CURSO 2016/17

CALL-PI

PROYECTO FIN DE CICLO ASIR

ÁLVARO GALLARDO, ANDER BUENDÍA, SERGIO PEREIRA
TUTOR DEL PROYECTO: SAMUEL ARRANZ
Departamento de Electrónica e Informática – SALESIANOS ATOCHA

ÍNDICE

1. Introducción

• Objetivo

Este proyecto nació con la idea de poder crear un servidor de llamadas económico y funcional. Teniendo en cuenta que disponemos con un presupuesto bajo, la idea de montar un servidor de llamadas en Raspberry es la mejor opción.

Nuestro proyecto está enfocado a oficinas, hoteles... En definitiva, sitios en los que haga falta tener muchos teléfonos que entre ellos se llamen y puedan además recibir y realizar llamadas al exterior, a través de una operadora virtual de coste reducido. Además las extensiones internas, las podemos utilizar en base a otorgar dispositivos móviles dentro de un entorno empresarial a nuestros empleados y con ello así controlar el tráfico de llamadas de una empresa.

Nuestro objetivo es ser capaz de utilizar cualquier dispositivo de telefonía, ya sean teléfonos IP o móviles, y hacerlos totalmente compatibles entre ellos. También somos capaces de realizar grabaciones de llamadas y también dispone de buzones de voz y a su vez realizar copias de respaldo en un servidor FTP.

• Justificación

Nuestro principal objetivo era la búsqueda de un servicio con documentación escasa del cual facilitar más contenido y poder aplicarlo a una PYMES. Utilizamos herramientas de las que disponemos, como nuestras Raspberrys, un teléfono IP y nuestros Smartphones, aprovechando así los conocimientos adquiridos durante el curso.

• Análisis de lo existente

Hemos utilizado Asterisk, como servidor de llamadas, en su estado más puro, a través de su modo consola que, al trabajar con él, así es mucho más completo y a su vez gestionar de una manera más precisa que las opciones que te da la interfaz gráfica.

A parte de Asterisk, existen más distribuciones como Elastix que es un sistema más gráfico, el cual está preconfigurado y es más sencillo de gestionar incluso a través de otros dispositivos distintos a un ordenador convencional.

Otra distribución es FreePBX, que se gestiona vía web, pero su apariencia gráfica es más clásica.

Por último, esta PBXes, que es directamente una web con las mismas características que las anteriores, pero sin tener directamente instalado el sistema operativo.

• Propuesta detallada

Vamos a utilizar 2 Raspberry Pi 3, ambas tendrán instalado el sistema operativo Debian. Una hará de servidor principal y la otra de almacenamiento de las llamadas realizadas y los backups por si hubiera algún problema con los archivos originales y para recuperarlos fácilmente. Configuraremos los siguientes servicios:

1. Servidor de llamadas (Asterisk). Se encontrará en el servidor principal y nos permitirá realizar llamadas entre extensiones internas, y como troncal, entre llamadas exteriores y extensiones internas. También podremos comprobar el tráfico de llamadas que se realizan y definir las propias extensiones antes definidas. Y utilizaremos el protocolo SIP.

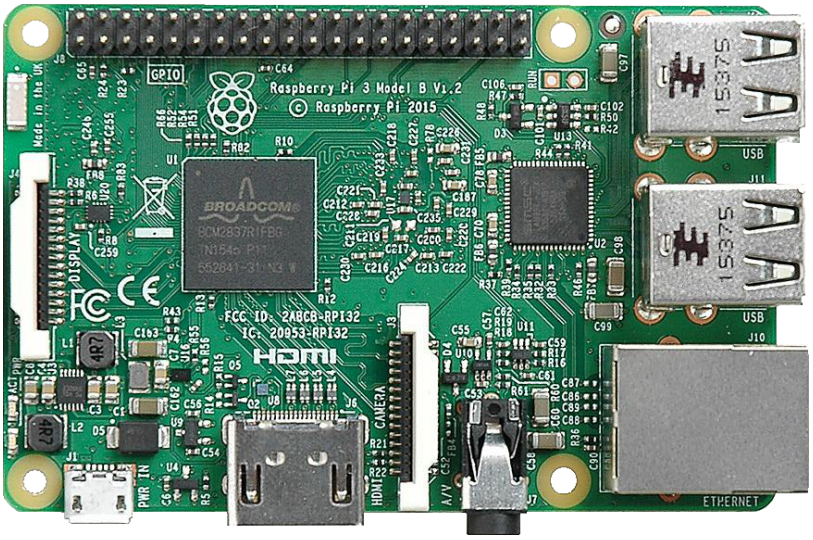
2. Servidor FTP. Para realizar transferencias entre el servidor principal y secundario de las llamadas y buzones de voz.

