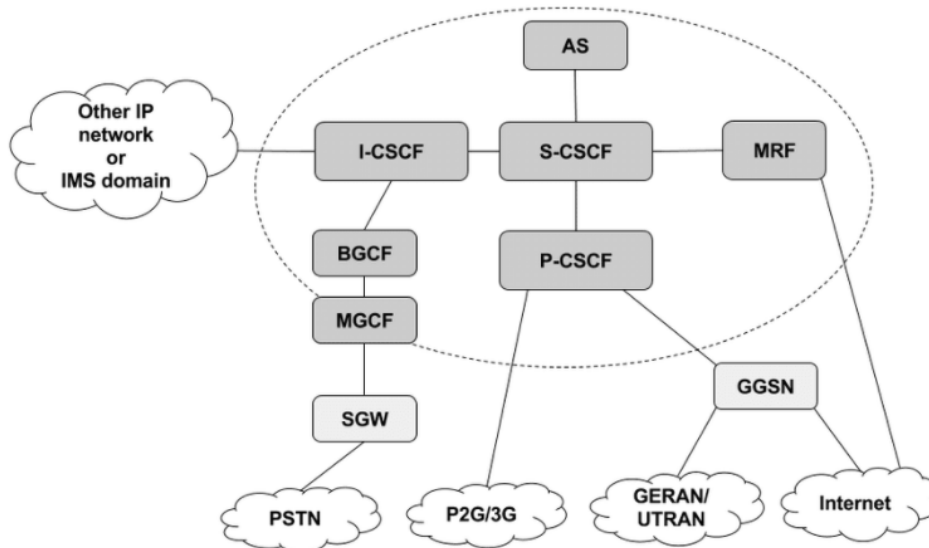
**Figure 2.1:** Arquitectura conceptual do IMS

Conceptualmente podemos dividir o IMS em três camadas:

- **Camada de Serviços** constituída por servidores aplicacionais, *Web Servers*, entre outras coisas, que no fundo oferecem serviços aos utilizadores.
- **Camada de Controlo** que permite controlar e gerir sessões, assim como gerir e armazenar informação à cerca dos seus utilizadores, assim como fornecer serviços para gerir o acesso dos mesmos à rede (ex.: localização, autenticação, autorização de serviços, etc).

- **Camada de Transporte** que suporta as arquitecturas das redes de *core* General Packet Radio Service (GPRS). É nesta camada que se situam os nós das redes, como *routers* ou *firewalls*, em conjunto com *gateways* que auxiliam na tradução de protocolos de tráfego provindo de redes de pacotes ou de circuitos.
- **Dispositivos de acesso** que representam ser uma das grandes vantagens do IMS, o facto de permitir uma acesso aos seus serviços independente da tecnologia que se está a usar (computador, telefone, tablet, etc).

Algumas das vantagens mais importantes do IMS incluem o facto já referido anteriormente, de ser independente da rede de acesso (cablada, ou wireless por exemplo), do dispositivo ou da aplicação. IMS também suporta Quality of Service (QoS), assim como uma grande variedade de serviço tais como mensagens de voz, texto, video conferência, etc. Outra vantagem que torna o IMS tão atractivo é o facto de ser escalável e facilmente permitir a integração com outros domínios IMS. A principal desvantagem do IMS é o facto de para a sua implementação ser necessário não só um conhecimento avançado em termos de redes de comunicação, mas também o conhecimento dos vários protocolos, componentes, softwares e interfaces que permitem a integração e construção de um *core* de suporte.



**Figure 2.2:** Arquitectura geral do core IMS

Em termos de arquitectura do core IMS identificamos os seguintes componentes:

- **Call Session Control Function (CSCF)** que é um servidor Session Initiation Protocol (SIP) que processa a sinalização de chamadas ou sessões. Existem várias entidades que constituem o CSCF e as principais serão descritas nos pontos seguintes.
- **Proxy - Call Session Control Function (P-CSCF)** primeiro ponto de contacto com o utilizador que está localizado no domínio do utilizador ou no rede do servidor, dependendo do ponto de acesso. Esta entidade comporta-se como um *proxy*, na medida

em que recebe e reencaminha pedidos do utilizador para o servidor e o mesmo na ordem inversa.

- **Interrogating - Call Session Control Function (I-CSCF)** primeiro ponto de contacto do domínio do servidor. Tem funções como registo, sessões, facturação e utilização de recursos. Age como uma *firewall* do domínio, restringindo acesso protegendo as outras entidades do domínio.
- **Serving - Call Session Control Function (S-CSCF)** localizado no domínio, é o ponto central de funcionamento. Actua como um servidor SIP gerindo perfis e sessões dos utilizadores. Através deste ponto é permitido aos operadores controlar e monitorizar o fornecimento de serviços.
- **Home Subscriber Server (HSS)** base de dados que funciona como um repositório para as informações relacionadas com os utilizadores da rede. Contém informações como identificação, endereçamento, segurança, localização e perfis dos utilizadores.

