

Comunicações Móveis

Projeto IMS

Miguel Cabral^[93091] , Rodrigo Santos^[93173]

Universidade de Aveiro
Departamento de Eletrónica Telecomunicações e Telemática



1 Introdução

O presente relatório foi realizado para fundamentar o projeto realizado no âmbito da cadeira de Comunicações Móveis do Mestrado Integrado em Engenharia de Computadores e Telemática, a decorrer no 1^o semestre do ano letivo 2021/2022. O projeto tem a finalidade da familiarização com o ecossistema IP Multimedia System (IMS) desenvolvido para que redes Universal Mobile Telecommunications System (UMTS) pudessem transportar tráfego multimédia a utilizadores mobile. O objetivo é configurar um servidor SIP, instanciar os componentes IMS (P-CSCF, I-CSCF, S-CSCF e HSS) e testar chamadas voz, videoconferência e outros serviços multimédia.

2 Teoria

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. O IMS pode ser dividido em três camadas:

- **Camada de Serviços** constituída por serviços aplicativos, *Web Servers* que oferecem basicamente serviços aos utilizadores;
- **Camada de Transporte** que suporta as arquiteturas das redes core de *General Packet Radio Service* (GPRS). É nesta camada que se situam as firewalls e as gateways que auxiliam na tradução de protocolos de tráfego provindo de redes de pacotes ou de circuitos;
- **Camada de Controlo** permite controlar e gerir sessões, assim como gerir e armazenar informação acerca dos seus utilizadores bem como fornecer serviços para gerir o acesso dos mesmos á rede.

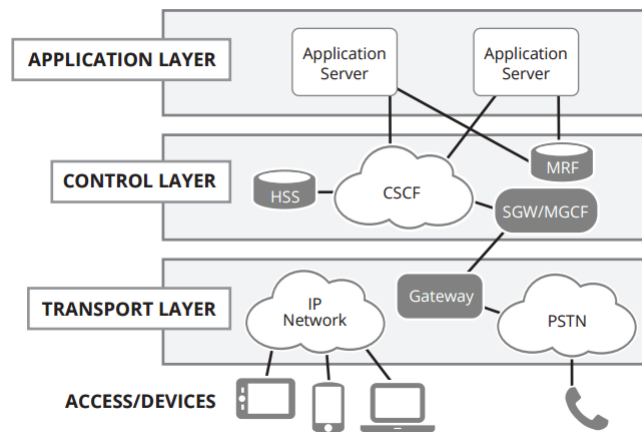


Fig. 1. Arquitetura base do IMS.

Algumas das vantagens do IMS são a qualidade de serviço (QoS) assim como uma grande variedade de serviços tais como mensagens de voz, texto, video, etc. É independente da rede de acesso, do dispositivo ou da aplicação. O facto de ser escalável e tão flexível também é uma das vantagens deste sistema pois facilmente permite a integração com outros domínios IMS. A principal desvantagem é ter uma implementação complexa devido ao conhecimento avançado sobre redes de comunicações móveis bem como sobre os diferentes componentes que integram a arquitetura base deste sistema.

A arquitetura do core IMS identificamos os seguintes componentes:

- **Call Session Control Function** (CSCF) envolvidos no processo de registo de terminais e estabelecimento de sessões. Existem três variantes de CSCF:
 - **Proxy Call Session Control Function** (P-CSCF) primeiro ponto de contacto do equipamento dos utilizadores (UE) com a rede IMS. Recebe e reencaminha os 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, faturação e utilização de recursos. Age como uma firewall do servidor.
 - **Serving Call Session Control Function** (S-CSCF) elemento central de toda a arquitectura do IMS. Atua como um servidor SIP gerindo perfis e sessões dos utilizadores.
- **Home Subscriber Server** (HSS) Base de dados fundamental para toda a informação dos utilizadores, como por exemplo, endereçamento, segurança e localização.

Para o controlo das sessões, o IMS usa o protocolo SIP que é um protocolo de sinalização usado para iniciar, manter e encerrar sessões em tempo real que incluem aplicações de voz, vídeo e mensagens.

3 Implementação

Uma das grandes dificuldades sentidas neste projeto foi a procura de versões de bibliotecas compatíveis e atualizadas com o OpenIMSCore visto que a sua implementação Open Source é de 2004-2008. O que implicou que grande parte do material encontrado estivesse desatualizado. Além disto, a escolha do ambiente distribuído foi outro problema porque havia versões que existiam num dos ambientes mas havia falta de outras bibliotecas.

3.1 Kamailio

Uma das tecnologias propostas foi o Kamailio que é um servidor SIP *Open Source* capaz de gerir milhares de estabelecimentos de chamadas por segundo. É escalável ao ponto de ser integrado em tecnologias como *Asterisk*, *FreeSWITCH* ou *SEMS* e também pode ser usado para plataformas de comunicações de tempo-real.

3.1.1 Instalação

A instalação foi feita no *Ubuntu 20.04 LTS*.

Instalação do MariaDB Database Server

É necessário uma base de dados para guardar os dados do hss e do icscf, decidimos usar MariaDB. Para instalar deve correr os seguinte comandos:

```
sudo apt install mariadb-server mariadb-client
sudo mysql_secure_installation
```

Instalação do Kamailio

Para instalar o kamailio deve-se primeiro fazer o download e adicionar a chave GPG para o repositório com :

```
wget -O- http://deb.kamailio.org/kamailiodebkey.gpg | sudo apt
-key add -
```

Depois criar o arquivo do repositório:

```
sudo vim /etc/apt/sources.list.d/kamailio.list
```

Inserir no ficheiro as seguintes linhas:

```
deb http://deb.kamailio.org/kamailio52 bionic main
deb-src http://deb.kamailio.org/kamailio52 bionic main
```

De seguida basta dar Update às packages do Ubuntu e instalar o kamailio com:

```
sudo apt update
sudo apt install kamailio kamailio-mysql-modules kamailio-
websocket-modules
```

Para verificar se a instalação foi bem sucedida correr o comando:

```
kamailio -V
```

Depois deve abrir o ficheiro de configuração:

```
sudo vim /etc/kamailio/kamctlrc
```

