NOC Workshop
VoIP in the NOC labs
SANOG10

New Delhi, India

August 29 - September 2, 2007

Lab Summary

We only have limited time for this portion of the lab, so we will not be interconnecting our Asterisk boxes in this lab. For simplicity, we'll use the exact same dial-plan for each lab group.

Server logins are as you have set up in previous labs.

Lab 1: Initial Asterisk Install

1. Install Asterisk

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

We need to download the current Zaptel source and compile it, as the packaged version doesn't work correctly on our lab Ubuntu servers.

```
cd /usr/src
wget http://www.ubuntu.sun.ac.za/asterisk/1.2/libpri-1.2.4.tar.gz
wget http://www.ubuntu.sun.ac.za/asterisk/1.2/libpri-1.2.4.tar.gz
tar -xvf libpri-1.2.4.tar.gz
tar -xvf zaptel-1.2.16.tar.gz

cd /usr/src/libpri-1.2.4.tar
make
make install
```

Before compiling we need modify the zaptel makefile to enable the ztdummy module which will be providing Asterisk a timing signal for things such as music on hold and conferencing:

```
cd /usr/src/zaptel-1.2.16.tar
nano Makefile
```

Search for '# ztdummy' and remove the # (i.e. uncomment ztdummy from that line)

```
make
make install

modprobe ztdummy

ztcfg -v  # only do this 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: 2001, 2002, 2003. Add to the bottom of sip.conf, repeating for each of the three SIP peers:

```
[2001]
type=friend
host=dynamic
username=2001
secret=supersecret
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]
exten => 2001,1,Dial(SIP/2001)
exten => 2002,1,Dial(SIP/2002)
exten => 2003,1,Dial(SIP/2003)

exten => 500,1,Answer()
exten => 500,2,Playback(demo-echotest)
exten => 500,3,Echo
exten => 500,4,Playback(demo-echodone)
exten => 500,5,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 Pro.. > Default*

Enabled:	Yes
Username:	SIP extension you are configuring (e.g. 2001)
Authorisation User:	Same as *Username*
Password:	the password defined in sip.conf for this extension (e.g. supersecret)
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 500 and you should be able to hear yourself!

5. Running remote asterisk commands

To be able to integrate our PABX and NOC installations - such as NAGIOS2 - we need to be able to have a way to remotely query our Asterisk server. ‘asterisk -rx [command]’ allows us to do this.

From a shell on your server (i.e. NOT from the Asterisk CLI) run:

```
asterisk -rx ‘sip show peers’
```

This runs the command ‘sip show peers’ remotely. More advanced shell scripts can be written to do something useful with this information, for example, the following will reboot all currently registered phones (in this case, they are all assumed to be Linksys SPA941 phones):

```
for i in `asterisk -rx 'sip show peers' | grep 'OK (' | cut -c 28-41`; do curl
--anyauth -u username:password http://$i/admin/reboot; done
```

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

Here's a start on the configuration files:

voicemail.conf

```
[default]
2001 => 1234,User 1,user1@email.address
2002 => 1234,User 2,user2@email.address
2003 => 1234,User 3,user3@email.address
```

extensions.conf

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

; allow checking of voicemails. try it out!
exten => 501,1,VoicemailMain()

; extension to allow dialling the IVR
exten => 555,1,Goto(ivr-test,s,1)

[ivr-test]
; based on the slides, create an IVR which allows you to
; ring your extensions
exten => s,1,Answer
; play a sound file, we'll just use a demo one for this example
exten => s,2,Background(privacy-prompt)

; if someone enters a 1, ring extension 2001, 2-2002, 3-2003
exten => 1,1,Dial(SIP/2001)
exten => 2,1,Dial(SIP/2002)
exten => 3,1,Dial(SIP/2003)
```

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

; check if DB key exists, if not, jump to key_no_exist
; function DB_Exists returns 1 if the key exists, 0 if not
```

```
exten => 30,1,GotoIf(DB_Exists(test/count)?key_no_exist)

; begin the counting!
exten => 30,n(start),Set(COUNT=${DB(test/count)})
exten => 30,n,SayNumber(${COUNT})
exten => 30,n,Set(COUNT=$[${COUNT} + 1])
; update the DB
exten => 30,n,Set(DB(test/count)=${COUNT})
exten => 30,n,Goto(start)

; if we got here it is because the key didn't exist in the DB
; create the key
exten => 30,n(key_no_exist),Set(DB(test/count)=1)
; and jump back to the start to begin counting
exten => 30,n,Goto(start)
```

Reload Asterisk, and have a look at the Asterisk DB

```
reload
database show
```

Now dial ..30, and look at the DB again. You should see a new key (test/count) in the DB containing the current count.

