VoIP Workshop

APRICOT 2007

Bali, Indonesia

21 February - 25 February 2007

Labs 1 - 4

Lab Summary

Root password for each box: voipWSbali

User login is: workshop, passwd voipUser

Server primary IP and default route / DNS servers, as DHCP'd by the conference network.

15 Groups:

Group	Extensions	Server Secondary IP
1	2100 - 2199	10.70.1.1/24
2	2200 - 2299	10.70.2.1/24
3	2300 - 2399	10.70.3.1/24
4	2400 - 2499	10.70.4.1/24
5	2500 - 2599	10.70.5.1/24
6	2600 - 2699	10.70.6.1/24
7	2700 - 2799	10.70.7.1/24
8	2800 - 2899	10.70.8.1/24
9	2900 - 2999	10.70.9.1/24
10	3000 - 3099	10.70.10.1/24
11	3100 - 3199	10.70.11.1/24
12	3200 - 3299	10.70.12.1/24
13	3300 - 3399	10.70.13.1/24
14	3400 - 3499	10.70.14.1/24
15	3500 - 3599	10.70.15.1/24

Dial Plan for each group:

‘xy’ is as per your group number extensions above.

xy00 - xy09	SIP phones (password for each, *extension*passwd, e.g. 2000passwd)
xy10 - xy19	Fun with IVRs
xy20 - xy29	Music on hold extensions
xy30	DB count application
xy40 - xy9	Agents
xy50	'Helpdesk' queue access
xy59	Agent Login / Logout
xy60	Echo test
xy70	Conference bridge
xy80 - xy89	Festival test to speech play extensions
xy90 - xy98	Other Lab exercises
xy99	Voicemail access

1.	Access an outside PSTN line
8.	Access another group
9.	Access INOC-DBA

Lab 1: Initial Asterisk Install

0. Host Setup

login as root

Set hostname to voip[group number], e.g. voip01, voip02...

```
hostname voipxx

nano /etc/hostname
```

Edit /etc/hostname to reflect the hostname for your group.

1. Install Asterisk

```
apt-get install asterisk
apt-get install asterisk-sounds-extra
apt-get install zaptel
apt-get install zaptel-source
apt-get build-dep asterisk


cd /usr/src
bunzip2 zaptel.tar.bz2
tar -xvf zaptel.tar
cd /usr/src/modules/zaptel
```

Remove original source file so that the module assistant (m-a) builds properly:

```
mv zaptel.tar zaptel.tar.old

m-a prepare
m-a build zaptel
dpkg -i zaptel-modules-xxxxxx.deb

depmod
modprobe zaptel
modprobe ztdummy

ztcfg -v  # only if you have real zaptel interfaces
```

Set RUNASTERISK=yes in /etc/default/asterisk

```
nano /etc/default/asterisk
```

2. Start Asterisk

```
/etc/init.d/asterisk start
```

Have a look at the available startup options:

```
asterisk -h
```

To connect to the Asterisk CLI:

```
asterisk -r
```

3. Edit Configuration Files in /etc/asterisk

Set up three SIP peers in sip.conf: xy00, xy01, xy02. Add to the bottom of sip.conf, repeating for each of the three SIP peers:

```
[xy00]
type=friend
host=dynamic
username=xy00
secret=passwdxy00
canreinvite=no
nat=yes
context=phones
dtmfmode=rfc2833
allow=all
```

Create backup of original extensions.conf:

```
mv extensions.conf orig_extensions.conf
```

Create new extensions.conf with the following:

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

[phones]
; remember to replace xy with your group's numbers!
exten => xy00,1,Dial(SIP/xy00)
exten => xy01,1,Dial(SIP/xy01)
exten => xy02,1,Dial(SIP/xy02)

exten => xy60,1,Answer()
exten => xy60,2,Playback(demo-echotest)
exten => xy60,3,Echo
exten => xy60,4,Playback(demo-echodone)
exten => xy60,5,Hangup
```

Connect to Asterisk (asterisk -r), up the debug output (set verbose 10), and reload the config (reload)

4. Configure Softphone

Download and configure the Xten Xlite Softphone - (http://www.xten.com/index.php?menu=download)

Input SIP settings in *Main Menu > System Settings > SIP Proxy > Default*

Enabled:	Yes
Username:	SIP extension you are configuring (e.g. 2000)
Authorization User:	Same as *Username*
Password:	extensionpasswd, e.g. 2000passwd
SIP Proxy:	The address of your Asterisk server
OutBound Proxy:	Same as *SIP Proxy*

You should now be able to call between your three phones.

Call the echo test on xy60 and you should be able to hear yourself!

Lab 2: Basic Asterisk Config

Configure the following, using the extensions given in the Lab summary:

- voicemail for each extension
- a sample IVR
- a meetme conference
- a sample MOH stream

Here's a start on the configuration files:

voicemail.conf

```
[default]
xy00 => 1234,User 1,user1@email.address
xy01 => 1234,User 2,user2@email.address
xy02 => 1234,User 3,user3@email.address
```

extensions.conf

