SANOG25, Kandy, Sri Lanka

Asterisk Advance Config

Anowar Hasan Sabir
Sumon Ahmed Sabir,
Simon Sohel Baroi, Senevi Herath

BDCOM

Variable Expressions

- Variables used to
  - reduce configuration complexity
  - add clarity
  - provide additional dialplan logic
- Basic expressions allow us to perform basic mathematical calculations
  - exten => 501,1,Set(Count=1)
    exten => 501,2,Set(Newcount=$[${Count}+1])
    exten => 501,3,SayNumber(${NewCount})

Substrings

- ${variable:offset:length}
- Returns the substring of ‘variable’ of length ‘length’, starting at offset
- Commonly used to strip access codes
  - exten => 1X.,1,Dial(SIP/${EXTEN:1})
  - Dials the extension minus the initial ‘1’
  - If ‘length’ is omitted, the rest of the string is returned
- To concatenate two strings, simply write them together:
  - ${string1}${string2}

Variable Operator

- Boolean operators (non-zero = true, zero=false)
  - Or - var1 | var2
  - And - var1 & var2
  - Comparisons - var1 {=, >, >=, <, <=, !=} var2
- Mathematical operators
  - Addition and subtraction - var1 {+, -} var2
  - Multiplication, integer division, remainder - var1 {*, /, %} var2

Dialplan Functions

- Basic syntax:
  - FUNTION_NAME(argument)
- To reference the value of a function
  - ${FUNCTION_NAME(argument)}
  - can be nested, i.e. ‘argument’ above replaced with another function reference
- Used for string manipulation

Dialplan Functions

- exten => 502,1,Set(TEST=example)
  exten => 502,2,SayNumber(${LEN(${TEST})})
  - Len() returns the length of a string
- Many more...

Functions

```
sujon-desktop*CLI> core show  functions
Installed Custom Functions:
-------------------------------------------------------------------------------
AES_DECRYPT           AES_DECRYPT(key,string)             Decrypt a string encoded in base64
with AES given a 16 character key.
AES_ENCRYPT           AES_ENCRYPT(key,string)             Encrypt a string with AES given a
16 character key.
AGC                   AGC(channeldirection)               Apply automatic gain control to
audio on a channel.
AGENT                 AGENT(AgentId[:item])               Gets information about an Agent
AMI_CLIENT            AMI_CLIENT(loginname,field)         Checks attributes of manager
accounts
ARRAY                 ARRAY(var1[,var2[,...][,varN]])     Allows setting multiple variables
at once.
AST_CONFIG            AST_CONFIG(config_file,category,var Retrieve a variable from a
configuration file.
AST_SORCERY           AST_SORCERY(module_name,object_type Get a field from a sorcery object
AUDIOHOOK_INHERIT     AUDIOHOOK_INHERIT()                 DEPRECATED: Used to set whether an
audiohook may be inherited to another channel. Due to architectural changes in Asterisk 12,
audiohook inheritance is performed automatically and this function now lacks function
```

Functions

```
BASE64_DECODE         BASE64_DECODE(string)              Decode a base64 string.
BASE64_ENCODE         BASE64_ENCODE(string)              Encode a string in base64.
BLACKLIST             BLACKLIST()                        Check if the callerid is on the
blacklist.
CALENDAR_BUSY         CALENDAR_BUSY(calendar)            Determine if the calendar is
marked busy at this time.
CALENDAR_EVENT        CALENDAR_EVENT(field)              Get calendar event notification
data from a notification call.
CALENDAR_QUERY        CALENDAR_QUERY(calendar[,start[,end Query a calendar server and store
the data on a channel
CALENDAR_QUERY_RESUL  CALENDAR_QUERY_RESULT(id,field[,ent Retrieve data from a previously
run CALENDAR_QUERY() call
CALENDAR_WRITE        CALENDAR_WRITE(calendar,field[,...] Write an event to a calendar
CALLCOMPLETION        CALLCOMPLETION(option)             Get or set a call completion
configuration parameter for a channel.
CALLERID              CALLERID(datatype[,CID])           Gets or sets Caller*ID data on the
channel.
CALLERPRES            CALLERPRES()                       Gets or sets Caller*ID
presentation on the channel.
CDR                   CDR(name[,options])                Gets or sets a CDR variabl
CDR_PROP              CDR_PROP(name)                     Set a property on a channe
CHANNEL               CHANNEL(item)                      Gets/sets various pieces of info
```