2. Set up a Conference room

Make meetme.conf look like the following

```
[rooms]
;
; Usage is conf => confno[,pin]
;
conf => 101,1234
conf => 102,1234
```

And then set up an extension in extensions.conf to allow people to call the conference bridge

```
; add to your [default] context:
exten => 501,1,MeetMe(101|M)
exten => 502,1,MeetMe(102|M)
```

It's often useful for others to be able to call your conference bridge via SIP from across the internet. To do this we need to configure Asterisk to allow incoming anonymous SIP connections

Edit sip.conf and change or add the following under the [general] section:

```
context=fromsip
allowguest=yes
```

This tells Asterisk to allow anonymous incoming SIP calls. These calls and those from any peers without a context= line in their peer definition will be sent to this context, in our case fromsip. We don't want to be giving away unfettered access to everyone, so we will only allow access to certain services from this context.

Add the [fromsip] context to extensions.conf:

```
[fromsip]
; allow access to dial phones
exten => _200X,1,Goto(phones,${EXTEN},1)

; allow access to conference bridge 101
exten => 501,1,MeetMe(101|M)
; add a 'nicely named' extension as well, for those calling from softphones
exten => conference,1,MeetMe(101|M)
```

Callers can now call specific services on our PABX by sending a SIP call to *extension@ipaddress*, or *extension@domain.com*

You can setup the X-Lite softphone to call into the conference bridge by simply adding a phone book entry with the SIP address you wish to call. This is handy, as you don't need a SIP account or provider to be able to make these calls.

In X-Lite preferences, go into *phonebook*. Click on *New Entry*, and add the following:

```
NAME:     Workshop conference bridge
SIP URL:  sip:conference@your.server.ip.here
```

You can now call your conference bridge by clicking on the entry in the phonebook.

Get other people in the class to add an entry for your conference server in their phonebook and call it.

3. Set up Agents

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

```
agent => 2001,1234,Agent one
agent => 2002,1234,Agent two
agent => 2003,1234,Agent three
```

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

```
[phones]
; hint in CLI, show application AgentCallbackLogin
exten => 600,1,AgentCallbackLogin()
```

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

```
show agents
```

4. 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/2001
member => Agent/2002
member => Agent/2003
```

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

```
[phones]
exten => 555,1,Queue(helpdesk)
```

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

5. Install Festival text to speech

Exit out of Asterisk and install Festival:

```
apt-get install festival
```

Configure Festival for Debian / Ubuntu. 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_american)
(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}')
```

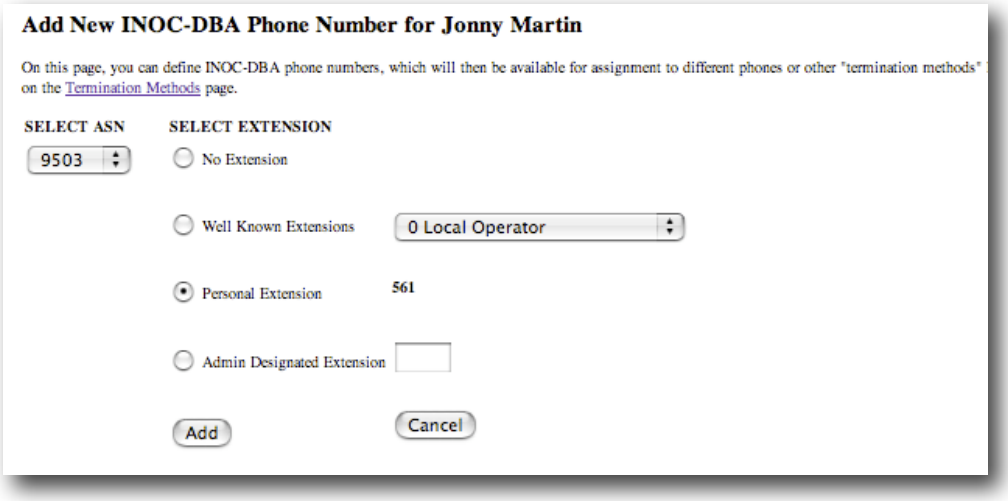
Lab 4: Connecting Asterisk to INOC-DBA

You will need to have set up an account and log in to the INOC-DBA administration system to do this.