A escalabilidade de componentes prende-se na capacidade que o IMS tem em suportar por exemplo vários HSSs na mesma rede, ou por exemplo vários S-CSCF no mesmo domínio, ou até mesmo a interligação de vários domínios IMS.

Para o controlo das sessões o IMS usa o protocolo SIP que possui características como por exemplo ser um protocolo baseado em texto, o que facilita bastante a sua compreensão, e também facto de ser um modelo cliente-servidor o que facilita bastante a sua integração com o IMS. Pelo facto do IMS usar bastante o SIP o primeiro passo do projeto foi então fazer uma chamada SIP num ambiente não IMS. Para isso foi configurado um servidor *Asterisk* com dois utilizador num computador com *Ubuntu 20.04.3 LTS* de raiz. Depois utilizando um telemóvel a correr um *Softphone* (neste caso o *MizuDroid*) e na mesma máquina (do servidor) registar um utilizador com outro *Softphone* (*Ekiga*), foi feita uma chamada entre os dois e observados os pacotes SIP capturados. Estes não são apresentados nesta secção pelo facto de mais à frente ir ser descrita um processo de instalação de IMS e uma chamada SIP utilizando como servidor a tecnologia referida.



## Capítulo 3

# Implementação

### 3.1 Kamailio

Uma das tecnologias sugeridas para este projeto foi o *Kamailio* [2]. O *Kamailio* é um servidor SIP *Open-Source* robusto o suficiente para aguentar milhares de estabelecimentos de chamadas por segundo. *Kamailio* pode ser usado para plataformas de comunicações multimédia em tempo real e é escalável o suficiente para ser integrado com tecnologias como *Asterisk*, *FreeSwitch* ou *SEMs*.

Inicialmente devido ao grande número de problemas que surgiram na instalação do *OpenIMSCore* tentámos mudar para esta solução seguindo o tutorial que se encontra em

<http://www.kamailio.org/wiki/tutorials/ims/installation-howto>

No entanto como surgiram novamente bastantes problemas na instalação deste serviço e como foi possível ao mesmo tempo instalar o *OpenIMSCore* esta solução foi abandonada para que o resto do projeto se concentrasse no *OpenIMSCore*.

### 3.2 OpenIMSCore

Para a implementação do core IMS foi utilizado o *OpenIMSCore* [3]. O *OpenIMSCore* é uma implementação *Open-Source* das funcionalidades de *Call Session Control Functions* (CSCFs) que em funcionamento conjunto constituem um *core* IMS. Este projeto *Open-Source* não foi desenvolvido com o objectivo da criação de um produto para venda comercial, mas sim da criação de uma implementação base de referência de um *core* IMS cuja finalidade era ser usada para testes, prototipagem e investigação do comportamento dos sistemas IMS.

#### 3.2.1 Instalação de pré-requisitos

Uma das principais dificuldades na implementação deste projeto foi a procura de recursos actualizados na Internet para que permitissem responder e resolver os problemas encontrados. Isto deveu-se ao facto do *OpenIMSCore* ser uma implementação *Open-Source* de 2004-2008 o que implica que bastante do material encontrado se encontre desactualizado. Por este motivo, grande parte das bibliotecas que são referidas no guião de instalação [4] neste momento estão disponíveis em versões muito mais avançadas que aquela disponível no momento da criação do *OpenIMSCore*.

Sendo, as bibliotecas instaladas foram as seguintes:

```
#Comandos necessários para instalar as bibliotecas
sudo apt-get install libcurl4-gnutls-dev
sudo apt-get install bison
sudo apt-get install curl
sudo apt-get install debhelper cdb lintian build-essential
sudo apt-get install devscripts pbuilder dh-make debbootstrap dpatch flex
sudo apt-get install libxml2-dev libmysqlclient15-dev ant docbook-to-man
sudo apt-get install ipsec-tools
sudo apt-get install subversion
sudo apt-get install mysql-server-5.5
sudo apt-get install openjdk-7-jre
sudo apt-get install openjdk-7-jdk
```

Também era necessário o compilador GNU Compiler Collection (GCC), pelo que a versão que foi utilizada foi a 4.9.2.

Para além deste conjunto de bibliotecas era também necessário um servidor Domain Name System (DNS), sendo que o que foi usado foi o *Bind9*. Como para implementar este sistema era necessário um ambiente Linux, foi inicialmente tomada a decisão de o fazer em máquinas que continham *Ubuntu* de raiz (*Ubuntu 20.04.3 LTS*). No entanto com o crescente aumento de problemas na instalação foi tomada a decisão de mudar a instalação para uma máquina virtual *Debian LXDE*.