GotoIf

```
[Syntax]
GotoIf(condition?[labeliftrue][:labeliffalse])

[Arguments]
labeliftrue
    Continue at <labeliftrue> if the condition is true.
Takes the form
    similar to Goto() of [[context,]extension,]priority.
labeliffalse
    Continue at <labeliffalse> if the condition is false.
Takes the form
    similar to Goto() of [[context,]extension,]priority.
```

GotoIfTime

```
[Syntax]
GotoIfTime(times,weekdays,mdays,months[,timezone]?
[labeliftrue][:labeliffalse])

[Arguments]
labeliftrue
    Continue at <labeliftrue> if the condition is true.
Takes the form
    similar to Goto() of [[context,]extension,]priority.
labeliffalse
    Continue at <labeliffalse> if the condition is false.
Takes the form
    similar to Goto() of [[context,]extension,]priority.
```

ExecIf

[Description]
If <expr> is true, execute and return the result of <appiftrue(args)>.
If <expr> is true, but <appiftrue> is not found, then the application will
return a non-zero value.

[Syntax]
ExecIf(expression?appiftrue(args)[:appiffalse(args)])

Macro

```
[Syntax]
Macro(name[,arg1[,arg2[,...]]])

[Arguments]
name
    The name of the macro
```

- Avoids repetition in the dial plan
- Akin to building a function in the dial plan
- Useful for building standard phone dialling logic
- Uses extra specific channel variables:
  ${ARGn}: The nth argument passed to the macro
  ${MACRO_CONTEXT}: Context of the extension that triggered this macro
  ${MACRO_EXTEN}: The extension that triggered this macro
  ${MACRO_PRIORITY}: The priority in the extension where this macro was triggered

Macro Example

```
[macro-voicemail]
exten => s,1,Dial(SIP/101,20)
same => n,GotoIf($["${DIALSTATUS}" = "BUSY"]?busy:unavail)
same => n(unavail),VoiceMail(101@default,u)
same => n,Hangup()
same => n(busy),VoiceMail(101@default,b)
same => n,Hangup()
```

calling the the Macro

```
exten => 101,1,Macro(voicemail)
```

Macro with Arguments

```
[macro-voicemail]
exten => s,1,Dial(SIP/${ARG1},20)
same => n,GotoIf($["${DIALSTATUS}" = "BUSY"]?busy:unavail)
same => n(unavail),VoiceMail(${MACRO_EXTEN}@default,u)
same => n,Hangup()
same => n(busy),VoiceMail(${MACRO_EXTEN}@default,b)
same => n,Hangup()
```

calling the the Macro with argument.

```
exten => 101,1,Macro(voicemail,${EXTEN})
```

IVR

"Thank you for calling Customer Service.
If you're calm and rational, press 1.
If you're a whiner, press 2.
If you're a hot head, press 3...."

Interactive Voice Response

- Interactive Voice Response (IVR) is inherent to the Asterisk dialplan
- Simply a matter of playing prompts, waiting, accepting input in a channel, and moving around the dial plan
  - Useful applications:
    - Background(prompt-to-play-whilst-waiting-for-intput)
    - Playback(prompt-to-play-whilst-NOT-accepting-input)
    - Goto(context,extension,priority)
    - Dial(SIP/2000)
    - Wait(seconds)

IVR Example

```
[test-ivr]
exten => s,1,Answer()
exten => s,2,Background(enter-ext-of-person)
exten => s,3,WaitExten(5)
exten => 1,1,Playback(digits/1)
exten => 1,2,Goto(incoming,s,1)
exten => 2,1,Playback(digits/2)
exten => 2,2,Goto(incoming,s,1)
exten => i,1,Playback(pbx-invalid)
exten => i,2,Goto(incoming,s,1)
exten => t,1,Playback(vm-goodbye)
exten => t,2,Hangup()
[phones]
; allow our phones to dial into the IVR
exten => 2010,1,Goto(test-ivr,s,1)
```

AGI

```
[Syntax]
AGI(command[,arg1[,arg2[,...]]])
```

- Asterisk Gateway Interface
- Dial plan can call Perl, Python, PHP scripts
- AGI script reads from STDIN to get information from Asterisk
- AGI script writes data to STDOUT to send information to Asterisk
- AGI script can write to STDERR to send debug information to the console
- Scripts stored in /usr/share/asterisk/agi-bin/ on Debian
- exten => 520,1,AGI(/path/to/agi-script.agi)