Elementos utilizados:

- 2 Raspberrys Pi 3
- 2 tarjetas Micro SD 16 GB
- 1 Teléfono VoIP
- 2 Smartphones
- 1 Switch

2. Planificación temporal y evaluación de costes:

- 2 kit Raspberry Pi 3 - 70€ cada una



- Telefono VoIp Yealink T21 - 50€

- Router-Switch - 30€

- App Smartphone - Gratuita

ZoiPer

3. Análisis de requisitos

Este servicio puede ser implementado en cualquier tipo de empresa si lo que quiere ofrecer es un servicio de comunicación laboral con los empleados. Viene a ser que se tendría un dispositivo móvil de empresa, pero la empresa tiene cierto control de lo que haces con ese móvil. Por lo tanto necesitaríamos:

- Datos SIP (Protocolo de Inicio de Sesiones) de una compañía virtual de telefonía móvil. Para este proyecto, hemos escogido una compañía llamada FreedomPop que nos ofrecía este servicio y la posibilidad de realizar llamadas gratis durante una cierta cantidad de minutos.

- Este protocolo SIP al hacer uso de la tecnología VoIP, nos exige del propio servidor de Asterisk en el que tengamos las cuentas de los clientes registradas, por lo que también necesitaremos de una conexión a Internet.

- Para facilitar la comunicación interna de cualquier empresa y una vez configurado el servidor, disponemos de varias aplicaciones en cualquier tienda de los dispositivos móviles que nos permitirá llamar entre los distintos clientes. Estas aplicaciones pueden ser Zoiper o CSIPSimple.

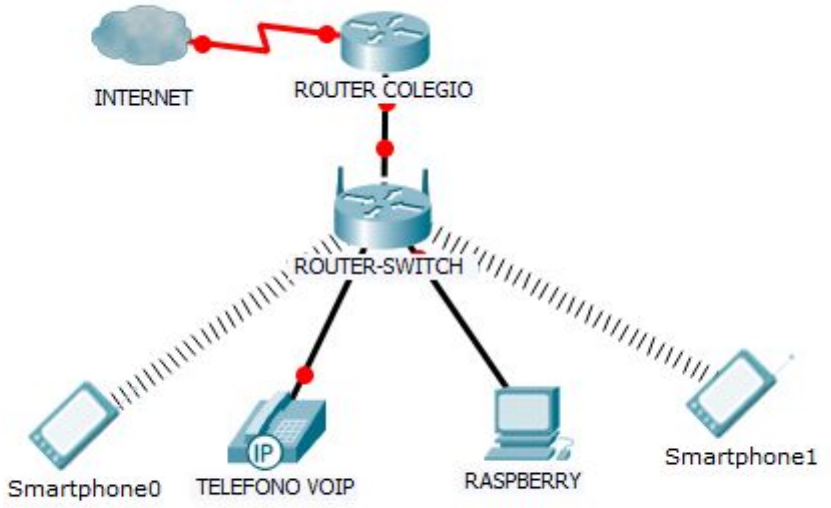
- Necesitamos de un servidor (utilizaremos una Raspberry Pi) en el que instalaremos el servicio Asterisk que usaremos como centralita y otro servidor de almacenamiento de las llamadas.

- Varios teléfonos móviles con la aplicación Zoiper que nos permitirá realizar llamadas con extensiones internas y llamar al exterior, además de un teléfono VoIP que utilizaremos como teléfono principal.

4. Diseño

• Diseño de la red

A través del programa Packet Tracer, se ha diseñado un esquema simple de la organización de los componentes de la red. En el podemos ver, que al Router-Switch se conecta por cable el teléfono VOIP y la Raspberry, y mediante la conexión Wi-Fi los dispositivos móviles. Después ese Router-Switch lo conectaremos al router del colegio, y de ahi saldremos a Internet.



5. Configuración del servidor principal

- ## Preparación de la tarjeta microSD

  Tenemos una tarjeta microSD de 16GB en la que podremos instalar el sistema operativo, que en este caso y con lo aprendido durante el curso, será Raspbian que está basado en Debian (una distro de Linux).

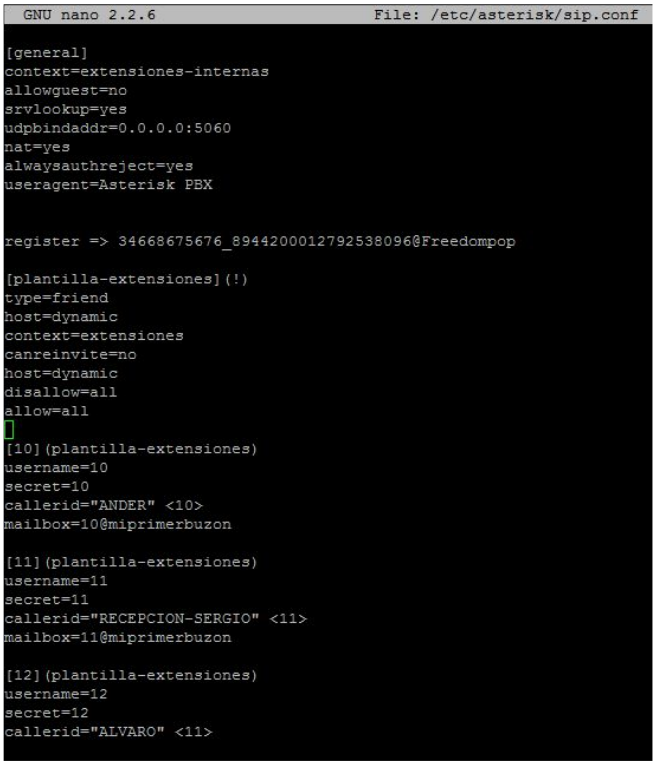
  Para comenzar, descargamos el sistema operativo desde la propia página de Raspbian (https://www.raspberrypi.org/downloads/raspbian).