Após ter instalado os pré-requisitos, o próximo passo é obter o código do OpenIMS-Core. Para o fazer criou-se um directório *opt/OpenIMSCore* e lá dentro mais dois directórios: *ser\_ims* e *FHoSS*. O código que permite implementar o HSS será colocado no directório *FHoSS*, por sua vez o resto do código que implementa os restantes componentes é colocado no directório *ser\_ims*. Para obter o código foram executados os seguintes comandos:

```
#Estando no directório \opt\OpenIMSCore
svn checkout https://svn.code.sf.net/p/openimscore/code/ser_ims/trunk ser_ims
svn checkout https://svn.code.sf.net/p/openimscore/code/FHoSS/trunk FHoSS
```

```
#Para compilar o código
cd ser_ims
sudo make install-libs all
cd ..

cd FHoSS
sudo make ant compile
sudo make ant deploy
cd ..
```

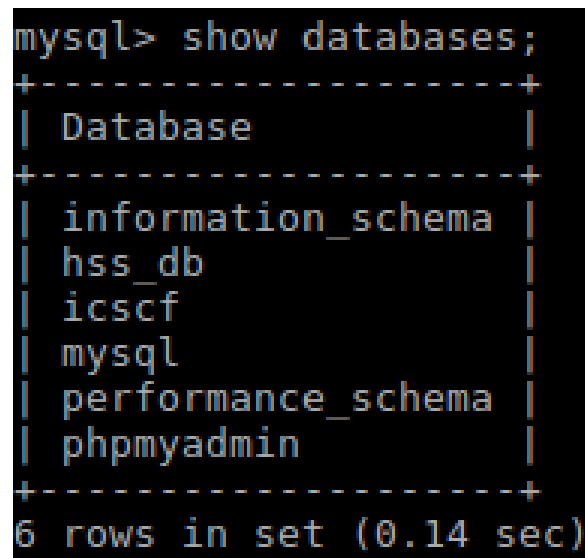
Ao terem sido cumpridos todos estes passos sem erros, temos todo o código presente e devidamente compilado pronto para ser executado. No entanto ainda é preciso configurar o ambiente de desenvolvimento para podermos conseguir executar todo o projeto da forma correta. Para isso é necessário copiar os ficheiros Structured Query Language (SQL) do projeto para o servidor *MySQL*. Para tal efeito é necessário correr os seguintes comandos:

```
mysql -u root -p -h localhost < ser_ims/cfg/icscf.sql
mysql -u root -p -h localhost < FHoSS/scripts/hss_db.sql
mysql -u root -p -h localhost < FHoSS/scripts/userdata.sql
```

Após a execução destes comandos é recomendável a verificação de se as bases de dados foram corretamente copiadas para o servidor para isso é necessário:

```
#Log in no servidor
mysql -u root -p
#inserir passe

#Após log in com sucesso correr
mysql> show databases;
```



```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| hss_db            |
| icscf             |
| mysql             |
| performance_schema |
| phpmyadmin        |
+-----+
6 rows in set (0.14 sec)
```

**Figure 3.1:** Bases de dados que devem estar contidas no mysql

Para terminar a configuração do ambiente de desenvolvimento falta apenas a preparação do servidor DNS. Para isso é necessário copiar o ficheiro *open-ims.dnszone* para o diretório bind através do seguinte comando:

```
sudo cp /opt/OpenIMScore/ser_ims/cfg/open-ims.dnszone /etc/bind/
```

Após a cópia do ficheiro é necessário editar o */etc/named.conf.local* que por sua vez deverá ter o conteúdo apresenta na Figura 3.2.

```
// Do any local configuration here
//

// Consider adding the 1918 zones here, if t

zone "open-ims.test"{
    type master;
    file "/etc/bind/open-ims.dnszone";
};
```

**Figure 3.2:** Conteúdo do ficheiro `/etc/bind/named.conf.local`

Finalmente é necessário garantir também que o ficheiro `/etc/resolv.conf` tem o conteúdo correto.

```
#Conteúdo do ficheiro /etc/resolv.conf
domain open-ims.test
search open-ims.test
nameserver <ip of the machine>
```

Após isso apenas é preciso iniciar o serviço `bind9`. Para garantir que está tudo a funcionar corretamente apenas é preciso fazer um `ping` para os domínios `pcscf.open-ims.test` ou `icscf.open-ims.test`. Se for obtido uma resposta significa que o ambiente está correctamente configurado.

