PRÁCTICA 1 - SERVIZOS MULTIMEDIA

UNIVERSIDADE DA CORUÑA

Estudante: Óscar Olveira Miniño

Correo: oscar.olveira@udc.es

Ejercicio 1: Configurar los clientes sip1 y sip2:

1. ¿Cual es el estado de los endpoints sip1 y sip2?

Estado inicial dos endpoints en ‘Unavailable’ porque non temos ningún usuario conectado:

```
root@freepbx:~

freepbx*CLI> pjsip show endpoints

 Endpoint:  <Endpoint/CID.................................>  <State.....>  <Channels.>
    I/OAuth:  <AuthId/UserName...........................................................>
        Aor:  <Aor............................................>  <MaxContact>
      Contact:  <Aor/ContactUri..........................> <Hash....> <Status> <RTT(ms)..>
  Transport:  <TransportId........>  <Type>  <cos>  <tos>  <BindAddress..................>
   Identify:  <Identify/Endpoint.........................................................>
        Match:  <criteria.........................>
    Channel:  <ChannelId......................................>  <State.....>  <Time.....>
        Exten: <DialedExten...........>  CLCID: <ConnectedLineCID.......>
==========================================================================================

 Endpoint:  sip1                                                Unavailable   0 of inf
     InAuth:  authsip1/sip1
        Aor:  sip1                                              1

 Endpoint:  sip2                                                Unavailable   0 of inf
     InAuth:  authsip2/sip2
        Aor:  sip2                                              1


Objects found: 2
```

Estado dos endpoints en ‘Not in use’, unha vez que os usuarios se conectaron:

```
root@freepbx:~

freepbx*CLI> pjsip show endpoints

 Endpoint:  <Endpoint/CID.................................>  <State.....>  <Channels.>
    I/OAuth:  <AuthId/UserName...........................................................>
        Aor:  <Aor............................................>  <MaxContact>
      Contact:  <Aor/ContactUri..........................> <Hash....> <Status> <RTT(ms)..>
  Transport:  <TransportId........>  <Type>  <cos>  <tos>  <BindAddress..................>
   Identify:  <Identify/Endpoint.........................................................>
        Match:  <criteria.........................>
    Channel:  <ChannelId......................................>  <State.....>  <Time.....>
        Exten: <DialedExten...........>  CLCID: <ConnectedLineCID.......>
==========================================================================================

 Endpoint:  sip1                                                Not in use    0 of inf
     InAuth:  authsip1/sip1
        Aor:  sip1                                              1
      Contact:  sip1/sip:sip1@192.168.1.51:40675;transport d4489a92da NonQual         nan

 Endpoint:  sip2                                                Not in use    0 of inf
     InAuth:  authsip2/sip2
        Aor:  sip2                                              1
      Contact:  sip2/sip:sip2@192.168.1.59;transport=udp   d35381f5f3 NonQual         nan


Objects found: 2
```

2. ¿En que direcciones IP y puertos se puede contactar con estos endpoints?

Como podemos ver nas imaxes anteriores, unha vez que os usuarios se conectaron, o endpoint do usuario sip1 ten a ip 192.168.1.51 co puerto 40675 e o endpoint do usuario sip2 ten a ip 192.168.1.59 sin puerto, polo que enténdese que é o puerto por defecto (5060)

3. ¿Cuantos contactos podrían registrarse como máximo para cada uno de los endpoints? ¿Cómo lo sabes?

Tal como está o archivo de configuración **pjsip_custom.conf,** na sección con tipo aor, en ambos endpoints está o parámetro 'max_contacts=1', polo que solamente se pode rexistrar 1 contacto por cada endpoint

Ejercicio 2: Configurar los buzones de voz para sip1 y sip2:

Configuracion final do archivo voicemail.conf:

```
[general]
format=wav49|wav
serveremail=voicemail@udc.test.es
attach=yes
maxsilence=10
maxlogins=3

[zonemessages]
european=Europe/Copenhagen|'vm-received' a d b 'digits/at' HM

[sm]
001 => 1234,Oscar,oscar.olveira@udc.test.es,,tz=central
002 => 0000,Pepe,pepe@udc.test.es,,tz=european|maxmsg=5
```

Salida obtida unha vez configurado o archivo:

```
freepbx*CLI> voicemail show users
Context    Mbox  User                     Zone       NewMsg
sm         001   Oscar                    central         0
sm         002   Pepe                     european        0
2 voicemail users configured.
freepbx*CLI>
```