E alterar as linhas seguintes, onde SIP_DOMAIN será o servidor DNS configurado mais à frente:

```
The Kamailio configuration file for the control tools.
#
# Here you can set variables used in the kamctl and ...
# scripts. Per default all variables here are ...
# will use their internal default values.

## your SIP domain
SIP_DOMAIN=mnc001.mcc001.3gppnetwork.org

## chrooted directory
# $CHROOT_DIR="/path/to/chrooted/directory"

## database type: MYSQL, PGSQL, ORACLE, DB_BERKELEY ...
# by default none is loaded
#
# If you want to setup a database with kamdbctl, ...
# this parameter.
DBENGINE=MYSQL
```

Para criar a base de dados:

```
kamdbctl create
```

Se ocorrer um **erro de acesso negado** para o *root@localhost*, siga os passos seguintes para o resolver, caso não ocorra nenhum erro ignorar os próximos 3 passos:

Primeiro dar login no servidor MariaBD:

```
sudo mysql -u root -p
```

Depois de estar ligado ao servidor da base de dados correr os comandos: Primeiro dar login no servidor MariaBD:

```
use mysql;
update user set plugin='' where User='root';
flush privileges;
exit
```

Depois reinicar o servidor MariaDB:

```
sudo systemctl restart mariadb.service
```

Correr novamente o comando para criar a base de dados do kamailio e responder com as configurações seguintes:

```
Enter character set name:
latin1
INFO: creating database kamailio .
INFO: granting privileges to database kamailio .
INFO: creating standard tables into kamailio .
INFO: Core Kamailio tables succesfully created.
Install presence related tables? (y/n): y
INFO: creating presence tables into kamailio .
INFO: Presence tables succesfully created.
Install tables for imc cpl siptrace domainpolicy
        carrierroute drouting userblacklist htable
        purple uac pipelimit mtree sca mohqueue
        rtpproxy rtpengine? (y/n): y
INFO: creating extra tables into kamailio .
INFO: Extra tables succesfully created.
Install tables for uid_auth_db uid_avp_db uid_domain
        uid_gflags uid_uri_db? (y/n): y
INFO: creating uid tables into kamailio .
INFO: UID tables succesfully created.
```

Depois abrir o ficheiro de configuração :

```
sudo vim /etc/kamailio/kamailio.cfg
```

Adicionar as seguintes linhas depois de *#!/KAMAILIO*:

```
#!/define WITHMYSQL
#!/define WITHAUTH
#!/define WITHUSRLOCDB
#!/define WITHACCDB
```

Reiniciar o kamailio:

```
sudo systemctl restart kamailio
```

Instalação do bind9 DNS Server

Para instalar o bind9

```
sudo apt-get install bind9
```

Para definir a zona do domínio abrir o ficheiro named.conf.local:

```
sudo vim /etc/bind/named.conf.local
```

E adicionar as seguintes linhas:

```
zone "mnc001.mcc001.3gppnetwork.org" {
    type master;
    file "/etc/bind/mnc001.mcc001.3gppnetwork.org";
};
```

De seguida é necessário configurar a forward and reverse lookup zone para o domínio o kamailio tem um ficheiro exemplo para o efeito, para o copiar para a pasta bind corra o seguinte comando:

```
sudo cp /usr/share/doc/kamailio/examples/ims/ims.dnszone/
mnc001.mcc001.3gppnetwork.org /etc/bind/
```

Depois abrir o ficheiro copiado e alterar o IP 127.0.0.1 para o IPV4 Público da máquina 192.168.1.200 no nosso caso:

```
sudo sed -i 's/127\.0\.0\.1/192\.168\.1\.200/g' /etc/bind/
mnc001.mcc001.3gppnetwork.org
```

Depois abrir o ficheiro resolv.conf ¹:

```
sudo vim /etc/resolv.conf
```

E adicionar o seguinte:

```
domain mnc001.mcc001.3gppnetwork.org
search mnc001.mcc001.3gppnetwork.org
nameserver 192.168.1.200
```

Reiniciar o bind9:

```
sudo systemctl restart bind9
```

Neste momento deve ser possível pingar o servidor DNS.

¹ Uma vez que o ficheiro /etc/resolv.conf é alterado sempre que se reinicia a máquina foi criado um script para alterar este ficheiro, esse script está em Anexo

Instalação do Siremis

O Siremis é uma interface gráfica para o Kamailio SIP server onde é possível criar e gerenciar perfis de usuários, regras de routing e comunicação com o servidor SIP, para a sua configuração é preciso um servidor HTTP no nosso caso usamos o Apache2.

Para instalar o servidor Apache2:

```
sudo apt install apache2
```

Para instalar o servidor PHP 7.2 e os módulos:

```
sudo apt-get install software-properties-common
sudo add-apt-repository ppa:ondrej/php
sudo apt update
sudo apt install php7.2 libapache2-mod-php7.2 php7.2-common
php7.2-gmp php7.2-curl php7.2-intl php7.2-mbstring php7.2-
xmlrpc php7.2-mysql php7.2-gd php7.2-imagick php-pear php7
.2-xml php7.2-cli php7.2-zip php7.2-sqlite
```

Depois da instalação do php 7.2 abrir o ficheiro de configuração para o Apache2

```
sudo vim /etc/php/7.2/apache2/php.ini
```

Depois alterar as seguintes linhas

```
file_uploads = On
allow_url_fopen = On
short_open_tag = On
memory_limit = 256M
upload_max_filesize = 100M
max_execution_time = 360
max_input_vars = 1500
date.timezone = Europe/Lisbon
```

Dar restart ao Apache2:

```
sudo systemctl restart apache2.service
```

Neste momento o servidor HTTP deve estar operacional

Para instalar o siremis é preciso instalar o XML_RPC, para isso é só correr o seguinte comando:

```
sudo apt-get install php-xmlrpc
```

De seguida, mudar o diretório para o do Apache web root, e fazer git clone do repositório do siremis:

```
cd /var/www/
sudo apt install git
sudo git clone https://github.com/asipto/siremis
```

Depois para compilar é so correr os seguintes comandos:

```
cd /var/www/siremis
sudo make prepare24
sudo make chown
```