  Antes de montar la imagen en la microSD, formateamos esta para tener todo el espacio disponible e insertamos la imagen de Raspbian.

  Esto se puede hacer una vez insertada nuestra tarjeta microSD en el equipo y con el programa “SD Formater”, formatear la tarjeta. Una vez formateada, descargamos el programa “Win32DiskImager” y montaremos la imagen dentro de la tarjeta. Y ya podríamos empezar a configurar nuestra Raspberry para introducir en ella el servicio Asterisk.

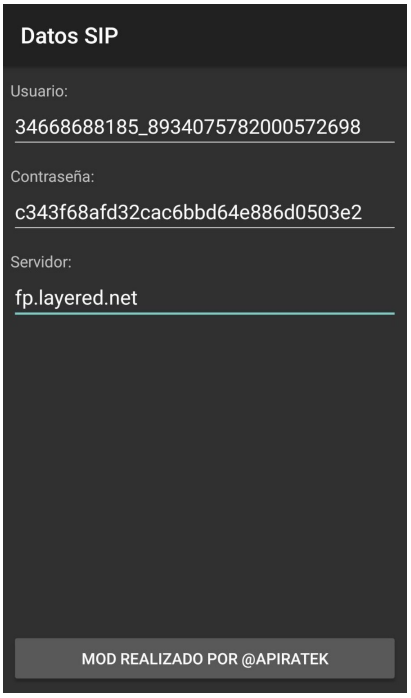
- **Configuración del servicio Asterisk**

  SIP.CONF



```
 GNU nano 2.2.6                          File: /etc/asterisk/sip.conf

[general]
context=extensiones-internas
allowguest=no
srvlookup=yes
udpbindaddr=0.0.0.0:5060
nat=yes
alwaysauthreject=yes
useragent=Asterisk PBX


register => 34668675676_894420001279253809 6@Freedompop

[plantilla-extensiones](!)
type=friend
host=dynamic
context=extensiones
canreinvite=no
host=dynamic
disallow=all
allow=all

[10](plantilla-extensiones)
username=10
secret=10
callerid="ANDER" <10>
mailbox=10@miprimerbuzon

[11](plantilla-extensiones)
username=11
secret=11
callerid="RECEPCION-SERGIO" <11>
mailbox=11@miprimerbuzon

[12](plantilla-extensiones)
username=12
secret=12
callerid="ALVARO" <11>
```

Como podemos ver tenemos configuradas 4 extensiones 2 de ellas con buzones de voz. Aparte también podemos ver que tenemos una línea con el Register que es la que hará de conexión con la compañía virtual y podamos tener acceso desde el servidor.



También configuraremos el troncal para que las llamadas del exterior puedan ser realizadas a nuestra red interna y viceversa. Para sacar los datos del troncal con nuestra compañía Freedompop lo que hemos hecho es descargarnos una aplicación de móvil modeada la cual nos saca los datos de usuario, tanto el nombre como la contraseña.

EXTENSIONS.CONF

```
  GNU nano 2.2.6                File: /etc/asterisk/extensions.conf

[general]
static=yes
writeprotect=yes
autofallthrough=yes
clearglobalvars=no
priorityjumping=no

[default]
; Recibe lo que no tiene un contexto propio definido. Rechaza todo por seguridad.
exten => _X.,1,Hangup(21)
exten => s,1,Hangup(21)

[extensiones]
include = extensiones-internas
include = llamadas-externas
include = emergencias
include = llamadas-no-validas


[extensiones-internas]
; Extensiones internas SIP
#exten => _1X,1,Answer()
#exten => _1X,2,SET(calltime=${STRFTIME(${EPOCH},,%C%y%m%d_%H%M%S)})
#exten => _1X,3,MixMonitor(/home/pi/llamadas/${calltime}-${CALLERID(num)}-${EXTEN}-llamada_interna.wav,b)
#exten => _1X,4,Dial(SIP/${EXTEN})

exten => 10,1,Answer()
exten =>  10,4,Dial(SIP/10,20,Ttm)
exten =>  10,2,SET(calltime=${STRFTIME(${EPOCH},,%C%y%m%d_%H%M%S)})
exten =>  10,3,MixMonitor(/home/pi/llamadas/${calltime}-${CALLERID(num)}-${EXTEN}-llamada_interna.wav,b)
exten => 10,5,Voicemail(10@miprimerbuzon)
same => 10,n,Hangup()

exten => 11,1,Answer()
exten => 11,4,Dial(SIP/11,20,Ttm)
exten => 11,2,SET(calltime=${STRFTIME(${EPOCH},,%C%y%m%d_%H%M%S)})
exten => 11,3,MixMonitor(/home/pi/llamadas/${calltime}-${CALLERID(num)}-${EXTEN}-llamada_interna.wav,b)
exten => 11,5,Voicemail(11@miprimerbuzon)
same => 11,n,Hangup()

exten => 15,1,VoicemailMain(@miprimerbuzon)
```

```
[llamadas-externas]
; moviles que empiezan por 6 y por 71, 72, 73 y 74
exten => _6XXXXXXXX,3,Dial(SIP/34${EXTEN}@Freedompop)
exten => _6XXXXXXXX,1,SET(calltime=${STRFTIME(${EPOCH},,%C%y%m%d_%H%M%S)})
exten => _6XXXXXXXX,2,MixMonitor(/home/pi/llamadas/${calltime}-${EXTEN}-llamada_externa.wav,b)

same => n,Hangup()

exten => _7[1-4]XXXXXXXX,3,Dial(SIP/34${EXTEN}@Freedompop)
exten => _7[1-4]XXXXXXXX,1,SET(calltime=${STRFTIME(${EPOCH},,%C%y%m%d_%H%M%S)})
exten => _7[1-4]XXXXXXXX,2,MixMonitor(/home/pi/llamadas/${calltime}-${EXTEN}.wav,b)
same => n,Hangup()

; gratuitos 900 y 900
exten => _[89]00XXXXXX,3,Dial(SIP/${EXTEN}@Freedompop)
exten => _[89]00XXXXXX,1,SET(calltime=${STRFTIME(${EPOCH},%C%y%m%d_%H%M%S)})
```