Finalmente apenas falta um passo antes de se poder correr os *scripts* que implementam o *Core IMS*.

```
#Cópia de ficheiros para o directório base do projeto
cp ser_ims/cfg/*.cfg .
cp ser_ims/cfg/*.xml .
cp ser_ims/cfg/*.sh .
```

Estando tudo configurado e estamos então em condições de finalmente correr o projeto. Para o fazer é necessário correr os seguintes *scripts* na base do projeto:

```
sudo ./pcscf.sh
sudo ./icscf.sh
sudo ./scscf.sh
sudo ./fhoss.sh
```

Todos estes *scripts* devem correr em paralelo. Se por acaso surgirem exceções *Java* ao correr o `fhoss.sh` então é porque provavelmente houve um erro com o caminho para a variável `JAVA_HOME`. Para o correr é necessário editar o ficheiro `/OpenIMSCore/FHoSS/scripts/startup.sh` e na linha aonde está a variável `$JAVA_HOME` editar colocando o seguinte caminho:

```
#Alteração a fazer ao startup.sh em caso de erro
( ... )
$JAVA_HOME/usr/bin/java ( ... )
```

Após a correcção do erro e a execução dos *scripts* o resultado obtido encontra-se representado na Figura 3.3.

Estando todos os componentes a correr, se num browser for visitado <http://localhost:8080> vamos ser apresentados com um *prompt* que pede um nome de utilizador e uma palavra passe. As credenciais a inserir são respectivamente hssAdmin e hss para o username e password. Se as credenciais forem corretamente inseridas será apresentada uma página *web* que permite a gestão do servidor IMS. Nesta página é possível a visualização, configuração e gestão de entidades, assim como a configurações mais avançadas em termos de serviços (ex.: servidores aplicativos, perfis de serviços ou trigger points) e gestão de rede. A interface gráfica está representada na Figura 3.4.

Presumindo todos os passos foram bem sucedidos e efectuados com sucesso, está assim completa a instalação do *OpenIMSCore*.

The image shows four terminal windows from an Oracle VM VirtualBox environment, each displaying logs for different components of the OpenIMSCore system. The windows are titled 'labcom@labcom: /opt/OpenIMSCore'.

- Top-left window (pscf):** Shows logs for the P-CSCF component. It includes messages like 'Registrar Contents end', 'Subscription list begin', 'Subscription list end', 'P-CSCF Dialog List begin', 'P-CSCF Dialog List end', and 'Registrar Contents begin'.
- Top-right window (sscf):** Shows logs for the S-CSCF component. It includes a 'Peer List' section with IP addresses and ports, and messages like 'Registrar Contents begin' and 'Registrar Contents end'.
- Bottom-left window (icscf):** Shows logs for the I-CSCF component. It includes a 'Peer List' section with IP addresses and ports, and messages like 'Registrar Contents begin' and 'Registrar Contents end'.
- Bottom-right window (hss):** Shows logs for the HSS component. It includes messages like 'org.hibernate.impl.SessionFactoryObjectFactory - Factory name: foo', 'org.hibernate.util.NamingHelper - JNDI InitialContext properties:()', 'org.hibernate.impl.SessionFactoryObjectFactory - Bound factory to JNDI', 'WARN org.hibernate.impl.SessionFactoryObjectFactory - InitialContext did not implement EventContext', 'de.fhg.fokus.diameter.DiameterPeer.DiameterPeer - Bean style constructor called, don't forget to configure!', 'de.fhg.fokus.diameter.DiameterPeer.DiameterPeer - FQDN: hss.open-ims.test', 'de.fhg.fokus.diameter.DiameterPeer.DiameterPeer - Realm: open-ims.test', 'de.fhg.fokus.diameter.DiameterPeer.DiameterPeer - Vendor ID : 10415', 'de.fhg.fokus.diameter.DiameterPeer.DiameterPeer - Product Name: JavaD', 'de.fhg.fokus.diameter.DiameterPeer.DiameterPeer - AcceptUnknownPeers: true', 'de.fhg.fokus.diameter.DiameterPeer.DiameterPeer - DropUnknownOnDisconnect: true', and 'de.fhg.fokus.hss.main.HSSContainer - Type "exit" to stop FHSS!'.