```
[phones]
; configure pattern match for local extensions
; e.g. _200X
exten => _xy0X,1,Dial(SIP/${EXTEN},15)
exten => _xy0X,n,Voicemail(u${EXTEN})
exten => _xy0X,n,Hangup()

; allow checking of voicemails. try it out!
exten => xy99,1,VoicemailMain()

; extension to allow dialling the IVR
exten => _xy10X,1,Goto(ivr-test,s,1)

[ivr-test]
; based on the slides, create an IVR which allows you to
; ring your extensions
```

If you're not sure about how specific applications work, from the Asterisk CLI try:

```
show applications
show application goto
```

Lab 3: Advanced Asterisk Configuration

1. Asterisk Database

Implement the following in extensions.conf:

```
[phones]
; start counting and store count progress in astdb
exten => xy30,1,Set(COUNT=${DB(test/count)})
exten => xy30,2,SayNumber(${COUNT})
exten => xy30,3,SetVar(COUNT=$[${COUNT} + 1]
exten => xy30,4,Set(DB(test/count)=${COUNT})
exten => xy30,5,Goto(1)
exten => xy30,102,Set(DB(test/count)=1)
exten => xy30,103,Goto(1)
```

Reload Asterisk, and have a look at the Asterisk DB

```
reload
database show
```

Now dial xy30, and look at the DB again. You should see a new key (test/count) in the DB containing the current count.

2. Implement Nightmode

We want the nightmode to work as follows:

We want to create an extension called xy50 for our 'main number'
We will create two new keys in the DB:
nightmode/open_time, and nightmode/close_time
When a call comes in, we will check to see if we are currently between those two times, and if so ring all three phones. If not, go straight to voicemail

Hints:

To manually set a DB key from the CLI:

```
database put family key value
```

Time based branching:

```
show application gotoif
```

Dialling multiple channels simultaneously:

```
Dial(SIP/1000&SIP/2000&SIP/3000)
```

3. Extension Macro

Look in the original /etc/extensions.conf (you should have moved it to orig_extensions.conf), and use it as a guide.

Create a simple extension macro to dial our extensions and branch to voicemail if not answered.

4. Set up Agents

Edit agents.conf - add three agents for you group to the bottom of the existing file:

```
agent => xy40,1234,Agent one
agent => xy41,1234,Agent two
agent => xy42,1234,Agent three
```

To enable Agent login and logout, add to extensions.conf:

```
[phones]
; hint in CLI, show application AgentCallbackLogin
exten => xy59,1,AgentCallbackLogin()
```

Reload Asterisk, then check the state of Agents before and after a login:

```
show agents
```

5. Set up a Queue

Edit queues.conf - use the existing defaults as a guide. Call the queue helpdesk (this is at the start of the file in [ ]). The important piece is to add to the bottom of queues.conf:

```
member => Agent/xy40
member => Agent/xy41
member => Agent/xy42
```

And in Extensions.conf create a means to enter the queue:

```
[phones]
exten => xy50,1,Queue(helpdesk)
```

Ring the queue with Agents all logged out, and all logged in.

6. Install Festival text to speech

Exit out of Asterisk and install Festival:

```
apt-get install festival
```

Configure Festival for Debian. Make /etc/festival.conf look like the following:

```
;; Enable access to localhost (needed by debian users)
(set! server_access_list '("localhost\\.localdomain" "localhost"))

;; set italian voice (comment the following 2 lines to use british_ameri-
can)
(language_italian)
(set! voice_default 'voice_pc_diphone)

;;; Command for Asterisk begin
(define (tts_textasterisk string mode)
    "(tts_textasterisk STRING MODE)
  Apply tts to STRING.  This function is specifically designed for
  use in server mode so a single function call may synthesize the string.
  This function name may be added to the server safe functions."
      (utt.send.wave.client (utt.wave.resample (utt.wave.rescale (utt.synth
                   (eval (list 'Utterance 'Text string))) 5) 8000)))
;;; Command for Asterisk end
```

To use Festival:

```
exten => 123,1,Festival('Hello World')
exten => 123,2,SetVar(speech='Hello World by variable')
exten => 123,3,Festival('${speech}')
```

Lab 4: Asterisk Exercises

1. Another Extensions Macro

Write an extension macro which looks up a database to get the following information:

callerID name
callerID number
Voicemail box
do not disturb flag

If the do not disturb flag is set, playback a prompt saying (sorry, <name> doesn't want to be disturbed). Make sure the macro correctly set the CallerID name and number.

2. DB lookup for incoming calls

Write a piece of code that does a DB lookup on inbound calls into the [incoming] context, looks up the number in the database, and uses the result to branch into the appropriate location in the dial plan.

In what circumstances do you think this would be handy?

3. Write a Prompt recording Macro.

This macro will need to take as input the filename to record, and optionally the format to record it in.

The macro needs to:

1. record the prompt
2. let the user play it back
3. let the user confirm they wish to use that prompt
4. save the prompt in the correct location

Note, Festival text to speech is handy to provide instructions here!

4. Write an application to ping a device

Create a context (starting with the 's' extension) which allows you to ping a device.

You'll need to work out how to accept DTMF input, run a ping command external to asterisk, and read the result back to the caller.