AGI Scripts

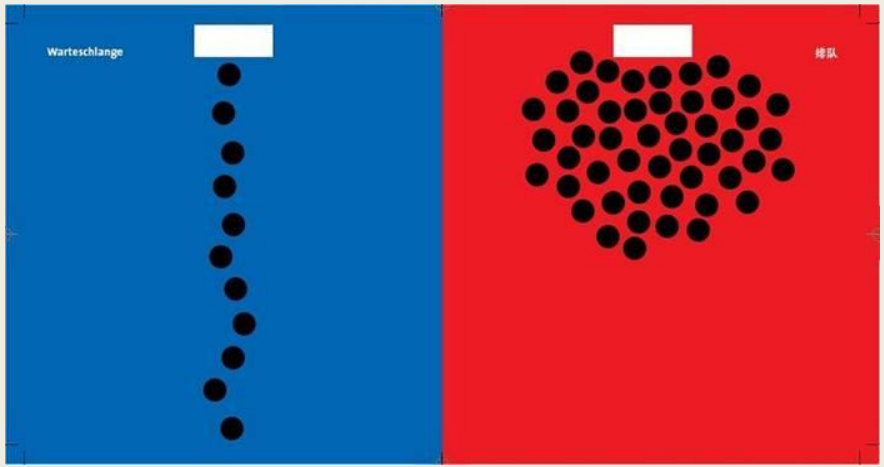
- Very very powerful
- A2Billing uses them to implement a complete billing system
- All the relevant call data is sent to the AGI
- MySQL lookups performed
- Relevant dial command returned to Asterisk
- Database updated at end of call

Agents

- Users can log in as an Agent
- Maps current extension to that user's Agent
- Agent can then be logged into queues
- Agents can log in / out at will, follow-me functionality
- Agents functionality still quite buggy - best not to use for anything complex

Queues

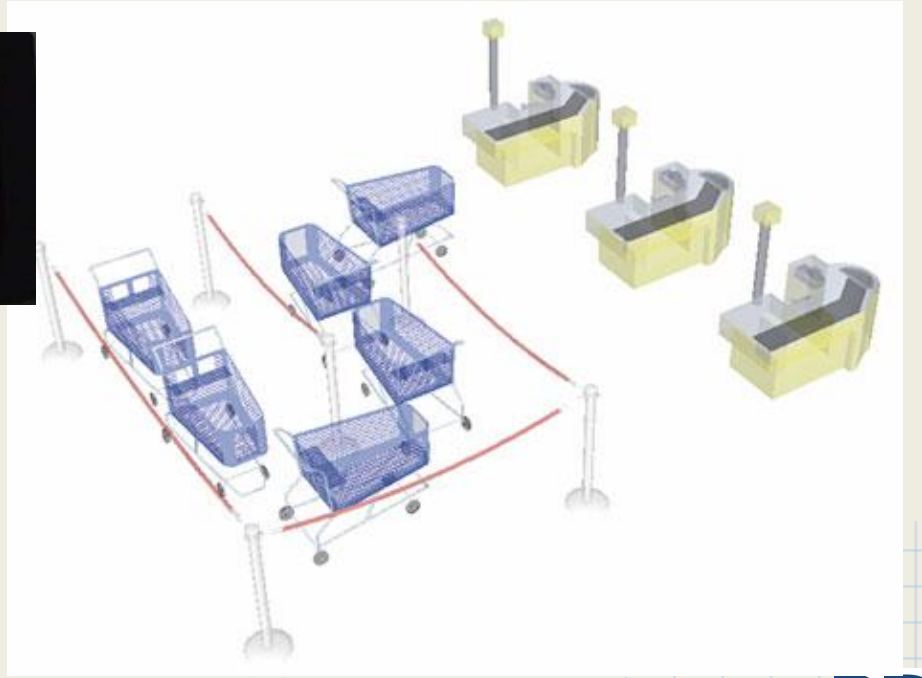
You need a call center app to make your service smooth by keeping your client happy and in line.