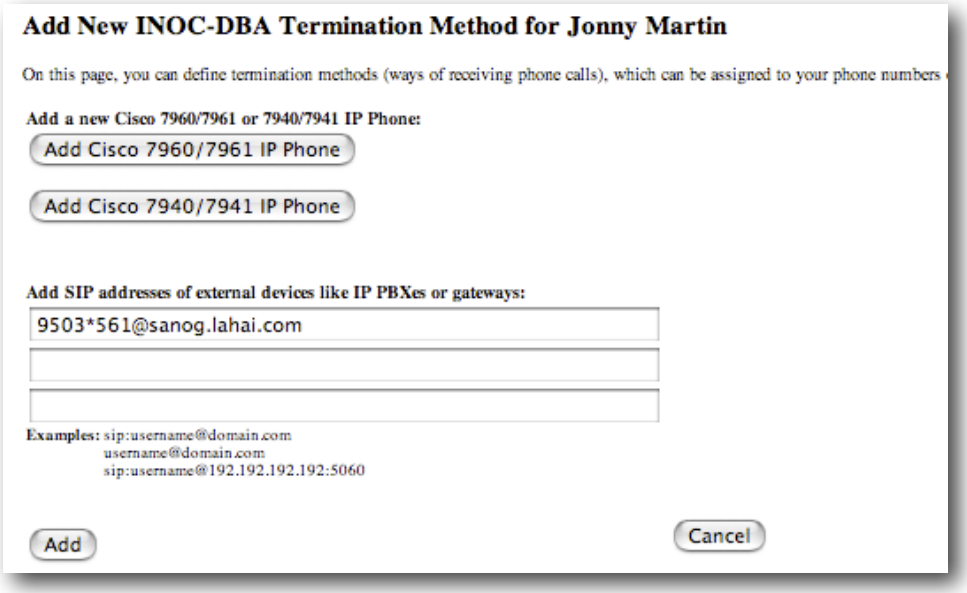
http://www.pch.net/inoc-dba/

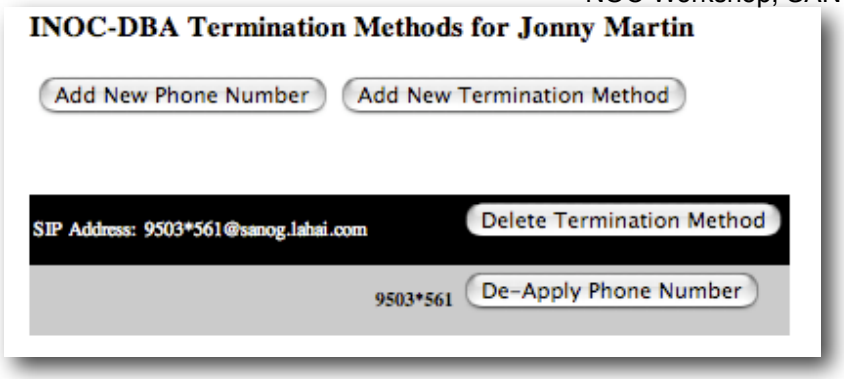
1. Set up INOC-DBA to send calls to your Asterisk server.
You need to set up a termination method through the INOC-DBA system to deliver calls to your asterisk server.

For this lab exercise we will will set up your INOC-DBA personal extension to terminate calls on your lab asterisk server. Select 'My Phone Numbers' from the menu and populate it accordingly:



Select 'Termination Methods' from the menu and add the IP address of your lab server, then select that as the termination method for your personal extension.





2. Configure Asterisk sip.conf
Asterisk needs to be configured to SIP REGISTER itself with the INOC-DBA servers. Add the following to the [general] section of sip.conf:

```
register => 9503*561:password:jonny@inoc-dba.pch.net/9503*561
```

Replacing 9503*561 with your INOC-DBA extension, and password:jonny@ with your password and login name.

This statement registers our Asterisk box with INOC-DBA. Inbound calls are sent to the default context.

3. Configure inbound calls
Inbound calls land in the default context. We want these calls to ring a phone, so add something like the following into the [default] context, substituting your details for SIP/2000 and 9503*561.

```
exten => 9503*561,1,Dial(SIP/2000,15)
exten => 9503*561,n,Voicemail(u2000)
exten => 9503*561,n,Hangup()
```

You may want to have inbound NOC calls ring multiple phones. Configure your INOC-DBA extension to ring multiple phones at once.

A nicer way to implement this is to use a GoTo statement in the default context to send inbound calls to 9503*561 to an extension elsewhere in your dialplan, enabling you to easily change the destination for inbound calls. Use this method to send a call to one of your existing extensions. This could be a phone, voicemail, conference, etc.