En el archivo extensions.conf configuraremos el DIALPLAN que es donde están establecidas todas las reglas para poder llamarse entre unos y otros o también limitar las llamadas o simplemente bloquearlas.

VOICEMAIL.CONF

```
  GNU nano 2.2.6                File: /etc/asterisk/voicemail.conf

[miprimerbuzon]
10 => 4528, Servicio de Habitaciones, shabitaciones@asterisk.com
11 => 7691, Recepcion, recepcion@asterisk.com
```

En el archivo voicemail.conf están configurados los buzones de voz tanto para la extensión 10 como para la 11.

CAMBIAR IDIOMA A ESPAÑOL

En primer lugar creamos la siguiente carpeta "/usr/share/asterisk/sounds/es", una vez la tengamos creada vamos a proceder a descargar los siguientes archivos:

```
sudo wget -O core.zip http://www.asterisksounds.org/es-es/download/asterisk-sounds-core-es-ES-sln16.zip
sudo wget -O extra.zip http://www.asterisksounds.org/es-es/download/asterisk-sounds-extra-es-ES-sln16.zip
```

Una vez los tengamos los descomprimimos con el comando unzip.

- Unzip core.zip
- Unzip extra.zip

```
[general]
context=extensiones-internas
allowguest=no
srvlookup=yes
udpbindaddr=0.0.0.0:5060
nat=yes
alwaysauthreject=yes
useragent=Asterisk PBX
language=es
```

Después nos dirigimos al archivo /etc/asterisk/sip.conf y añadimos la linea language=es y reiniciamos el servidor.

6. Configuración del servidor de almacenamiento

Al igual que hicimos en el servidor principal, seguimos los pasos para preparar la tarjeta microSD. Con este servidor, podremos almacenar y tener una copia de seguridad de las llamadas que se realicen a través del servidor principal (tanto internas como externas).

Para ello hemos creado un script que nos ayudará a realizar esta tarea automáticamente, además de añadirla al administrador de procesos (cron) en el archivo crontab para que esta tarea se ejecute cada día a una determinada hora (@daily).

- ## Script: backup de las llamadas

Este primer script que hemos realizado en el servidor principal, realizará la copia de seguridad de todas las llamadas que se hayan hecho en un día y las guardará y comprimirá en un archivo gzip por fecha. A continuación, con el protocolo FTP, se enviará el archivo comprimido hacia una carpeta específica en el servidor de almacenamiento. Como hemos dicho anteriormente, este script lo añadiremos en el archivo crontab para que se ejecute una vez al día y así tener el proceso de copias de seguridad automatizado.

```
#!/bin/bash
USER=backupllamadas
FECHA=`date +%Y-%m-%d`
fecha2=`date +%Y%m%d`
if `ls /home/backups/ | grep $FECHA > /dev/null`
then
echo "Ya esta creado el archivo.";
else
mkdir /home/backups/backup-$FECHA
fi

echo "Haciendo el Backup de las llamadas."
ls /home/pi/llamadas/ | grep $fecha2 > copias.txt
while read copia
do
sudo cp /home/pi/llamadas/$copia /home/backups/backup-$FECHA/
done < copias.txt
sudo rm copias.txt

echo "Comprimiendo el backup."
/bin/tar cvfz /home/backups/backup-$FECHA.tar.gz /home/backups/backup-$FECHA/

echo "Transferencia completada"
echo "Enviando el backup al servidor de FTP"
sh /home/backups/enviar.sh
```