Queues

- Reasonably powerful queuing support within Asterisk
- Queues can have static or dynamic members
- Members can be channels, or Agents
- Automatic distribution of calls based on queue strategy

Queues



queues.conf

```
Configuring first ACD.
[general]
persistentmembers = yes
[q_sample]
announce-frequency = 30
periodic-announce-frequency = 15
announce-holdtime = yes
announce-position = yes
announce-position-limit = 1
announce-round-seconds = 10
periodic-announce = queue-periodic-announce
queue-youarenext = queue-youarenext ; ("You are now first in line.")
queue-thereare = queue-thereare ; ("There are")
strategy = rrmemory
timeout = 20
retry = 5
maxlen = 0
ringinuse = no
announce-frequency = 0
announce-holdtime = no
servicelevel = 15
monitor-type = MixMonitor
monitor-format = wav
wrapuptime = 5
music = default
```

queues.conf

Add static Members/Agents

After the queue configuration add member like this.

```
member=> SIP/2000,Mr. ABC
member=> SIP/2001,Mr. EFG
member=> SIP/2002,Mr. HIJ
```

If you command "queue show q-sample", it will give you this output in asterisk console.

```
q-sample has 0 calls (max unlimited) in 'rrmemory' strategy (0s holdtime,
0s talktime), W:0, C:0, A:0, SL:0.0% within 15s
   Members:
      SIP/2002 (Unavailable) has taken no calls yet
      SIP/2000 (Unavailable) has taken no calls yet
      SIP/2001 (Unavailable) has taken no calls yet
   No Callers
```

Queues cont.

in your dialplan write this ...

```
exten => _YOUR_NUM,1, Answer()
same => n,Queue(q-sample,tTwi)
same => n, Hangup()
```

you are on.....

Make sure all three agents are in live sip extension. you will start to receive calls in an arranged manner.

Queues cont.

Make Agents Dynamic.

To do so first you have to remove the members from the queues.conf. and then try something new in your dialplan.

login >>>>

```
exten => 11,1,AddQueueMember(q-sample,SIP/${CALLERID(num)})
```

logout >>>>

```
exten => 12,1,RemoveQueueMember(q-sample,SIP/${CALLERID(num)})
```

Queues cont.

Pause >>>>

```
exten => 13,1,PauseQueueMember(q-sample,SIP/${CALLERID(num)})
```

UnPause >>>>

```
exten => 14,1,UnpauseQueueMember(q-sample,SIP/${CALLERID(num)})
```

MeetMe.

- Powerful application built in to Asterisk
- Some use Asterisk purely for it's conferencing abilities
- Ad Hoc MeetMe conferencing, or individual conference rooms with PIN

```
/etc/asterisk/meetme.conf
  ; Configuration file for MeetMe simple conference rooms
  ;
  [rooms]
  ; Usage is conf => confno[,pin]
  ;
  conf => 101,1234
  conf => 102,2345
  /etc/asterisk/extensions.conf
  exten => 5101,1,Meetme(101|M)
  exten => 5102,2,Meetme(102|M)
```

ConfBridge

Same as MeetMe, Newer and advanced…

```
[general]

[default_user]
type=user
music_on_hold_when_empty=yes
music_on_hold_class=default
;announce_user_count_all=yes
;announce_join_leave=yes
;dsp_drop_silence=yes
denoise=yes
pin=654
```

ConfBridge

```
[default_bridge]
type=bridge
max_members=20
mixing_interval=10
internal_sample_rate=auto
;record_conference=yes
video_mode = follow_talker
```

ConfBridge

```
[sample_user_menu]
type=menu
*=playback_and_continue(conf-usermenu)
*1=toggle_mute
1=toggle_mute
*4=decrease_listening_volume
4=decrease_listening_volume
*6=increase_listening_volume
6=increase_listening_volume
*7=decrease_talking_volume
7=decrease_talking_volume
*8=leave_conference
8=leave_conference
*9=increase_talking_volume
9=increase_talking_volume
```

ConfBridge

```
[sample_admin_menu]
type=menu
*=playback_and_continue(conf-adminmenu)
*1=toggle_mute
1=toggle_mute
*2=admin_toggle_conference_lock ; only applied to admin users
2=admin_toggle_conference_lock  ; only applied to admin users
*3=admin_kick_last       ; only applied to admin users
3=admin_kick_last        ; only applied to admin users
*4=decrease_listening_volume
4=decrease_listening_volume
*6=increase_listening_volume
6=increase_listening_volume
*7=decrease_talking_volume
7=decrease_talking_volume
*8=no_op
8=no_op
*9=increase_talking_volume
9=increase_talking_volume
```

ConfBridge

Calling it from the dialplan…

```
exten => 9999,1,Answer()
same => n,ConfBridge(2,default_bridge,default_user,sample_user_menu)
same => n,Hangup()
```

Monitor

"Thank you for calling Customer Service.
This call may be monitored so we can play it back
to your mother if you're rude or use bad words."