**Figure 3.3:** Componentes da esquerda para a direita de cima para baixo (pscf, sscf, icscf e hss)

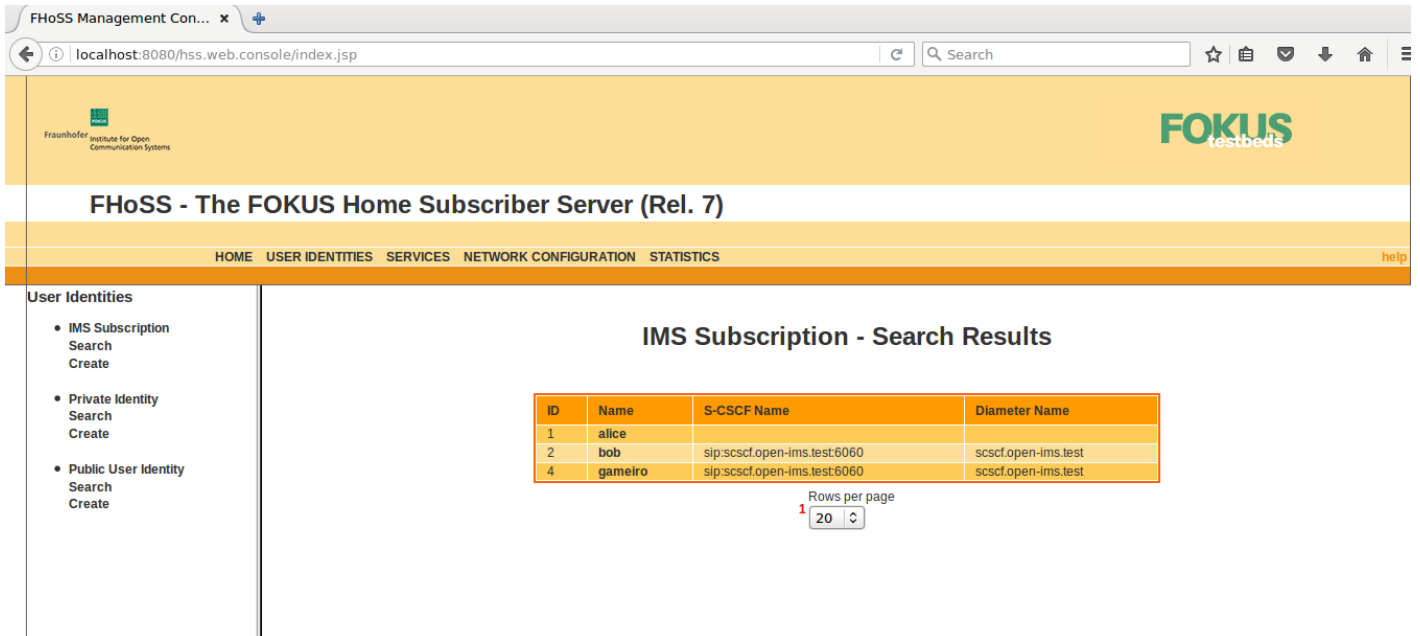
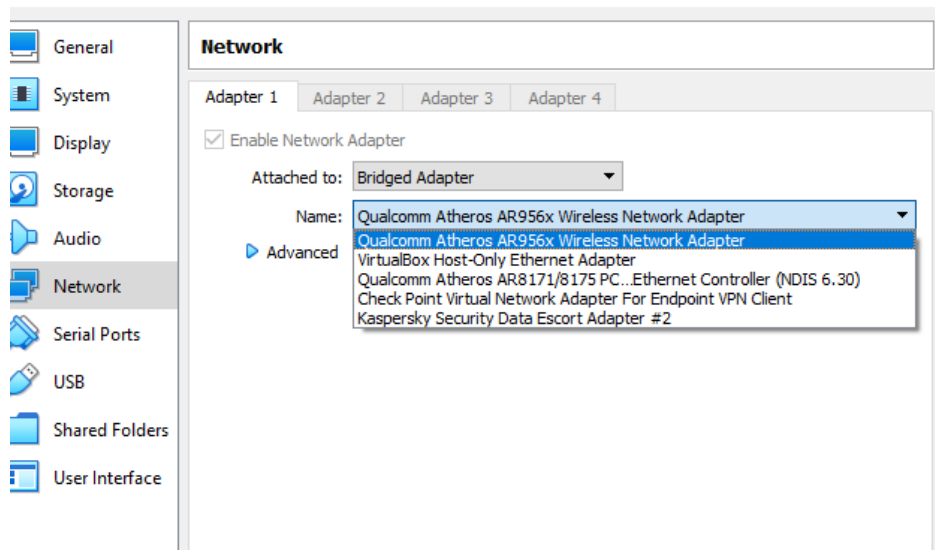


Figure 3.4: Interface Web para gestão do HSS

### 3.2.2 Configuração da Máquina Virtual

Tal como referido na secção anterior, o projeto foi instalado numa máquina virtual *Debian LXDE*. Essa decisão implicou uma preocupação adicional: como deveria ser feita a preparação do servidor para a primeira chamada SIP? Era necessário configurar a máquina virtual de modo a que fosse possível ter conectividade entre a mesma e o *host* ou com outros dispositivos como por exemplo um telemóvel. Ou seja para a chamada SIP precisávamos de dois clientes que tivessem conectividade com o servidor (a correr na máquina virtual).