Ejercicio 3: Implementar la subrutina sub_recordcallinfo:

Código da subrutina sub_recordcallinfo completado:

```
[sub_recordcallinfo]
; Record call information using AstDB
; ARG1: User
; ARG2: Dialstatus
exten => start,1,Verbose(3,Recording call info: User ${ARG1},Status: ${ARG2})
same => n,ExecIf($[${ARG2}=ANSWER]?Set(LOCAL(key)=a))
same => n,ExecIf($[${ARG2}=BUSY]?Set(LOCAL(key)=b))
same => n,ExecIf($[${ARG2}=CHANUNAVAIL]?Set(LOCAL(key)=u))
same => n,ExecIf($[${ARG2}=NOANSWER]?Set(LOCAL(key)=n))
same => n,ExecIf($[${ISNULL(${LOCAL(key)})}=1]?Set(LOCAL(key)=o))
same => n,Set(LOCAL(familykey)=calls_${ARG1}/${LOCAL(key)})
same => n,Set(LOCAL(count)=${DB(${LOCAL(familykey)})})
same => n,ExecIf($[${ISNULL(${LOCAL(count)})}=1]?Set(LOCAL(count)=0))
same => n,Set(DB(${familykey})=$[${LOCAL(count)}+1])
same => n,Return()
```

Ejercicio 4: Implementar la subrutina sub_sipdial:

Código da subrutina sub_sipdial completado:

```
[sub_sipdial]
; Dial a user and save call information
; Parameters:
; ARG1 Endpoint (sip1 or sip2)
; ARG2 Timeout
; ARG3 Options

exten => start,1,Verbose(1,Dialing SIP user ${ARG1})
same => n,Dial(pjsip/${ARG1},{ARG2},{ARG3}
same => n(record),GoSub(sub_recordcallinfo,start,1(${ARG1},${DIALSTATUS}))
same => n,Return()

exten => h,1,Goto(start,record)
```

EJERCICIO 5: Implementar las extensiones 01 y 02:

Código completado das extensións 01 e 02:

```
exten => 01,1,Verbose(1,Dial user sip1)
same => n,GoSub(sub_sipdial,start,1(sip1,,))
same => n,HangUp()

exten => 02,1,Verbose(1,Dial user sip2)
same => n,GoSub(sub_sipdial,start,1(sip2,,))
same => n,HangUp()
```

Como se pode ver na imaxe, rexistrarse perfectamente as chamadas:

```
[root@freepbx asterisk]# asterisk -rvvv
Asterisk 18.13.0, Copyright (C) 1999 - 2021, Sangoma Technologies Corporation 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 18.13.0 currently running on freepbx (pid = 1756)
freepbx*CLI> database show calls_sip1
/calls_sip1/a                                     : 1
/calls_sip1/b                                     : 2
/calls_sip1/o                                     : 3
/calls_sip1/u                                     : 4
4 results found.
freepbx*CLI> database show calls_sip2
/calls_sip2/b                                     : 1
/calls_sip2/o                                     : 3
/calls_sip2/u                                     : 4
3 results found.
freepbx*CLI>
```

EJERCICIO 6: Implementar la subrutina sub_voicemail:

Código completado da subrutina sub_voicemail:

```
[sub_voicemail]
; Call voicemail after a Dial
; Parameters
; ARG1 mailbox@context
; ARG2 DIALSTATUS
; ARG3 Options to voicemail

exten => start,1,Verbose(1,Executing sub_voicemail for ${ARG1})
same => n,GotoIf($[${ISNULL(${ARG2})}|${ARG2}=ANSWER]?exit)
same => n,Set(LOCAL(message)=${IF($[${ARG2}=BUSY]?b:u)})
same => n,VoiceMail(${ARG1},${message}${ARG3})
same => n(exit),Return()
```

Ejercicio 7: Implementar la extensión 00X:

Código da extensión 00X completado:

```
exten => _00[1,2],1,Verbose(1,Say extension and dial user)
same => n,Playback(you-have-dialed)
same => n,SayDigits(${EXTEN})
same => n,Set(user=sip${EXTEN:2:1})
same => n,GoSub(sub_sipdial,start,1(${user},10))
same => n,GoSub(sub_voicemail,start,1(${MAILBOX_${user}},${DIALSTATUS}))
same => n,HangUp()
```

