Asterisk - Advanced Configuration

APRICOT2010 VoIP Workshop
Kuala Lumpur, February 2010

Jonny Martin - jonny@jonnynet.net
Steve Kurzeja - stevek@callplus.co.nz

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 Operators

- 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

```
*CLI> show functions
Installed Custom Functions:
--------------------------------------------------------------------------
URIENCODE               URIENCODE(<data>)                 Encodes a string to URI-
safe encoding.
URIDECODE               URIDECODE(<data>)                 Decodes an URI-encoded
string.
SQL_ESC                 SQL_ESC(<string>)                 Escapes single ticks for
use in SQL statements
ODBC_PRESENCE           ODBC_PRESENCE(<arg1>[...[,<argN>]])  Runs the referenced
query with the specified arguments
ODBC_ANTIGF             ODBC_ANTIGF(<arg1>[...[,<argN>]])   Runs the referenced
query with the specified arguments
ODBC_SQL                ODBC_SQL(<arg1>[...[,<argN>]])      Runs the referenced
query with the specified arguments
TXTCIDNAME              TXTCIDNAME(<number>)              TXTCIDNAME looks up a
caller name via DNS
ENUMLOOKUP              ENUMLOOKUP(number[,Method-type[,opt  ENUMLOOKUP allows for
general or specific querying of NAPTR records or counts of NAPTR types for ENUM or
ENUM-like DNS pointers
CALLERID                CALLERID(datatype)                Gets or sets Caller*ID
data on the channel.
ARRAY                   ARRAY(var1[,var2[...][,varN]])    Allows setting multiple
```

Asterisk Database

- astdb - simple database forms part of Asterisk

- Dial plan and CLI can insert and remove data

- Data stored in a file, so is retained across Asterisk reloads and server reboots

- Data stored in groupings of families containing keys

  - exten => s,1,Set(DB(family/key)=${some_variable})

  - exten => s,1,Set(DB(system/nightmode_on)=1)

  - exten => s,1,Dial(${DB(exten/${EXTEN}/dial_string)},15)

Asterisk Database - Example

```
; 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)
```

GotoIf

```
;    GotoIf(condition?label1[:label2])
;
; Go to label1 if condition is true or to next step (or label2 if defined) if
condition is false, or
;
;    GotoIf(condition?[label1]:label2)
;
; Go to next step (or label1 if defined) if condition is true or to label2 if
condition is false.
```

Macros

- 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-stdexten]

;
; Standard extension macro:
;    ${ARG1} - Extension   (we could have used ${MACRO_EXTEN} here as well
;    ${ARG2} - Device(s) to ring
;
; ring the interface for 20sec max
exten => s,1,Dial(${ARG2},20)
; jump based on status (NOANSWER,BUSY,CHANUNAVAIL,CONGESTION,ANSWER)
exten => s,2,Goto(s-${DIALSTATUS},1)

exten => s-NOANSWER,1,Voicemail(u${ARG1}) ; If unavailable, send to voicemail
exten => s-NOANSWER,2,Goto(default,s,1)   ; If they press #, return to start

exten => s-BUSY,1,Voicemail(b${ARG1})       ; If busy, send to voicemail w/ busy
announce
exten => s-BUSY,2,Goto(default,s,1)         ; If they press #, return to start

exten => _s-.,1,Goto(s-NOANSWER,1)          ; Treat anything else as no answer

exten => a,1,VoicemailMain(${ARG1})         ; If they press *, send to VoicemailMain
```

AGI Scripts

- 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

agents.conf

```
/etc/asterisk/agents.conf

[general]
; Define whether callbacklogins should be stored in astdb for persistence
persistentagents=yes

[agents]
;autologoff=15 ; time (s) before agent auto logoff if no answer
;ackcall=no
wrapuptime=1000
;musiconhold => default
;updatecdr=no
; Enable recording calls addressed to agents. It's turned off by default.
recordagentcalls=yes
;recordformat=gsm

; agent => agentid,agentpassword,name
group=1  ; Junior NOC staff
agent => 600,1234,Lilly

group=2  ; Senior NOC staff
agent => 610,1234,Steve
```

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.conf

```
/etc/asterisk/queues.conf

[general]
; Store each dynamic agent in each queue in the astdb for persistence
persistentmembers = yes

; Queue(queuename|[options]|[optionalurl]|[announceoverride]|[timeout])
; example: Queue(dave|t|||45)