- **Script: transferencia de archivos con el protocolo FTP**

  Con este segundo script, transferiremos los archivos desde el servidor principal a una carpeta específica (/home/backups) en el servidor de almacenamiento. Introducimos en la IP del servidor de almacenamiento y los datos de un usuario que hayamos creado junto con el comando ftp junto a la opción -n (esta opción servirá para restringir el auto-login ya que hemos añadido dentro del script los datos de un usuario). Una vez se ejecuten los dos scripts, los archivos de sonido registrados al hacer llamadas, se comprimirán y transferirán al servidor de almacenamiento para poder ser reproducidos en cualquier momento.

```
#!/bin/bash

echo "Conectando y autenticando con el servidor de FTP"
FECHA=`date +%Y-%m-%d`
HOST=172.16.112.53
USER=backupllamadas
PASSWD=1234
cd /home/backups/
echo "Enviando back-up de las llamadas"
ftp -n $HOST << END_SCRIPT
quote USER $USER
quote PASS $PASSWD
binary
put backup-$FECHA.tar.gz
quit
END_SCRIPT
echo "Archivo enviado correctamente"
```

```
pi@raspberrypi:/home/backups $ ls
backup-2017-02-15  backup-2017-02-15.tar.gz  enviar.sh  home
```

Podemos ver que se crea el fichero comprimido.

```
pi@raspberrypi:/home/backups/backup-2017-02-15 $ ls
20170215_130458-11-10-llamada_interna.wav  20170215_133354-11-10-llamada_interna.wav      20170215_140928-11.wav
20170215_132623-11-10-llamada_interna.wav  20170215_133358-11-11-llamada_interna.wav      20170215_141006-llamada_entrante-+34697719668.wav
20170215_133336-11-10-llamada_interna.wav  20170215_140417-11.wav
20170215_133345-11-10-llamada_interna.wav  20170215_140541-697719668-llamada_externa.wav
```

Y al extraerlo, podemos comprobar que tenemos en él una copia de todas las llamadas grabadas hechas hasta la fecha del backup.

7. Codificación

• Entorno de programación

El entorno de programación es en linux bajo la distro de Raspbian que está basada en Debian

• Lenguajes y herramientas

El lenguaje de programación que se utilizamos es Shell, el lenguaje de programación de Linux, aunque ASTERISK utilice sus propios términos y atributos.

• Aspectos relevantes de la implementación

USERAGENT

```
[general]
context=extensiones-internas
allowguest=no
srvlookup=yes
udpbindaddr=0.0.0.0:5060
nat=yes
alwaysauthreject=yes
useragent=Asterisk PBX
```

Estableceremos en el archivo sip.conf la línea que está dentro del recuadro para evitar dar información sobre la versión de nuestro servidor para que cualquier vulnerabilidad que haya les sea más complicado acceder a ella.

USUARIOS ANÓNIMOS

```
[general]
context=extensiones-internas
allowguest=no
srvlookup=yes
udpbindaddr=0.0.0.0:5060
nat=yes
alwaysauthreject=yes
useragent=Asterisk PBX
```

```
[extensiones-internas]
; Extensiones internas SIP
#exten => _1X,1,Answer()
#exten => _1X,2,SET(calltime=${STRFTIME(${EPOCH},,%C%y%m%d_%H%M%S)})
#exten => _1X,3,MixMonitor(/home/pi/llamadas/${calltime}-${CALLERID(num)}-${EXTEN}-llamada_interna.wav,b)
#exten => _1X,4,Dial(SIP/${EXTEN})

exten => 10,1,Answer()
exten =>  10,4,Dial(SIP/10,20,Ttm)
exten =>  10,2,SET(calltime=${STRFTIME(${EPOCH},,%C%y%m%d_%H%M%S)})
exten =>  10,3,MixMonitor(/home/pi/llamadas/${calltime}-${CALLERID(num)}-${EXTEN}-llamada_interna.wav,b)
exten => 10,5,Voicemail(10@miprimerbuzon)
same => 10,n,Hangup()

exten => 11,1,Answer()
exten => 11,4,Dial(SIP/11,20,Ttm)
exten => 11,2,SET(calltime=${STRFTIME(${EPOCH},,%C%y%m%d_%H%M%S)})
exten => 11,3,MixMonitor(/home/pi/llamadas/${calltime}-${CALLERID(num)}-${EXTEN}-llamada_interna.wav,b)
exten => 11,5,Voicemail(11@miprimerbuzon)
same => 11,n,Hangup()

exten => 15,1,VoicemailMain(@miprimerbuzon)
```

Con el *context* lo que haremos es que no se puedan loguear ningun usuarios anónimo, por lo tanto estamos limitando el acceso a los clientes.

USUARIOS INVITADOS

```
[general]
context=extensiones-internas
allowguest=no
srvlookup=yes
udpbindaddr=0.0.0.0:5060
nat=yes
alwaysauthreject=yes
useragent=Asterisk PBX
```

Otra vez en el archivo sip.conf estableceremos la regla de allowguest=no para que no puedan realizar llamadas los usuarios que no estén logueados.

BLOQUEAR INFORMACIÓN DE EXTENSIONES

```
[general]
context=extensiones-internas
allowguest=no
srvlookup=yes
udpbindaddr=0.0.0.0:5060
nat=yes
alwaysauthreject=yes
useragent=Asterisk PBX
```

Con el parámetro de alwaysauthreject=yes lo que haremos es que el servidor no dé información sobre las extensiones para que así el atacante no tenga información sobre el rango de extensiones, de esta manera siempre devolverá el mismo tipo de información.