Depois da compilação é necessário criar o ficheiro de configuração do servidor Apache2 para o siremis:

```
sudo vim /etc/apache2/sites-available/siremis.conf
```

E colar a seguinte configuração onde o ServerName é o IPV4 Publico da máquina:

```
<VirtualHost *:80>
    ServerAdmin admin@example.com
    DocumentRoot /var/www/siremis/siremis
    Alias /siremis "/var/www/siremis/siremis"
    ServerName 192.168.1.200
    ServerAlias www.kamailio.example.com

    <Directory "/var/www/siremis/siremis">
        Options Indexes FollowSymLinks MultiViews
        AllowOverride All
        Require all granted

        <FilesMatch "\.xml$">
            Require all denied
        </FilesMatch>
        <FilesMatch "\.inc$">
            Require all denied
        </FilesMatch>
    </Directory>

    ErrorLog ${APACHELOG_DIR}/error.log
    CustomLog ${APACHELOG_DIR}/access.log combined

</VirtualHost>
```

Depois é necessário ativar o VirtualHost com:

```
sudo a2ensite siremis.conf
sudo a2enmod rewrite
sudo systemctl restart apache2.service
```

Finalmente é necessário garantir que o siremis tem acesso à base de dados:

```
sudo mysql -u root -p
```

```
GRANT ALL PRIVILEGES ON siremis.* TO siremis@localhost
IDENTIFIED BY 'password';
FLUSH PRIVILEGES;
EXIT;
```

Para prosseguir com a instalação é só abrir o browser e colocar na barra de endereços o ServerName se tudo estiver bem aparecerá a seguinte página:

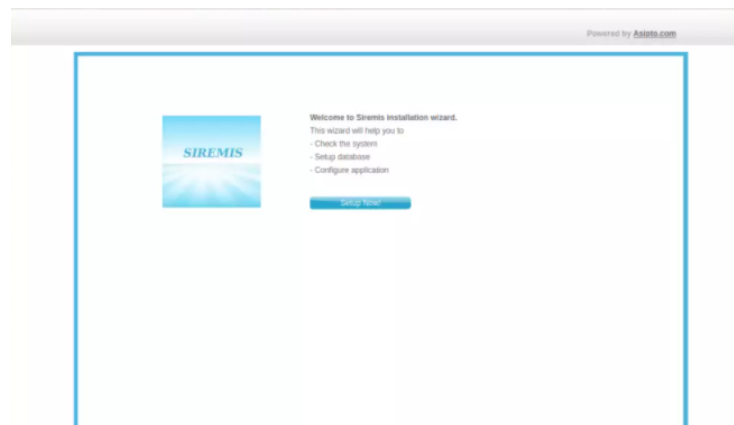


Fig. 2. Siremis Beginning of installation wizard

Clicar em *Setup Now* e prosseguir com a instalação

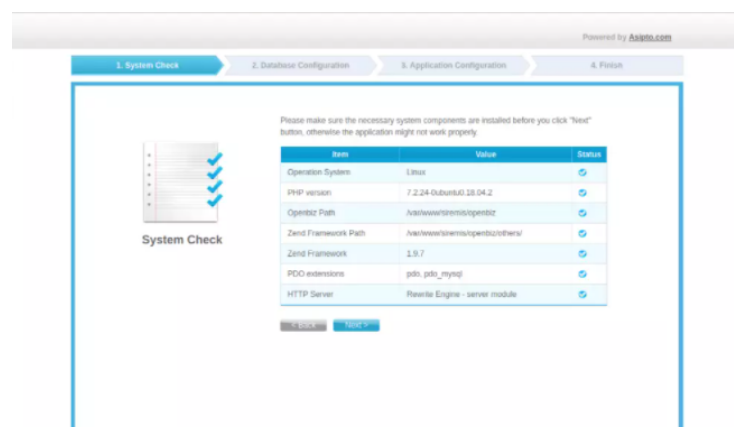


Fig. 3. Siremis installation wizard system check

Depois de clicar em *Next* colocar as informações da base de dados apresentadas na próxima figura

The screenshot shows the 'Database Configuration' step of the Siremis installation wizard. It is divided into two main sections: Siremis DB and SIP DB. Both are configured with MySQL on localhost:3306. The Siremis DB name is 'siremis' and the SIP DB name is 'kamailio'. There are checkboxes for 'Create Siremis DB', 'Update SIP DB', 'Import Default Data', and 'Replace DB Config'. A 'Next' button is visible at the bottom right.

Fig. 4. Siremis installation wizard database configuration

Se não houver erros a instalação está concluída e é possível fazer login com as credenciais :

```
Username: admin
Password: admin
```

E é possível ver a página inicial do Siremis.

Agora basta dar restart a tudo e verificar se está tudo a correr:

```
sudo systemctl restart bind9
sudo systemctl restart apache2
sudo systemctl restart kamailio
#Para verificar se esta tudo a correr sem erros:
sudo journalctl -xe
```

3.2 OpenIMScore

Outra das tecnologias sugeridas foi o OpenIMScore que é uma implementação *Open Source* das funcionalidades de *Call Session Control Functions* (CSCFs) que em conjunto com a *Home Subscriber Server* (HSS) constituem um core funcional do IMS.

3.2.1 Instalação

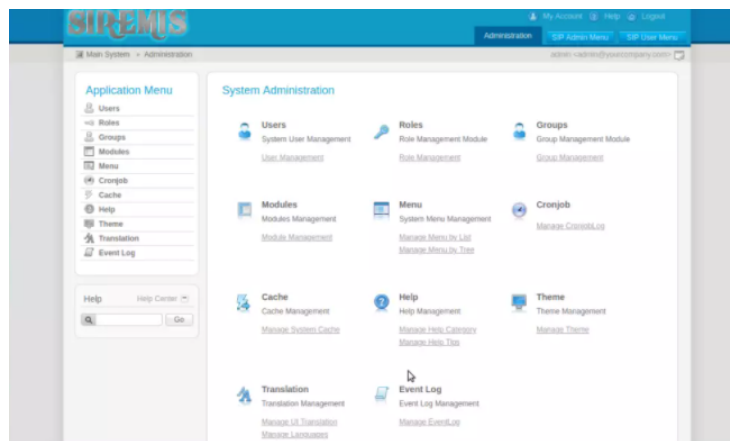


Fig. 5. Siremis Initial Page

As bibliotecas instaladas foram as seguintes:

```
sudo apt-get install libcurl4-gnutls-dev
sudo apt-get install bison
sudo apt-get install curl
sudo apt-get install debhelper cdb libtinfo build-essential
sudo apt-get install devscripts pbuilder dh-make debootstrap
sudo apt-get install libxml2-dev libmysqlclient-dev ant
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
```