O primeiro passo passou então por descobrir como ter conectividade com o *host*. Para foi alterado o adaptador de rede da máquina virtual, através do software *Virtual Box* para a opção *Bridged Adapter*, tendo escolhido a placa como opção Name o nome da placa wireless (Figura 3.5). Através da selecção desta opção, interligámos a VM à placa wireless, modo a que a mesma possa estar ligada à rede WiFi, evitando o protocolo de rede do sistema operativo do *host*. Deste modo conseguimos ter o computador, a máquina virtual e por exemplo um terceiro dispositivo todos ligados à mesma rede.



**Figure 3.5:** Configuração do adaptador de rede da Máquina Virtual

Estando configurado o adaptador de rede, é necessário então testar se evidentemente existe conectividade entre o *host* e a VM. Para isso foi necessário desligar a Firewall do Windows pelo facto de esta bloquear *pings* entre a VM e o *host*. Depois dentro da máquina virtual foi feita a ligação à rede, e após ter um IP atribuído, fez-se o teste (*pings* do *host* para o IP obtido na VM).

Se tudo correr bem e houver conectividade total então temos de atualizar o *core ims* para funcionar com o IP correto da VM e não com o 127.0.0.1 que é que está configurado por default. Para efectuar tais alterações é necessário correr o script *configurator.sh* que nos permite definir um IP (neste caso o que foi atribuído à VM) e um domínio (*open-ims.test*) de modo a que todos os ficheiros de configuração sejam alterados para passar a funcionar com esse IP. Finalmente é só preciso alterar se necessário o ficheiro */etc/resolv.conf* para ter o IP da máquina e domínio *open-ims.test* (exemplo referido na secção anterior).

### 3.2.3 Chamada SIP

Para a chamada SIP eram necessários dois clientes, sendo que a escolha recaiu sobre o MicroSIP [5] para um e MizuDROID [6] para o outro. Para a configuração dos clientes SIP foram usadas as credenciais dos clientes *default* que estão presentes no sistema OpenIMSCore: o bob e alice. Para isso era necessário configurar as credenciais tanto no MizuDroid como no MicroSIP.

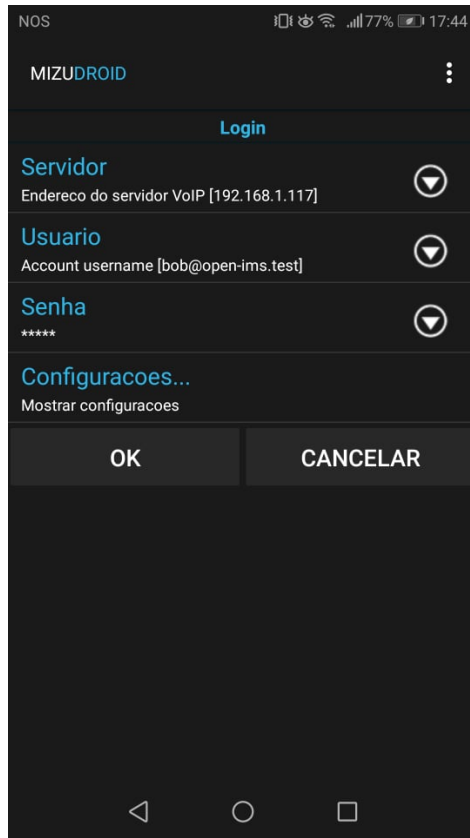
No caso do MicroSIP foi configurada a cliente alice com as credenciais especificadas na Figura 3.6. A sua palavra-passe era alice. Neste caso a máquina virtual quando foi configurado o *Bridged Adapter* ficou com o ip 192.168.1.117. O P-CSCF estava à escuta no porto 4060. AO registar

Nome da Conta	<input type="text" value="alice"/>	
Servidor SIP	<input type="text" value="open-ims.test"/>	<a href="#">?</a>
Proxy SIP	<input type="text" value="192.168.1.117:4060"/>	<a href="#">?</a>
Usuário *	<input type="text" value="alice@open-ims.test"/>	<a href="#">?</a>
Domínio *	<input type="text" value="192.168.1.117"/>	<a href="#">?</a>
Login	<input type="text" value="alice@open-ims.test"/>	<a href="#">?</a>
Senha	<input type="password" value="*****"/>	<a href="#">?</a>

**Figure 3.6:** Credenciais da alice

Por sua vez o bob foi registado num telemóvel que tinha instalada a aplicação MizuDroid. As credenciais de registo de utilizador foram semelhantes ao caso da alice. No caso do bob foi necessário, após inserir as configurações base representadas na Figura 3.7, carregar também na opção 'Configuracoes...' e no campo 'Proxy Address' inserir o IP da máquina virtual com o respectivo porto em que o P-CSCF está à escuta (no nosso caso 192.168.1.117:4060). Após carregar na opção 'OK' se tudo estiver configurado da forma correta será possível o registo do bob no servidor IMS.