4. Configure outbound calls
Calls prefixed with a 9 will be sent out to INOC-DBA. We need to first strip the 9, and then set our outgoing callerID correctly. Add the following to the appropriate context in extensions.conf:

```
; This extension is for outgoing calls to inoc-dba
; 9 for an outside-inoc-dba-line
exten => _9.,1,SetCIDName(Jonny Martin)
exten => _9.,n,SetCIDNum(9503*561)
exten => _9.,n,Dial(SIP/${EXTEN:1}@inoc-dba)
exten => _9.,n,Hangup
```

Lab 5. Using VoIP to interact with your network

1. A voice interface to 'ping'
There are many ways to use VoIP as a means to monitor your network. In this very simple example, we will create an Asterisk extension which we can call, and then be prompted for a device to ping. It will then read back to us the ping results.

```
; add an extension to your [phones] context so we can call our new application
; in [phones]:
exten => 510,1,Goto(pingtest,s,1)

[pingtest]
exten => s,1,Ringing()
exten => s,2,Wait(2)
exten => s,3,Answer()
exten => s,4,Wait(1)
```

```
exten => s,5,Festival('Welcome to the ping master two thousand')
exten => s,6,Festival('Please enter host')
;exten => s,7,Setvar(rawhost="")

exten => _XXX*XXX*XXX*XXX,1,Setvar(rawhost=${EXTEN})
exten =>
_XXX*XXX*XXX*XXX,2,Setvar(host=${rawhost:0:3}.${rawhost:4:3}.${rawhost:8:3}.${rawhos
t:12:3})
exten => _XXX*XXX*XXX*XXX,3,System(ping -c 1 ${host} >> /tmp/ping.tmp)
exten => _XXX*XXX*XXX*XXX,4,System(text2wave /tmp/ping.tmp -o /tmp/ping.alaw -otype
alaw)
exten => _XXX*XXX*XXX*XXX,5,Playback(/tmp/ping)
exten => _XXX*XXX*XXX*XXX,6,System(rm /tmp/ping.*)

exten => i,1,Playback(invalid)
exten => i,2,Goto(s,6)
```

2. Having Asterisk automatically create calls

It's useful to be able to have Asterisk automatically create a call, for example when a NAGIOS alert is not acknowledged.

First, create a context with Asterisk which will play an outgoing message:

```
[outboundmsg1]
exten => s,1,Answer
exten => s,2,Wait(1)
exten => s,3,Background(workshop-msg1)         ; playback pre-recorded message
; "Press 1 to replay or 2 to acknowledge receiving this message"
exten => s,4,Background(workshop-how_to_ack)

exten => 1,1,Goto(s,5)   ; replay message
exten => 2,1,Goto(msgack,s,1) ; acknowledge message

exten => t,1,Playback(vm-goodbye)
exten => t,2,Hangup
; at this point we could do something like reschedule the call to try again later
; or send an email saying the msg was not received, or...

[msgack]
exten => s,1,Playback(workshop-thankyou)
exten => s,2,Playback(vm-goodbye)
exten => s,3,Hangup
; at this point we might want to log the message acknowledgement somewhere
; and perhaps trigger some additional processing
```

A simple means to record a message:

```
; add to the [phones] context
; Then call
;   3051 to Record a new outbound msg1
;   3061 to Record the msg played when the recipient acks the message
;   3062 to Record the "How to ACK message"
; After dialing one of the extensions above:
;   Wait for the record start tone
;   Record your message
;   Press # to stop recording
;   Listen to an automatic playback of your new message
;
; outbound msg1
exten => 3051,1,Wait(2)
exten => 3051,2,Record(workshop-msg1:gsm)
exten => 3051,3,Wait(2)
exten => 3051,4,Playback(workshop-msg1)
exten => 3051,5,wait(2)
exten => 3051,6,Hangup
;
```

```
; Msg played when msg is acked
exten => 3061,1,Wait(2)
exten => 3061,2,Record(workshop-thankyou:gsm)
exten => 3061,3,Wait(2)
exten => 3061,4,Playback(workshop-thankyou)
exten => 3061,5,wait(2)
exten => 3061,6,Hangup
;
; Msg played after outbound msg: "Press 1 to replay or 2 to acknowledge receiving
this message"
exten => 3062,1,Wait(2)
exten => 3062,2,Record(workshop-how_to_ack:gsm)
exten => 3062,3,Wait(2)
exten => 3062,4,Playback(workshop-how_to_ack)
exten => 3062,5,wait(2)
exten => 3062,6,Hangup
```

Now we need to trigger Asterisk to make the call:

```
; Creating a call file to call 2001 in context phones
;
cat <<EOF > /var/spool/asterisk/tmp01
Channel: Local/2001@phones/n
Callerid: 7022340175
MaxRetries: 5
RetryTime: 300
WaitTime: 45
Context: outboundmsg1
Extension: s
Priority: 1
EOF

mv /var/spool/asterisk/tmp01 /var/spool/asterisk/outgoing
; or, if you have created this on another box, you could SCP the callfile to your
; asterisk box
```