Como se pode ver na imaxe, rexístranse os mensaxes no buzón de voz:

```
[root@freepbx asterisk]# asterisk -rvvv
Asterisk 18.13.0, Copyright (C) 1999 - 2021, Sangoma Technologies Corporation 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 18.13.0 currently running on freepbx (pid = 1756)
freepbx*CLI> voicemail show users
Context    Mbox  User                      Zone       NewMsg
sm         001   Oscar                     central         3
sm         002   Pepe                      european        1
2 voicemail users configured.
freepbx*CLI>
```

Ejercicio 8: Implementar la extensión 120:

Código da extensión 120 completado:

```
exten => 120,1,Verbose(1,Check voicemail messages)
same => n,Set(user=${CHANNEL(endpoint)})
same => n,Set(mailbox=${MAILBOX_${user}})
same => n,GotoIf(${ISNULL(${mailbox})}?end:)
same => n,VoiceMailMain(${mailbox})
same => n,HangUp()
same => n(end),Verbose(1,Mailbox for user ${user} not defined)
same => n,Playback(feature-not-avail-line)
same => n,HangUp()
```

Si tecleamos a extensión vainos pedir a contrasinal que se asignou ao usuario no archivo de configuración **voicemail.conf**. Unha vez a introducimos, podemos escoitar os mensaxes pulsando a tecla 1 e si temos varios mensaxes, escoitamos os seguintes pulsando a tecla 6 para cada un.

Ejercicio 9: Implementar la extensión 99X:

Código da extensión 99X completado:

```
exten => _99[1-2],1,Verbose(1,Reject call in given times)
same => n,Verbose(1,System time is: ${STRFTIME()})
same => n,Verbose(1,Local time is: ${STRFTIME(,${TIMEZONE})})
same => n,GoToIfTime(15:30-20:00,mon-thu,*,sep-jul?accept)
same => n,Playback(unavailable)
same => n,Playback(please-try-again-later)
same => n,Goto(exit)
same => n(accept),GoSub(sub_sipdial,start,1(sip${EXTEN:2:1},,))
same => n(exit),HangUp()
```

Ejercicio 10: Implementar la subrutina sub_getcallscount:

Código da subrutina sub_getcallscount completado:

```
[sub_getcallscount]
; Gets the count of calls using AstDB
; ARG1: Endpoint (sip1 or sip2)
; ARG2: Type: a(answer), b(BUSY), u(CHANUNAVAIL), n(NOANSWER), o(OTHER)
exten => start,1,Verbose(3,Getting call numer for: User ${ARG1}, type: ${ARG2})
same => n,Set(LOCAL(familykey)=calls_${ARG1}/${ARG2})
same => n,Set(LOCAL(count)=${DB(${LOCAL(familykey)})})
same => n,ExecIf($[${ISNULL(${LOCAL(count)})} = 1]?Set(LOCAL(count)=0))
same => n,Return(${LOCAL(count)})
```

Ejercicio 11: Implementar la extensión #0:

Primeiro programamos a subrutina para o menú:

```
[menu1]
exten => start,1,Background(press)
same => n,Background(digits/0)
same => n,Background(or-press)
same => n,Background(digits/1)
same => n,Background(or-press)
same => n,Background(digits/2)
same => n,Background(or-press)
same => n,Background(digits/3)
same => n,WaitExten(10)

exten => 0,1,Set(LOCAL(key)=u)
same => n,Set(LOCAL(play)=unavaliable)
same => n,GoTo(2,avisar)

exten => 1,1,Set(LOCAL(key)=b)
same => n,Set(LOCAL(play)=on-busy)
same => n,GoTo(2,avisar)

exten => 2,1,Set(LOCAL(key)=a)
same => n,Set(LOCAL(play)=route-sip)

same => n(avisar),GoSub(sub_getcallscount,start,1(${CHANNEL(endpoint)},${LOCAL(key)}))
same => n,SayNumber(${GOSUB_RETVAL})
same => n,Playback(${LOCAL(play)})
same => n,GoTo(start,1)

exten => 3,1,Playback(vm-goodbye)
same => n,HangUp()

exten => i,1,Playback(pbx-invalid)
same => n,GoTo(menu1,start,1)

exten => t,1,Playback(vm-goodbye)
same => n,HangUp()
```

Unha vez temos a subrutina, creamos a extensión #0:

```
exten => #0,1,Verbose("Menu con extension #0")
same => n,GoSub(menu1,start,1())
```