Instalação do OpenIMSCore

Como foi necessário um servidor Domain Name System (DNS) adquirimos o *Bind9* e para compilação usámos o GNU Compiler Collection (GCC) na versão 4.9.2. Seguidamente, foi necessário criar o repositório para obter o código source do OpenIMSCore. Então optámos pelo diretório */opt/OpenIMSCore* que contém mais dois diretórios: *ser_ims* e *FHoSS*. No *ser_ims* fica o código que vai implementar os componentes dos serviços do IMS exceto o HSS que vai ser implementado no diretório *FHoSS*.

```
sudo mkdir /opt/OpenIMSCore
cd /opt/OpenIMSCore
#Criar directorio e compilar
sudo mkdir ser_ims
```

```

sudo svn checkout https://svn.code.sf.net/p/openimscore/
code/ser_ims/trunk ser_ims
cd ser_ims
make install--libs all
cd ..
#Criar diretorio e compilar
sudo mkdir FHoSS
sudo svn checkout https://svn.code.sf.net/p/openimscore/
code/FHoSS/trunk FHoSS
cd FHoSS
ant compile
ant deploy
cd ..

```

Instalação do bind9 DNS Server

A configuração do DNS é muito simples. Primeiro, é necessário copiar o ficheiro *open-ims.dnszone* para o diretório destino.

```

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

```

Para definir a zona do domínio, abrir o ficheiro *named.conf.local*:

```

sudo vim /etc/bind/named.conf.local

```

E adicionar as seguintes linhas:

```

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

```

Depois abrir o ficheiro *resolv.conf*:

```

sudo vim /etc/resolv.conf

```

E adicionar o seguinte:

```

domain open-ims.test
search open-ims.test
nameserver 192.168.1.200

```

Reiniciar o bind9:

```

sudo systemctl restart bind9

```

Para garantir que está tudo funcional basta fazer um *ping* para os domínios *pcscf.open-ims.test* ou *icscf.open-ims.test*.

Configuração do MySQL

Seguidamente, é preciso configurar o ambiente de desenvolvimento para podermos conseguir executar o projeto de forma correta. Para isso é necessário copiar os ficheiros Structured Query Language (SQL) do projeto para o servidor *MySQL*. Para tal, os seguintes comandos devem ser executados:

```
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
```

Verificando se as bases de dados foram corretamente copiadas para o servidor é necessário executar:

```
#No diretorio /opt/OpenIMSCore/
sudo mysql -u root -p
mysql> show databases;
```

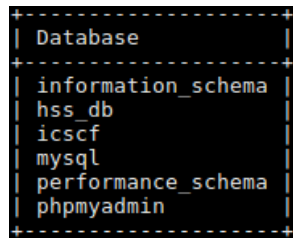


Fig. 6. Resultado esperado do MySQL.

Configuração do IMS Core

Finalmente, copiar os ficheiros para o diretório base do projeto:

```
cp ser_ims/cfg/*.cfg .
cp ser_ims/cfg/*.xml .
cp ser_ims/cfg/*.sh .
```

O ficheiro *startup.sh* deve ainda ser alterado na linha *\$JAVA_HOME/usr/bin/java -cp* além de que a variável *JAVA_HOME* deve ser *export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-i386/jre*.

Estando tudo configurado, falta só iniciar os scripts finais de cada componente do IMS:

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

cd FHoSS/deploy/
sudo ./startup.sh
```

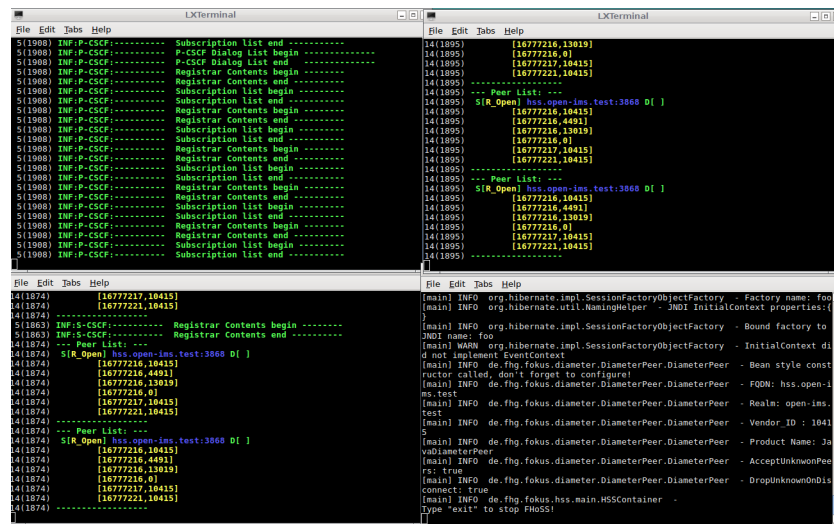


Fig. 7. Serviços P-CSCF, I-CSCF, S-CSCF e HSS.

<div><div><div></div></div><div></div></div>		FHoSS - The FOKUS Home Subscriber Server (Rel. 7)																	
HOME		USER IDENTITIES SERVICES NETWORK CONFIGURATION STATISTICS																	
User identities		IMS Subscription - Search Results																	
• All Subscriptions Search Create		<table><tr><th>#</th><th>Name</th><th>IMS Name</th><th>Subscriber Name</th></tr><tr><td>1</td><td>alice</td><td>alice@fokus.org.uk</td><td>alice@fokus.org.uk</td></tr><tr><td>2</td><td>bob</td><td>bob@fokus.org.uk</td><td>bob@fokus.org.uk</td></tr><tr><td>3</td><td>carol</td><td>carol@fokus.org.uk</td><td>carol@fokus.org.uk</td></tr></table>		#	Name	IMS Name	Subscriber Name	1	alice	alice@fokus.org.uk	alice@fokus.org.uk	2	bob	bob@fokus.org.uk	bob@fokus.org.uk	3	carol	carol@fokus.org.uk	carol@fokus.org.uk
#	Name	IMS Name	Subscriber Name																
1	alice	alice@fokus.org.uk	alice@fokus.org.uk																
2	bob	bob@fokus.org.uk	bob@fokus.org.uk																
3	carol	carol@fokus.org.uk	carol@fokus.org.uk																
• Private Identity Search Create		Results per page 10 20 50																	
• Public User Identity Search Create																			

Fig. 8. Interface gráfica do HSS.

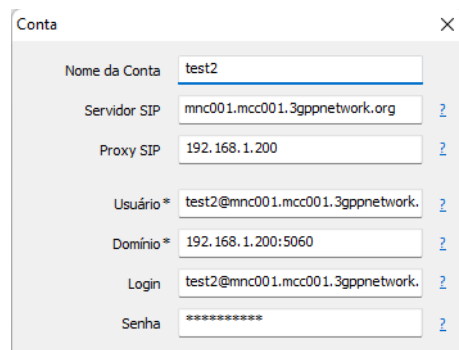
4 Resultados

4.1 Kamailio

Para os testes com o Kamailio o servidor estava a correr numa máquina virtual Linux 20.04 LTS com o adaptador de rede configurado como **Bridge Adapter** para que a máquina esteja ligada à placa de rede de modo a que outros dispositivos na mesma rede consigam estabelecer conexão com o servidor. Foi também configurado no router um ip estático **192.168.1.200** para a mesma. O Kamailio está a correr sem as componentes CSCF ativas.

4.1.1 Registo de Utilizador

Para este teste como UE (User Equipment) foi usado o MicroSIP numa máquina Windows ligada à mesma rede do servidor com ip **192.168.1.165**. Foi criado um novo usuário no Siremis o *test2* e foi registado no MicroSIP com as credenciais presentes na figura seguinte.



Conta	
Nome da Conta	test2
Servidor SIP	mnc001.mcc001.3gppnetwork.org
Proxy SIP	192.168.1.200
Usuário *	test2@mnc001.mcc001.3gppnetwork.
Domínio *	192.168.1.200:5060
Login	test2@mnc001.mcc001.3gppnetwork.
Senha	*****

Fig. 9. Configuração da conta test2 no MicroSIP

Foi usado o Wireshark na máquina onde o kamailio está a correr para capturar os pacotes SIP. O que acontece é apresentado a seguir estando o fluxo de pacotes presente na figura10.

1. O UE enviar um pacote SIP REGISTER com o seu usuário para o SIP SERVER, de notar que como as componentes CSCF não foram configuradas esta ligação é feita diretamente com o servidor e não com o P-CSCF.
2. O SIP SERVER responde com um Challenge pedindo as credenciais.
3. O UE responde com o as credenciais (User ID e Password).
4. O SIP SERVER valida as credenciais com sucesso, regista o utilizador e envia uma resposta 200 OK.

No.	Time	Source	Destination	Protocol	Length	Info
5	11.539991478	192.168.1.165	192.168.1.200	SIP	642	Request: REGISTER sip:mnc001.mcc001.3gppnetwork.org (1 binding)
6	11.539993167	192.168.1.200	192.168.1.165	SIP	602	Status: 401 Unauthorized
7	11.540993945	192.168.1.165	192.168.1.200	SIP	879	Request: REGISTER sip:mnc001.mcc001.3gppnetwork.org (1 binding)
8	11.540704483	192.168.1.200	192.168.1.165	SIP	543	Status: 200 OK (1 binding)