**Figure 3.7:** Credenciais do bob

Após o registo dos utilizadores é importante tentarmos perceber o que se passou e que pacotes foram trocados. A partir do momento em que um utilizador se tentou o registar o que aconteceu foi:

- É enviado um pacote SIP REGISTER ao P-CSCF que contém as credenciais do utilizador a ser registado.
- O servidor por sua vez responde com um pacote *Status:401 Unauthorized - challenging the UE* que contém no campo *WWW-Authenticate* que deve ser usada pelo utilizador que se tentou registar na autenticação. Neste caso a autenticação será criada a partir de um *hash MD5* gerado a partir da password e do nome de utilizador do cliente.
- Por sua vez o utilizador responde agora novamente com um *request* REGISTER que contém um novo cabeçalho a indicara informação necessária para efectuar a autenticação com sucesso.
- Se tudo correr bem o servidor (neste caso o proxy porque é o ponto de contacto com o utilizador) irá responder ao pacote do cliente com um 200 OK.

192.168.1.72	192.168.1.117	SIP	930 Request: REGISTER sip:192.168.1.117 (1 binding)
192.168.1.117	192.168.1.72	SIP	977 Status: 401 Unauthorized - Challenging the UE
192.168.1.72	192.168.1.117	SIP	931 Request: REGISTER sip:192.168.1.117 (1 binding)
192.168.1.117	192.168.1.72	SIP	1006 Status: 200 OK - SAR succesful and registrar saved (1 binding)

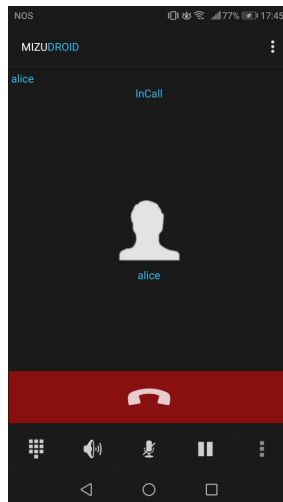
**Figure 3.8:** Troca de pacotes relativas a uma tentativa de autenticação

Por sua vez do lado do servidor as coisas já acontecem de forma diferente:

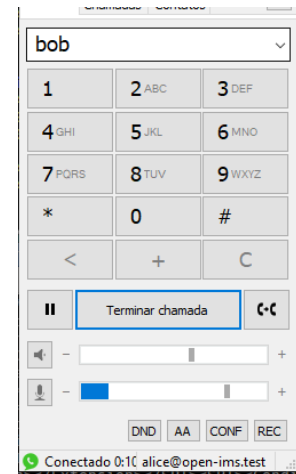
- O P-CSCF reencaminha o SIP REGISTER do utilizador para o I-CSCF.
- Por sua vez o I-CSCF envia um *User Authorization-Request* ao HSS que por sua vez irá responder se o pedido foi bem sucedido ou não. Neste caso se o pedido for bem sucedido o a mensagem enviada ao I-CSCF irá conter o nome do S-CSCF no qual o utilizador irá ser registado.

De forma muito simples e resumida foi isto que aconteceu, e é importante referir que neste caso apenas existe um *Core IMS* mas que em cenários complexos estas trocas de pacotes não são tão triviais como o que foi descrito anteriormente.

Finalmente estando ambos os utilizadores registados e corretamente configurados faltou apenas a parte de realizar a chamada em si. Deste modo no MicroSIP escreveu-se o nome do utilizador do telefone (*bob@open-ims.test*) e carregou-se no botão de chamar. No telefone foi imediatamente recebida e atendido o pedido de chamada.



(a) Chamada do lado do telemóvel (bob)



(b) Chamada do lado do PC (alice)

Em termos de pacotes o que se observou foi:

- Foi enviado um pacote SIP/SDP ao endereço IP do bob (neste caso 192.168.1.64) com origem no IP do servidor.
- Posteriormente, do lado do computador foram recebidos dois pacotes, um 100 SIP Trying e um 180 SIP Ringing (origem IP do bob, destino IP do servidor).
- Assim que a chamada foi atendida os pacotes passaram a ser dos protocolos Real-Time Transport Protocol (RTP) que especifica o formato dos dados de áudio que estão a ser trocados e do protocolo RTP Control Protocol (RTCP) que é responsável por enviar informação de controlo e estatística relativa à chamada.
- No término da chamada é enviado um decline tanto ao servidor como ao outro utilizador.