FAIL2BAN

```
  GNU nano 2.2.6                File: /etc/asterisk/logger.conf

;
; Logging Configuration
;
; In this file, you configure logging to files or to
; the syslog system.
;
; "logger reload" at the CLI will reload configuration
; of the logging system.

[general]
dateformat=%F %T        ; ISO 8601 date format
```

```
  GNU nano 2.2.6                          File: /etc/asterisk/logger.conf

; Directory for log files is configures in asterisk.conf
; option astlogdir
;
[logfiles]
security => security
```

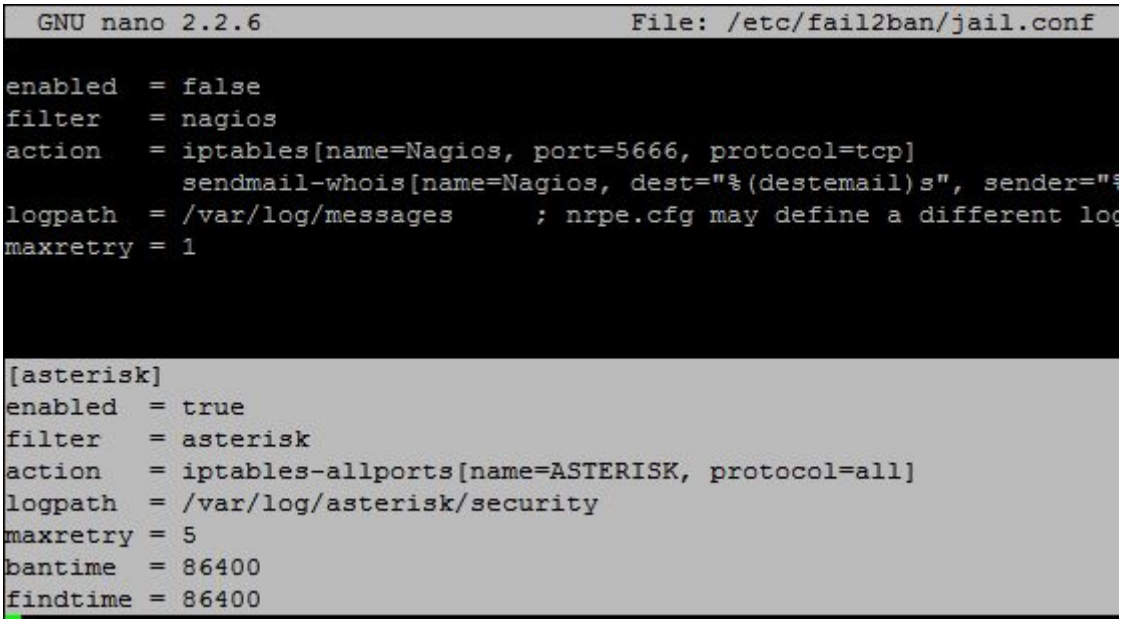
En el archivo /etc/asterisk/logger debemos dejar según esta el dateformat y tambien la linea de security para que el Fail2Ban interprete bien los Logs.

```
pi@raspberrypi:~ $ sudo apt-get install fail2ban
```

Después procedemos a instalar el Fail2ban con el comando de apt-get install fail2ban

```
  GNU nano 2.2.6                File: /etc/fail2ban/filter.d/asterisk.conf

# Fail2Ban configuration file
#
#
# $Revision: 250 $
#

[INCLUDES]

# Read common prefixes. If any customizations available -- read them from
# common.local
#before = common.conf

[Definition]

#_daemon = asterisk

# Option:  failregex
# Notes.:  regex to match the password failures messages in the logfile. The
#          host must be matched by a group named "host". The tag "<HOST>" can
#          be used for standard IP/hostname matching and is only an alias for
#          (?:::f{4,6}:)?(?P<host>\S+)
# Values:  TEXT
#

failregex = SECURITY.* SecurityEvent="FailedACL".*RemoteAddress=".+?/.+?/<HOST>/.+?".*
            SECURITY.* SecurityEvent="InvalidAccountID".*RemoteAddress=".+?/.+?/<HOST>/.+?".*
            SECURITY.* SecurityEvent="ChallengeResponseFailed".*RemoteAddress=".+?/.+?/<HOST>/.+?".*
            SECURITY.* SecurityEvent="InvalidPassword".*RemoteAddress=".+?/.+?/<HOST>/.+?".*

# Option:  ignoreregex
# Notes.:  regex to ignore. If this regex matches, the line is ignored.
# Values:  TEXT
#
ignoreregex =
```

Una vez instalado tendremos que añadir un filtro específico para los logs de seguridad de asterisk, escribimos todo según aparece en la imagen de arriba