```

Internet Protocol Version 4, Src: 192.168.1.165, Dst: 192.168.1.200
User Datagram Protocol, Src Port: 53771, Dst Port: 5060
Session Initiation Protocol (REGISTER)
  Request-Line: REGISTER sip:mnc001.mcc001.3gppnetwork.org SIP/2.0
  Message Header
    Via: SIP/2.0/UDP 192.168.1.165:53771;rport;branch=z9hG4bKPj9ffb15aa05a4351a294acfdccfd6215
    Route: <sip:192.168.1.200;lr>
    Max-Forwards: 70
    From: <sip:test2@mnc001.mcc001.3gppnetwork.org>;tag=4815dcc9f7634e60a9e1b91ab90afae4
    To: <sip:test2@mnc001.mcc001.3gppnetwork.org>
    Call-ID: c85ffc55eb9e42548339f47a7426ab95
    [Generated Call-ID: c85ffc55eb9e42548339f47a7426ab95]
    CSeq: 42504 REGISTER
    User-Agent: MicroSIP/3.20.7
    Contact: <sip:test2@192.168.1.165:53771;ob>
    Expires: 300
    Allow: PRACK, INVITE, ACK, BYE, CANCEL, UPDATE, INFO, SUBSCRIBE, NOTIFY, REFER, MESSAGE, OPTIONS
    Content-Length: 0

```

Fig. 10. Captura Wireshark para registo do utilizador test2

4.1.2 Chamada SIP

Para este teste foi usado o MicroSIP em duas máquina Windows ligadas à mesma rede do servidor com ip **192.168.1.165** com o usuário test2 e **192.168.1.215** com o usuário rodrigo registado da mesma maneira que o anterior. Foi usado o Wireshark na máquina onde o kamailio está a correr para capturar os pacotes SIP. O que acontece é apresentado de seguida estando o fluxo de pacotes presente na figura 11.

1. Foi enviado um pacote SIP/SDP INVITE do UE do user test2 ao user rodrigo@mnc001.mcc001.3gppnetwork.org que representa o numero de telefone para o qual está a tentar ligar no domínio do servidor.
2. O servidor responde com um Proxy Authentication Required a informar que o user test2 tem que se autenticar antes de iniciar a chamada.
3. O UE do test2 responde com um ACK para informar o servidor que recebeu o seu pedido.
4. O UE do test2 envia novamente um pacote SIP/SDP INVITE mas desta vez com as credenciais como se pode ver na figura 11.
5. O servidor informa o UE do test2 que recebeu a mensagem e vai processá-la
6. O servidor informa o UE do rodrigo que tem um pedido pendente e envia um SIP/SDP INVITE com o IP e o User ID do user test2
7. O UE do rodrigo informa o servidor que recebeu e vai processar o pedido
8. O UE do rodrigo processa o pedido e começa a tocar informando com um 180 Ringing o servidor para que ele tenha conhecimento do evento.
9. O servidor informa o UE do test2 que o UE do rodrigo processou o pedido e está a tocar
10. O UE do rodrigo atende a chamada e envia um pacote SIP/SDP 200 OK
11. O servidor informa o UE do test2 que o UE do rodrigo atendeu a chamada
12. O UE do test2 informa o servidor com um ACK que recebeu a mensagem e o servidor informa o UE do rodrigo e a chamada começa.

13. A chamada começa os pacotes passam a ser protocolos *Real-Time Transport Protocol* (RTP) que contém informação de audio que está a ser trocado e protocolos *Real-Time Transport Control Protocol* (RTCP) que contém informação estatística da chamada como se pode ver na figura 12.
14. O UE do test2 faz um pedido para terminal a chamada.
15. O servidor informa o UE do rodrigo que há um pedido para a chamada terminar.
16. O UE do Rodrigo responde com um SIP 200 OK, o servidor informa o UE do test2 e a chamada é terminada.

15	16.809452504	192.168.1.165	192.168.1.200	SIP/SDP	1093 Request: INVITE sip:rodrigo@mnc001.mcc001.3gppnetwork.org
16	16.809736726	192.168.1.200	192.168.1.165	SIP	621 Status: 407 Proxy Authentication Required
17	16.810027138	192.168.1.165	192.168.1.200	SIP	502 Request: ACK sip:rodrigo@mnc001.mcc001.3gppnetwork.org
18	16.810083641	192.168.1.165	192.168.1.200	SIP/SDP	1344 Request: INVITE sip:rodrigo@mnc001.mcc001.3gppnetwork.org
19	16.810606707	192.168.1.200	192.168.1.165	SIP	476 Status: 100 trying -- your call is important to us
20	16.810785472	192.168.1.200	192.168.1.215	SIP/SDP	1207 Request: INVITE sip:rodrigo@192.168.1.215:56082;ob
21	16.823153642	192.168.1.215	192.168.1.200	SIP	550 Status: 100 Trying
22	16.823293554	192.168.1.215	192.168.1.200	SIP	733 Status: 180 Ringing
23	16.823434981	192.168.1.200	192.168.1.165	SIP	625 Status: 180 Ringing
28	19.765974012	192.168.1.215	192.168.1.200	SIP/SDP	1182 Status: 200 OK (INVITE)
29	19.766229487	192.168.1.200	192.168.1.165	SIP/SDP	1074 Status: 200 OK (INVITE)
30	19.770509284	192.168.1.165	192.168.1.200	SIP	486 Request: ACK sip:rodrigo@192.168.1.215:56082;ob
31	19.770664195	192.168.1.200	192.168.1.215	SIP	569 Request: ACK sip:rodrigo@192.168.1.215:56082;ob
52	26.270593300	192.168.1.165	192.168.1.200	SIP	515 Request: BYE sip:rodrigo@192.168.1.215:56082;ob
53	26.270891582	192.168.1.200	192.168.1.215	SIP	598 Request: BYE sip:rodrigo@192.168.1.215:56082;ob
54	26.275944948	192.168.1.215	192.168.1.200	SIP	542 Status: 200 OK (BYE)
55	26.276136846	192.168.1.200	192.168.1.165	SIP	434 Status: 200 OK (BYE)

Min-SE: 90
User-Agent: MicroSIP/3.20.7
> [truncated]Proxy-Authorization: Digest username="test2@mnc001.mcc001.3gppnetwork.org", realm="mnc001.mcc001.3gppnetwork.org", nonce="
Content-Type: application/sdp
Content-Length: 342
▼ Message Body
> Session Description Protocol

Fig. 11. Captura Wireshark para chamada de voz do lado do servidor

138	23.482637	192.168.1.165	192.168.1.215	RTP	214 PT=ITU-T G.711 PCMA, SSRC=0x1940095F, Seq=7721, Time=1600
139	23.496651	192.168.1.215	192.168.1.165	RTP	214 PT=ITU-T G.711 PCMA, SSRC=0x3A6122CD, Seq=11046, Time=1760
140	23.504185	192.168.1.165	192.168.1.215	RTP	214 PT=ITU-T G.711 PCMA, SSRC=0x1940095F, Seq=7722, Time=1760
141	23.513440	192.168.1.165	192.168.1.215	RTP	214 PT=ITU-T G.711 PCMA, SSRC=0x1940095F, Seq=7723, Time=1920
142	23.514947	192.168.1.215	192.168.1.165	RTP	214 PT=ITU-T G.711 PCMA, SSRC=0x3A6122CD, Seq=11047, Time=1920
143	23.520163	192.168.1.165	192.168.1.215	RTCP	122 Sender Report Source description
144	23.534811	192.168.1.165	192.168.1.215	RTP	214 PT=ITU-T G.711 PCMA, SSRC=0x1940095F, Seq=7724, Time=2080

Fig. 12. Captura Wireshark para chamada de voz do lado do UE

Também foi feita uma chamada de vídeo, o fluxo de pacotes é o mesmo tirando a chamada que é feita através de pacotes UDP como se pode ver na figura 13.

No.	Time	Source	Destination	Protocol	Length	Info
1602	5.050402	192.168.1.215	192.168.1.165	UDP	223	4010 → 4002 Len=181
1603	5.050404	192.168.1.165	192.168.1.215	UDP	83	4002 → 4010 Len=41
1604	5.050920	192.168.1.165	192.168.1.215	UDP	113	4002 → 4010 Len=71
1605	5.052479	192.168.1.215	192.168.1.165	UDP	182	4010 → 4002 Len=140
1606	5.052558	192.168.1.165	192.168.1.215	UDP	214	4000 → 4000 Len=172
1607	5.052736	192.168.1.215	192.168.1.165	UDP	191	4010 → 4002 Len=149
1608	5.059609	192.168.1.215	192.168.1.165	UDP	187	4010 → 4002 Len=145
1609	5.059887	192.168.1.215	192.168.1.165	UDP	205	4010 → 4002 Len=163
1610	5.059887	192.168.1.215	192.168.1.165	UDP	239	4010 → 4002 Len=197