Monitor

[Description]
Used to start monitoring a channel. The channel's input and output voice packets are logged to files until the channel hangs up or monitoring is stopped by the StopMonitor application.
By default, files are stored to "/var/spool/asterisk/monitor/". Returns '-1' if monitor files can't be opened or if the channel is already monitored, otherwise '0'.

```
[Syntax]
Monitor([file_format[:urlbase]][,fname_base[,options]])
```

[Arguments]
file_format
    optional, if not set, defaults to 'wav'
fname_base
    if set, changes the filename used to the one specified.

LAB

Advance Asterisk Configuration….

CDR MySQL Backend

cdr_mysql.conf

```
[global]
hostname=localhost
dbname=asterisk
table=cdr
password=xyz
user=root
port=3306
sock=  /var/run/mysqld/mysqld.sock
```

CDR Database

```
CREATE TABLE `cdr` (
`id` INT(11) UNSIGNED NOT NULL AUTO_INCREMENT,
`calldate` DATETIME NOT NULL DEFAULT '0000-00-00 00:00:00',
`clid` VARCHAR(80) NOT NULL DEFAULT '',
`src` VARCHAR(80) NOT NULL DEFAULT '',
`dst` VARCHAR(80) NOT NULL DEFAULT '',
`dcontext` VARCHAR(80) NOT NULL DEFAULT '',
`lastapp` VARCHAR(200) NOT NULL DEFAULT '',
`lastdata` VARCHAR(200) NOT NULL DEFAULT '',
`duration` FLOAT UNSIGNED NULL DEFAULT NULL,
`billsec` FLOAT UNSIGNED NULL DEFAULT NULL,
`disposition` ENUM('ANSWERED','BUSY','FAILED','NO ANSWER','CONGESTION') NULL DEFAULT NULL,
`channel` VARCHAR(50) NULL DEFAULT NULL,
`dstchannel` VARCHAR(50) NULL DEFAULT NULL,
`amaflags` VARCHAR(50) NULL DEFAULT NULL,
`accountcode` VARCHAR(20) NULL DEFAULT NULL,
`uniqueid` VARCHAR(32) NOT NULL DEFAULT '',
`userfield` FLOAT UNSIGNED NULL DEFAULT NULL,
`answer` DATETIME NOT NULL,
`end` DATETIME NOT NULL,
PRIMARY KEY (`id`),
INDEX `calldate` (`calldate`),
INDEX `dst` (`dst`),
INDEX `src` (`src`),
INDEX `dcontext` (`dcontext`),
INDEX `clid` (`clid`),
)
COLLATE='utf8_bin'
ENGINE=InnoDB;
```

Asterisk CLI

- Should be quite familiar with it by now
- Can run remote Asterisk CLI commands from server
- asterisk -rx “sip reload”
- Primarily useful for triggering reloads and setting DB keys

Asterisk Manager Interface

- Allows client programs to connect to Asterisk
- Issues commands and reads events
- Used by Flash Operator Panel to keep track of Asterisk's state
- Telnet to the listening TCP/IP port (5038 by default)
- Login checked against credentials in manager.conf
- Specific message types subscribed to in manager.conf

Manager API Commands

Action	Synopsis
WaitEvent	Wait for an event to occur.
DeviceStateList	List the current known device states.
PresenceStateList	List the current known presence states.
QueueReset	Reset queue statistics.
QueueReload	Reload a queue, queues, or any sub-section of
QueueRule	Queue Rules.
QueueMemberRingInUse	Set the ringinuse value for a queue member.
QueuePenalty	Set the penalty for a queue member.
QueueLog	Adds custom entry in queue_log.
QueuePause	Makes a queue member temporarily unavailable.
QueueRemove	Remove interface from queue.
QueueAdd	Add interface to queue.
QueueSummary	Show queue summary.
QueueStatus	Show queue status.
Queues	Queues.
ControlPlayback	Control the playback of a file being played to
PlayDTMF	Play DTMF signal on a specific channel.

LAB

Playing with MySQL CDR and AMI