- Finalmente um SIP BYE tanto ao outro utilizador como ao servidor que por sua vez responde com 200 OK

192.168.1.64	192.168.1.66	SIP	967 Status: 603 Decline
192.168.1.64	192.168.1.117	SIP	967 Status: 603 Decline
192.168.1.64	192.168.1.66	SIP	634 Request: BYE sip:alice@192.168.1.66:59551;ob
192.168.1.66	192.168.1.64	SIP	378 Status: 481 Call/Transaction Does Not Exist
192.168.1.64	192.168.1.117	SIP	634 Request: BYE sip:alice@192.168.1.66:59551;ob
192.168.1.64	192.168.1.66	SIP	662 Request: BYE sip:alice@192.168.1.66:59551;ob
192.168.1.64	192.168.1.117	SIP	662 Request: BYE sip:alice@192.168.1.66:59551;ob
192.168.1.66	192.168.1.64	SIP	377 Status: 200 OK
192.168.1.66	192.168.1.64	RTCP	130 Sender Report Source description Goodbye
192.168.1.66	192.168.1.64	RTCP	110 Receiver Report Source description Goodbye

**Figure 3.10:** Troca de pacotes relativas ao término da chamada

Por sua vez do lado do servidor o que acontece já é um bocado complexo, mas lógica geral de forma muito resumida é:

- O P-CSCF se a entidade especifica no pacote está registada no sistema. E se estiver reencaminha o pedido para o S-CSCF.
- O P-CSCF envia o pacote INVITE para o utilizador que indica que o processo de *setup* da chamada está em execução.
- Por sua vez o S-CSCF faz um *querie* ao icscf, que reencaminha o mesmo para o HSS para se descobrir em que S-CSCF está o utilizador que se quer chamar (neste caso é no mesmo).
- Assim que se decobre é reencaminhada a mensagem para o P-CSCF que por sua vez trata de "contactar" o outro utilizador.

É importante referir que a descrição acima apresentada é bastante simples e resumida. Na realidade este processo é bastante mais complexo que o foi descrito e se considerarmos ambientes IMS maiores essa mesma complexidade aumenta.

## Capítulo 4

# Conclusão

Como conclusão final achamos importante referir que aprofundámos o nosso conhecimento em sistemas UNIX, assim como na área da virtualização e em configuração de máquinas virtuais. Foi possível também aprofundar o conhecimento de IMS e SIP. Em conjunto com este relatório seguirá um anexo que contém duas capturas de dois testes de chamadas SIP feitas como prova de trabalho. Importante referir também que na entrega anterior (materiais da apresentação), estavam contidos dois vídeos que mostram o momento em que foi feita a respetiva chamada.

Infelizmente é também importante referir que os objectivos cumpridos para o projeto ficaram bastante aquém do que era inicialmente esperado, na medida em que apenas foi possível instanciar os componentes base e fazer uma chamada SIP. Tal situação aconteceu pelo facto da tecnologia proposta no projeto estar desactualizada em termos de bibliotecas e de ser muito difícil encontrar recursos na Internet que ajudem a resolver os erros que surgem, o que causa atrasos muito grandes no projeto. A instalação do projeto foi a parte que mais tempo demorou o que condicionou todo o desenvolvimento do projeto e deixou muito pouco tempo para se aprofundar o conhecimento em IMS e implementar funcionalidades mais avançadas. Não possível ir mais além que o esperado pelo facto da instalação dos serviços ter gerado tantos problemas e atrasos inesperados. No entanto apesar disso sentimos que foi possível desenvolver competências importantes em termos de trabalho autónomo e resolução de problemas.

# Bibliografia

- [1] Gabriel Fartaria Ferreira. *Dissertação Serviços IP Multimédia em Redes VoIP/3G*. Universidade de Aveiro, 2008.
- [2] Kamailio. <https://www.kamailio.org/w/>.
- [3] Fraunhofer FOKUS NGNI. Open source ims. <http://openimscore.sourceforge.net/>.
- [4] Fraunhofer FOKUS NGNI. Openimscore instalation guide. [http://openimscore.sourceforge.net/?q=installation\\_guide](http://openimscore.sourceforge.net/?q=installation_guide).
- [5] Microsip. <https://www.microsip.org/>.
- [6] Mizudroid. [https://play.google.com/store/apps/details?id=com.mizuvoip.mizudroid.app&hl=pt\\_PT&gl=US](https://play.google.com/store/apps/details?id=com.mizuvoip.mizudroid.app&hl=pt_PT&gl=US).