1611	5.059887	192.168.1.215	192.168.1.165	UDP	305	4010 → 4002 Len=263

> Frame 1607: 191 bytes on wire (1528 bits), 191 bytes captured (1528 bits) on interface \Device\NPF_{42142A69-B923-4569-9050-3D84EAB007B6}, id 0
 > Ethernet II, Src: IntelCor_56:92:53 (f4:d1:00:56:92:53), Dst: IntelCor_92:8c:c2 (84:1b:77:92:8c:c2)
 > Internet Protocol Version 4, Src: 192.168.1.215, Dst: 192.168.1.165
 > User Datagram Protocol, Src Port: 4010, Dst Port: 4002
 > Data (149 bytes)

Fig. 13. Captura Wireshark para chamada de video do lado do UE

4.2 OpenIMSCore

Para os testes com o OpenIMSCore o servidor estava a correr numa máquina virtual Debian 8 LXDE com o adaptador de rede configurado como **Bridge Adapter** para que a máquina esteja ligada à placa de rede de modo a que outros dispositivos na mesma rede consigam estabelecer conexão com o servidor. Foi também configurado no router com um ip estático **192.168.1.155** para a mesma. O OpenIMSCore está a correr com as componentes CSCF ativas, nomeadamente, os serviços I-CSCF, P-CSCF e S-CSCF e a base de dados HSS.

4.2.1 Registo de Utilizador

Para este teste foi usado o MicroSIP numa máquina Windows ligada à mesma rede do servidor com ip **192.168.1.165**. Foi criado um novo usuário no *FOKUS Home Subscriber Server* (FHoSS), o *miguel*, e foi registado no MicroSIP com as credenciais presentes na figura seguinte.

The screenshot shows the 'Conta' (Account) configuration window in MicroSIP. The fields are filled with the following information:

- Nome da Conta: miguel
- Servidor SIP: open-ims.test
- Proxy SIP: 192.168.1.155:4060
- Usuário*: miguel@open-ims.test
- Domínio*: 192.168.1.155
- Login: miguel@open-ims.test
- Senha: *****

Fig. 14. Configuração da conta miguel no MicroSIP

Foi usado o Wireshark na máquina onde o OpenIMSCore está a correr para capturar os pacotes SIP e o fluxo de pacotes é o seguinte:

1. O UE envia um pacote SIP request REGISTER com o seu usuário para o P-CSCF.
2. O SIP SERVER responde com um pacote *Status:401 Unauthorized - challenging the UE* pedindo as credenciais. A autenticação será criada a partir de um *hash MD5* gerado a partir da password e do nome de utilizador do cliente
3. O UE responde com as credenciais (User ID e Password) num novo request REGISTER.
4. O proxy valida as credenciais com sucesso, regista o utilizador e envia uma resposta 200 OK.

Por sua vez, do lado do servidor o P-CSCF reencaminha o SIP REGISTER para o I-CSCF que envia um *User Authorization Request* ao HSS que irá responder com o S-CSCF onde o utilizador vai ser registado, isto se o pedido for aceite. Como só existe um core IMS a troca de pacotes parece ser simples, mas é bastante complexa noutros cenários.

67	51.70530700	192.168.1.165	192.168.1.155	SIP	602 Request: REGISTER sip:open-ims.test (1 binding)
68	51.77014100	192.168.1.155	192.168.1.165	SIP	998 Status: 401 Unauthorized - Challenging the UE
69	51.77042400	192.168.1.165	192.168.1.155	SIP	873 Request: REGISTER sip:open-ims.test (1 binding)
70	51.85733900	192.168.1.155	192.168.1.165	SIP	1033 Status: 200 OK - SAR succesful and registrar saved (1 binding)

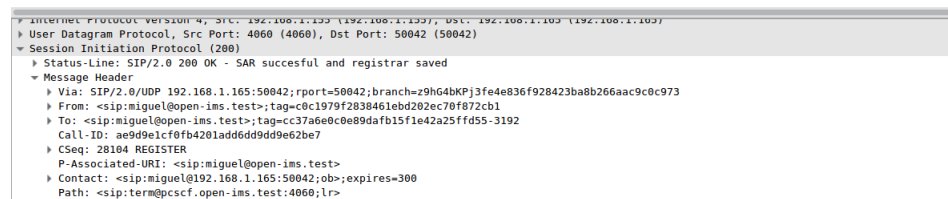


Fig. 15. Captura Wireshark para registo do utilizador miguel

4.2.2 Chamada SIP

Para este teste foi usado o MicroSIP em duas máquinas Windows ligadas à mesma rede do servidor com ip **192.168.1.165** com o usuário miguel e **192.168.1.215**

com o usuário alice registado da mesma maneira que o anterior. Foi usado o Wireshark na máquina onde o OpenIMSCore está a correr para capturar os pacotes SIP. O que acontece com o fluxo de pacotes é o seguinte:

1. Foi enviado um pacote SIP/SDP INVITE do UE do user miguel ao user alice@open-ims.test que representa o numero de telefone para o qual está a tentar ligar no domínio do servidor.
2. O servidor informa o UE do miguel que recebeu a mensagem e vai processá-la
3. O servidor informa o UE da alice que tem um pedido pendente e envia um SIP/SDP INVITE com o IP e o User ID do user miguel
4. O UE da alice informa o servidor que recebeu e vai processar o pedido
5. O UE da alice processa o pedido e começa a trocar informando com um 180 Ringing o servidor para que ele tenha conhecimento do evento.
6. Assim que a chamada é atendida os pacotes passam a ser protocolos *Real-Time Transport Protocol* (RTP) que contém informação de audio que está a ser trocado e protocolos *Real-Time Transport Control Protocol* (RTCP) que contém informação estatística da chamada.
7. No fim da chamada é enviado um SIP BYE tanto ao utilizador como ao servidor.

No.	Time	Source	Destination	Protocol	Length	Info
12	7.255734	192.168.1.165	192.168.1.155	SIP	602	Request: REGISTER sip:open-ims.test (1 binding)
13	7.282184	192.168.1.155	192.168.1.165	SIP	998	Status: 401 Unauthorized - Challenging the UE
14	7.282593	192.168.1.165	192.168.1.155	SIP	873	Request: REGISTER sip:open-ims.test (1 binding)
15	7.329251	192.168.1.155	192.168.1.165	SIP	1033	Status: 200 OK - SAR successful and registrar saved (REGISTER) (1 binding)
32	16.075302	192.168.1.215	192.168.1.155	SIP	590	Request: REGISTER sip:open-ims.test (1 binding)
33	16.102804	192.168.1.155	192.168.1.215	SIP	996	Status: 401 Unauthorized - Challenging the UE
34	16.106042	192.168.1.215	192.168.1.155	SIP	869	Request: REGISTER sip:open-ims.test (1 binding)
35	16.146325	192.168.1.155	192.168.1.215	SIP	1029	Status: 200 OK - SAR successful and registrar saved (REGISTER) (1 binding)
38	19.244936	192.168.1.165	192.168.1.155	SIP/SDP	1095	Request: INVITE sip:alice@open-ims.test
39	19.245459	192.168.1.155	192.168.1.165	SIP	630	Status: 180 trying -- your call is important to us
41	19.246532	192.168.1.155	192.168.1.215	SIP/SDP	269	Request: INVITE sip:alice@192.168.1.215:64613;ob
42	19.266455	192.168.1.215	192.168.1.155	SIP	881	Status: 100 Trying
43	19.266517	192.168.1.215	192.168.1.155	SIP	1062	Status: 180 Ringing
44	19.267142	192.168.1.155	192.168.1.165	SIP	755	Status: 180 Ringing
56	24.307941	192.168.1.215	192.168.1.155	SIP/SDP	1510	Status: 200 OK (INVITE)
57	24.308582	192.168.1.155	192.168.1.165	SIP/SDP	1203	Status: 200 OK (INVITE)
59	24.611043	192.168.1.215	192.168.1.155	SIP/SDP	1510	Status: 200 OK (INVITE)
60	24.812214	192.168.1.155	192.168.1.165	SIP/SDP	1203	Status: 200 OK (INVITE)
61	25.810397	192.168.1.215	192.168.1.155	SIP/SDP	1510	Status: 200 OK (INVITE)
62	25.811484	192.168.1.155	192.168.1.165	SIP/SDP	1203	Status: 200 OK (INVITE)
65	27.809392	192.168.1.215	192.168.1.155	SIP/SDP	1510	Status: 200 OK (INVITE)
66	27.810302	192.168.1.155	192.168.1.165	SIP/SDP	1203	Status: 200 OK (INVITE)
74	31.812612	192.168.1.215	192.168.1.155	SIP/SDP	1510	Status: 200 OK (INVITE)
75	31.812800	192.168.1.155	192.168.1.165	SIP/SDP	1185	Status: 200 OK (INVITE)