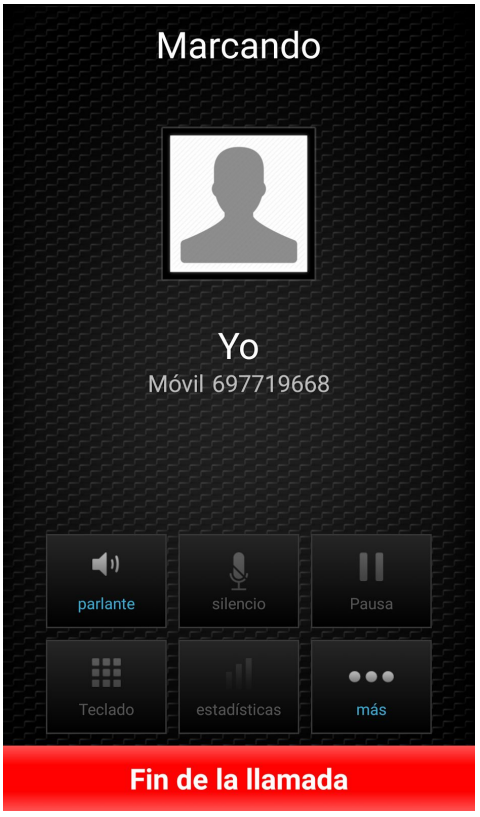
Por último añadimos las reglas de seguridad y esta lo que hará en concreto es que solamente dejará 5 intentos y si no será baneado durante 24 horas

8. Pruebas de ejecución

- **Pruebas funcionales**

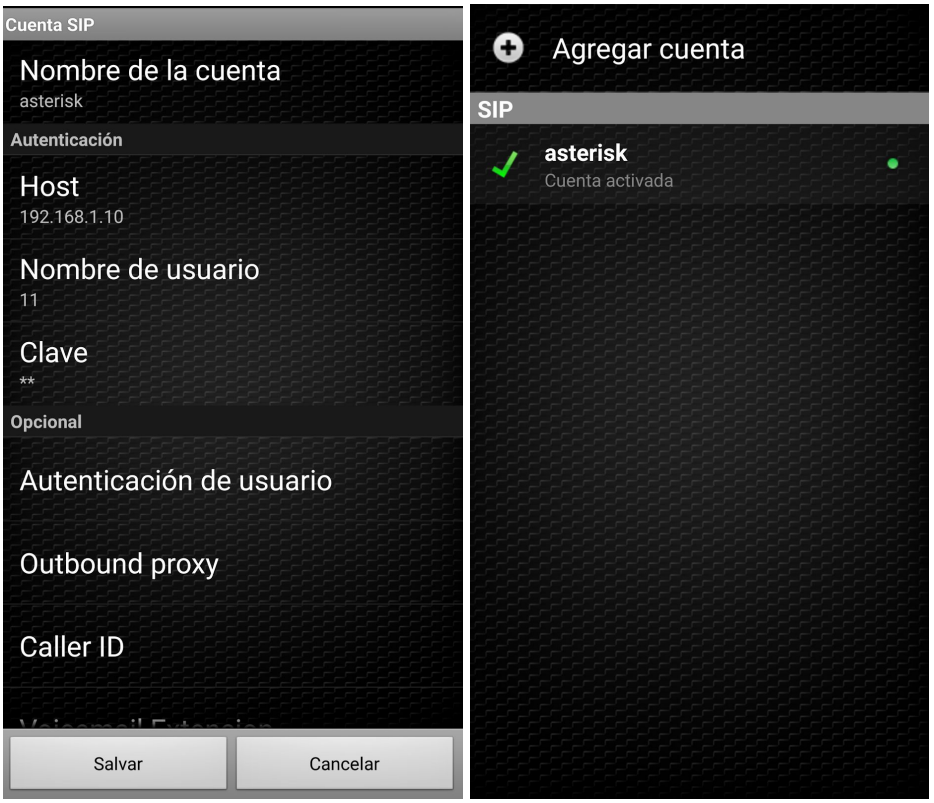
Podemos ver una prueba de cómo podemos llamarnos entre extensiones incluso a uno mismo, en el sip.conf según lo que tengas configurado en cada extensión te mostrará un nombre u otro.

- **Pruebas de usabilidad**



Para las llamadas al exterior dentro del territorio nacional simplemente lo que haremos será marcar el número sin la extensión geográfica ya que viene por defecto para los teléfonos y móviles españoles, en nuestro caso solo podrá llamar al exterior la extensión 11.

- **Pruebas de accesibilidad**



Para acceder al servidor como cliente es tan fácil como loguearnos en la aplicación conociendo la ip del servidor, el usuario y contraseña no se necesita nada más.

- **Pruebas de seguridad**

```
pi@raspberrypi:~ $ sudo fail2ban-client status asterisk
Status for the jail: asterisk
|- filter
|  |- File list:        /var/log/asterisk/security
|  |- Currently failed: 0
|  `- Total failed:     0
`- action
   |- Currently banned: 0
   |  `- IP list:
   `- Total banned:     0
```

Si ejecutamos este comando lo que veremos es si ha habido errores a la hora de iniciar sesión con las extensiones y si algunos ha sido baneado.