[noc]
musiconhold = default
strategy = ringall ; ringall, roundrobin, leastrecent, fewest calls, random, rrmemory

servicelevel = 30  ; SLA setting (s). stats for calls answered in this time
timeout=15         ; How long the phone rings before it's considered a timeout
retry=0            ; How long do we wait before trying all the members again?
; Weight of queue - when compared to other queues, higher weights get preference
weight=2
wrapuptime=5       ; how long before sending agent another call
maxlen = 0         ; of queue, 0 for no maximum

; How often to announce queue position and/or estimated holdtime to caller (0=off)
announce-frequency = 0
;announce-holdtime = yes|no|once
;announce-round-seconds = 10
; How often to make any periodic announcement (see periodic-announce)
;periodic-announce-frequency=60
```

Queuing Example

```
; Using Agents
; agent login to helpdesk queue
exten => *4,1,Answer()
exten => *4,n,AddQueueMember(noc|Agent/${CALLERID(NUM)})
exten => *4,n,AgentCallbackLogin(${CALLERID(NUM)}||q${CALLERID(NUM)}@sip)
exten => *4,n,Hangup()

; agent logout from noc queue
; note # is sent through by as a %23 in some sip headers
; so may need to repeat with exten => %23
exten => #4,1,Answer()
; send trigger to flash panel
exten => #4,n,System(/usr/sbin/asterisk -rx "agent logoff Agent/${CALLERID(NUM)}")
exten => #4,n,RemoveQueueMember(noc|Agent/${CALLERID(NUM)})
exten => #4,n,Playback(agent-loggedoff)
exten => #4,n,Hangup

; Or, using dynamic login of channel instead of agents, doesn't send triggers to flash panel
exten => *4,1,Answer()
exten => *4,n,AddQueueMember(noc|${CALLERID(NUM)})
exten => *4,n,Playback(logged-in)
exten => *4,n,Hangup()

exten => #4,n,RemoveQueueMember(noc|${CALLERID(NUM)})
exten => #4,n,Playback(agent-loggedoff)
exten => #4,n,Hangup
```

Festival

- Festival - Open sources text to speech engine

  - http://www.cstr.ed.ac.uk/projects/festival/

- Text to speech is a bit rough, but useable

- Easy to use once installed

- Useful for putting together quick IVRs

```
exten => 1,1,Festival('Record your message now')
exten => 1,n,Record(filename:alaw)
exten => 1,n,Festival('You recorded')
exten => 1,n,Playback(filename)
exten => 1,n,Festival('message saved.')
exten => 1,n,Goto(s,1)
```

Lab 3: Advanced Asterisk Configuration

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 API

- 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

Asterisk Manager API Commands

Action	Privilege	Synopsis
AbsoluteTimeout	call,all	Set Absolute Timeout
AgentCallbackLo	agent,all	Sets an agent as logged in by callback
AgentLogoff	agent,all	Sets an agent as no longer logged in
Agents	agent,all	Lists agents and their status
ChangeMonitor	call,all	Change monitoring filename of a channel
Command	command,all	Execute Asterisk CLI Command
DBGet	system,all	Get DB Entry
DBPut	system,all	Put DB Entry
Events	<none>	Control Event Flow
ExtensionState	call,all	Check Extension Status
Getvar	call,all	Gets a Channel Variable
Hangup	call,all	Hangup Channel
IAXnetstats	<none>	Show IAX Netstats
IAXpeers	<none>	List IAX Peers
ListCommands	<none>	List available manager commands
Logoff	<none>	Logoff Manager
MailboxCount	call,all	Check Mailbox Message Count
MailboxStatus	call,all	Check Mailbox
Monitor	call,all	Monitor a channel
Originate	call,all	Originate Call
ParkedCalls	<none>	List parked calls

Asterisk Performance

- Performance heavily dependant on what your Asterisk server is doing

- 'Switching' calls - can easily get up to ~200 calls/sec

- Terminating media streams - around 30 simultaneous calls on a fast server

- Codecs - low bitrate codecs typically require a lot of CPU

Purpose	Number of channels	Minimum recommended
Hobby system	No more than 5	400-MHz x86, 256 MB RAM
SOHO[a] system	5 to 10	1-GHz x86, 512 MB RAM
Small business system	Up to 15	3-GHz x86, 1 GB RAM
Medium to large system	More than 15	Dual CPUs, possibly also multiple servers in a distributed architecture