```

> User Datagram Protocol, Src Port: 64613, Dst Port: 4060
▼ Session Initiation Protocol (100)
  > Status-Line: SIP/2.0 100 Trying
  ▼ Message Header
    > Via: SIP/2.0/UDP 192.168.1.155:4060;received=192.168.1.155;branch=z9hG4bK54fe.946f6c27.0
    > Via: SIP/2.0/UDP 192.168.1.155:6060;rport=6060;received=192.168.1.155;branch=z9hG4bK54fe.a7173c17.0
    > Via: SIP/2.0/UDP 192.168.1.155:6060;branch=z9hG4bK54fe.97173c17.0
    > Via: SIP/2.0/UDP 192.168.1.155:4060;branch=z9hG4bK54fe.846f6c27.0
    > Via: SIP/2.0/UDP 192.168.1.165:65519;rport=65519;branch=z9hG4bKpJ977ab987f2b849efb77f074c01f05ad9
    > Record-Route: <sip:mt@pcscf.open-ims.test:4060;lr>
    > Record-Route: <sip:mt@scscf.open-ims.test:8060;lr>
    > Record-Route: <sip:mo@pcscf.open-ims.test:8060;lr>
    > Record-Route: <sip:mo@pcscf.open-ims.test:4060;lr>
    Call-ID: 2a9c71bbf15a45f999906054e6a69fc4
    [Generated Call-ID: 2a9c71bbf15a45f999906054e6a69fc4]
    > From: <sip:miguel@open-ims.test>;tag=f6010310d9b9480c8f21648562b41437
    > To: <sip:alice@open-ims.test>
    > CSeq: 32130 INVITE
    Content-Length: 0

```

Fig. 16. Fluxo de pacotes da chamada SIP

5 Discussão

Analisando os resultados obtidos, conseguimos instanciar os componentes do IMS numa implementação *Open Source* de funcionalidades de CSCFs combinada com o HSS, designado por OpenIMSCore. A partir desta instanciação, foi possível estabelecer uma chamada de voz entre dois terminais usando o protocolo SIP. No servidor *Open Source* do Kamailio apenas conseguimos fazer uma chamada SIP sem implementação dos serviços CSCFs. Conseguimos, ainda, analisar o tráfego de pacotes entre as entidades terminais e o servidor, usando o Wireshark.

Concluindo, podemos afirmar que apesar da dificuldade inicial de instalação e configuração de certos elementos que compõem o sistema, que nos atrasou bastante no tempo útil de execução do trabalho, conseguimos cumprir os objectivos iniciais propostos, explorando e implementando algumas das funcionalidades dos diferentes módulos. Podemos assim dizer que os resultados foram positivos mas um pouco aquém dos nossos objectivos iniciais.

6 Anexo

```
1  #!/bin/sh
2  FILE="/etc/resolv.conf"
3  /bin/cat <<EOM >$FILE
4  domain mnc001.mcc001.3gppnetwork.org
5  search mnc001.mcc001.3gppnetwork.org
6  nameserver 192.168.1.200
7  EOM
```

Listing 1: dns.sh