• Pruebas de carga

```
pi@raspberrypi:~ $ sudo asterisk -rvvvvvv
Asterisk 11.13.1~dfsg-2+deb8u2, Copyright (C) 1999 - 2013 Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 11.13.1~dfsg-2+deb8u2 currently running on raspberrypi (pid = 2449)
raspberrypi*CLI> sip show peers
Name/username             Host                                    Dyn Forcerport Comedia    ACL Port     Status      Des
cription
10/10                     (Unspecified)                            D  Yes        Yes            0        Unmonitored

11/11                     172.16.110.92                            D  Yes        Yes            5062     Unmonitored

12/12                     172.16.110.74                            D  Yes        Yes            52296    Unmonitored

Freedompop/34668675676_89 169.55.60.25                                Yes        Yes            5060     Unmonitored

4 sip peers [Monitored: 0 online, 0 offline Unmonitored: 3 online, 1 offline]
raspberrypi*CLI> sip show peers
Name/username             Host                                    Dyn Forcerport Comedia    ACL Port     Status      Description
10/10                     (Unspecified)                            D  Yes        Yes            0        Unmonitored
11/11                     172.16.110.92                            D  Yes        Yes            5062     Unmonitored
12/12                     172.16.110.74                            D  Yes        Yes            52296    Unmonitored
Freedompop/34668675676_89 169.55.60.25                                Yes        Yes            5060     Unmonitored
4 sip peers [Monitored: 0 online, 0 offline Unmonitored: 3 online, 1 offline]
```

```
    -- Executing [11@extensiones:1] Answer("SIP/11-0000000a", "") in new stack
       > 0x55f599e8 -- Probation passed - setting RTP source address to 192.168.1.4:42754
[2017-03-11 16:54:24] NOTICE[4220][C-00000006]: res_rtp_asterisk.c:4364 ast_rtp_read: Unknown RTP codec 95 received from '192.168.1.4:42754
'
    -- Executing [11@extensiones:2] Set("SIP/11-0000000a", "calltime=20170311_165424") in new stack
    -- Executing [11@extensiones:3] MixMonitor("SIP/11-0000000a", "/home/pi/llamadas/20170311_165424-11-11-llamada_interna.wav,b") in new s
tack
    -- Executing [11@extensiones:4] Dial("SIP/11-0000000a", "SIP/11,20,Ttm") in new stack
  == Begin MixMonitor Recording SIP/11-0000000a
  == Using SIP RTP CoS mark 5
    -- Called SIP/11
    -- Started music on hold, class 'default', on SIP/11-0000000a
[2017-03-11 16:54:24] NOTICE[4220][C-00000006]: channel.c:4301 __ast_read: Dropping incompatible voice frame on SIP/11-0000000a of format u
law since our native format has changed to (gsm)
    -- SIP/11-0000000b is ringing
```

Como se puede comprobar en la primera imagen, vemos los usuarios que están conectados y también la ip que utiliza el troncal.

En la imagen de abajo podemos ver que se está realizando la llamada y también se activa el MixMonitor que es una herramienta de Asterisk que se utiliza para grabar las conversaciones.

9. Conclusiones

• Conclusiones sobre el trabajo realizado

Como resultado de los datos obtenidos en la realización de este proyecto, se concluye que no se necesita ni de muchos recursos económicos ni de una estación de trabajo la cual ocupe una notable cantidad de espacio para tener un servidor centralizado por el que discurra toda una pequeña red telefónica y poderla incluir en una empresa y que esté funcionando, utilizando una cantidad de tiempo mínima.

• Conclusiones sobre el sistema desarrollado

Con Asterisk como servidor de llamadas, hemos visto la potencia que puede tener el servicio, siendo su interfaz en modo terminal sin ser gráfica e intuitiva.

• Conclusiones personales

Al realizar este proyecto, hemos aplicado conocimientos más allá de lo ya aprendido durante el curso en relación a este. Además es un proyecto que no solo se puede aplicar en lo personal si no que puede ser exportado a una faceta comercial.

• Posibles ampliaciones y mejoras

Una de las ampliaciones que tuvimos en cuenta pero no pudimos realizar fue la implementación de un dispositivo que convierte la señal analogica de un telefono a digital para que funcionará en nuestro servidor, pero el que disponíamos no tenía compatibilidad con la arquitectura de nuestra raspberry, pero sería algo bastante importante ya que podríamos darle a uso al típico teléfono de casa.

Tambien se podria mejorar los scripts de tal manera que con el tiempo el espacio se agotará y tendríamos que ir borrando las llamadas de una manera ordenada.

Otras de las mejoras sería que la calidad del sonido de las llamadas fuera mejor gracias a otro tipo de codecs.

10. Bibliografía

La información recopilada para la realización de este proyecto, ha sido recogida a través de internet y en su defecto, de distintas páginas web. Como no hay gran cantidad de información, cada cosa que se encontraba en la siguientes páginas era de gran ayuda.

- Página web donde comprar la Raspberry Pi:

  https://www.amazon.es/Raspberry-Pi-Modelo-Quad-core-Cortex-A53/dp/B01CD5VC92/ref=sr_1_1?ie=UTF8&qid=1489251929&sr=8-1&keywords=raspberrypi

- Página web donde descargar Zoiper (aplicación móvil):

  https://play.google.com/store/apps/details?id=com.zoiper.android.app&hl=es_419

- Diferentes foros de ayuda en la instalación y configuración de Asterisk:

  www.axelko.com

  www.asteriskguru.com

  www.asteriskmx.org/foros

  www.youtube.com

  www.forocoches.com