/etc/asterisk/extensions/Receptionist.inc

Dialplan Notes

Dialplan Global Variables

The Receptionist dialplan uses several GLOBAL variables to direct it's operation. Two variables name public and private telecommunications channels, one names the public voicemail mailbox, and three specify time-of-day intervals for pre-recorded greetings.

Variable	What goes in it	Example value
`HANDSET`	the channel specification of the private "inside" telephone device	`SIP/Grandstream`
`OUTSIDE`	the channel specification of the public incoming telephone interface	`SIP/POTSviaVOIP`
`MAILBOX`	the mailbox # and context of the public voicemail mailbox	`0@default`
`MORNING`	the time of day in which callers will hear the `Receptionist/GoodMorning` greeting	`00:00-11:59`
`AFTERNOON`	the time of day in which callers will hear the `Receptionist/GoodAfternoon` greeting	`12:00-16:59`
`EVENING`	the time of day in which callers will hear the `Receptionist/GoodEvening` greeting	`17:00-23:59`

Dialplan [Handset] context

The inside telephone device (channel variable `HANDSET`) should reference this context for all calls that it initiates.

Extensions

This context includes three predefined extensions; one to access voicemail, and two to manage this call forward feature:

`#0`	accesses the voicemail saved in voicemailbox `0` of the `default` voicemail context
`#1`	enables External Call Forwarding by setting the **E**xternal **C**all **F**orward **N**umber
`#2`	cancels External Call Forwarding

If the dialed number is not one of the above, it is dialed directly on the channel defined by the channel variable `OUTSIDE`. There is no need to "dial 9 for an outside line"; anything you dial (other than #0, #1, or #2) is immediately dialed on the outside line.

Audio

This context uses the following audio files, stored in **`/usr/share/asterisk/sounds`**. These are recordings of fixed phrases that the dialplan will play back as you manipulate the *External Call Forwarding* feature. The corresponding texts are suggestions only.

Playback()	*soundclip says*
`Receptionist/EnterCallFwdNumber`	*"Please dial the number to forward calls to, followed by pound."*
`Receptionist/NoCallForwardEntered`	*"No call forward number was entered"*
`Receptionist/CallForwardSet`	*"Incoming call forwarding has been set to"*

Receptionist/CallForwardCancelled	*"Incoming Call forwarding is cancelled."*
Receptionist/CallForwardUnchanged	*"Incoming call forwarding is not changed."*

AstDB

This context manipulates a "call forwarding" setting stored in the AstDB.

Settings/ECFN

We allow for the option of forwarding our calls through the outbound channel. If `Settings/ECFN` is set, it contains the external phone number we will forward all accepted calls to (assuming that we can originate an outbound call while receiving an inbound call from outside; this works with SIP and IAX channels, but not necessarily with other channels).

Dialplan [Receptionist] context

The outside telephone connection (channel variable `OUTSIDE`) should reference this context for all calls that it receives.

Audio

This context uses the following audio files, stored in **/usr/share/asterisk/sounds**. These are recordings of fixed phrases that the dialplan will play back to callers as it processes their call. The corresponding texts are suggestions only; these phrases convey the intent of the processing in the mildly polite manner you might get from a butler or receptionist.

Please note that you may wish to customize **Receptionist/Greeting** to include more details about whom the caller has called, such as your name and/or the telephone number that they have reached. Also note that the dialplan gives "**REFUSE**" callers a 1 in 4 chance of hearing **Receptionist/StopCalling**; this soundbyte is intended to discourage persistant callers from calling again.

Playback()	***soundclip says***
Receptionist/GoodMorning	*"Good morning"*
Receptionist/GoodAfternoon	*"Good afternoon"*
Receptionist/GoodEvening	*"Good evening"*
Receptionist/Greeting	*"You have reached the receptionist at my home."*
Receptionist/Not-accepting-calls	*"Regrettably, we cannot accept your call at this time."*
Receptionist/StopCalling	*"Please add this telephone number to your DO NOT CALL list"*
Receptionist/Ringthrough	*"One moment, please, while I ring you through."*
Receptionist/Noone-available	*"My apologies; we cannot take your call at this time."*
Receptionist/Leave-message	*"If you wish, you may leave a message after the tone."*
Receptionist/Goodbye	*"Thank you for calling; goodbye."*
Receptionist/KnownCallers/*	*prerecorded name (if available), keyed by value retrieved from AstDB `CallerId/${CALLERID(num)}` in caller look up*

AstDB

This context looks for values in these AstDB families:

- **CallerHandling/**
- **CallerId/**

- **`Settings/`**

While Asterisk can identify calls by the CallerID they present, *we* recognize that individuals and organizations can use multiple phone numbers, and want the Asterisk dialplan to identify and handle calls based on that identification. This dialplan uses two AstDB families to accomplish this goal:

CallerHandling/*

The "**`CallerHandling`**" family is keyed by a unique and arbitrary index. Each entry represents a single caller already known to us, and contains a string that represents a *TREATMENT*, or how we want Asterisk to handle this caller. These entries each carry one of 6 possible values:

Value	**applies to**	**Handling policy**	**Handling schedule(s)**
`"REFUSE"`	CallerIDs that we do not want to have anything to do with		
`"VMONLY"`	CallerIDs that we do not want to talk directly with		
`"MASTER"`	CallerIDs that we talk to anytime, & have special privileges	`Hpolicy`	
`"DIRECT"`	CallerIDs that we want to talk to any time they call	`Dpolicy`	
`"SCREEN"`	CallerIDs that we only want to talk to during "personal" hours	`Ppolicy`	`Pschedule`
`"FILTER"`	CallerIDs that we only want to talk to during "business" hours	`Bpolicy`	`Pschedule` `Bschedule`

Should the dialplan not be able to locate an appropriate **`CallerHandling`** entry for the incoming call, it will use the string "**`CALLER`**", a value which does not occur in the database.

Value	**applies to**	**Handling policy**	**Handling schedule(s)**
`"CALLER"`	CallerIDs not in our database (an "unknown" callers)	`Upolicy`	`Pschedule` `Bschedule`

We also have one special **`CallerHandling`** entry:

`CallerHandling/AUTOREFUSE`	preloaded with the *TREATMENT* we wish to give callers that we autoreject (like blocked callerids and known telemarketers). Normally, this value is "**`REFUSE`**" (to give them the brush-off), but you can change it to any of the 6 predefined values (say "**`VMONLY`**" for voicemail-only, or "**`FILTER`**" for business-hours only).

CallerId/*

The "`CallerId`" family is keyed by `${CALLERID(num)}`, and, for each recognized key, contains the corresponding `CallerHandling` key for each unique known caller. This permits a single, unique known caller to have only 1 "`CallerHandling`" entry, and as many "`CallerId`" entries as unique phone numbers.

This two-part callerID handling was a concious design decision, so that I could implement the back-end that supplies this information as a pair of tables in a MySQL relational database. This back-end database serves as a queriable telephone book, with individual entries for each person and organization that Asterisk handles.

As an example of how the dialplan uses these AstDB entries, we know five callers:

- "*Mary*", family, who has 2 phone numbers (`123-456-7890` and `999-888-7654`), that we take calls from no matter when she calls ,
- "*Joe*", a friend, who has 1 phone number (`123-555-1000`), that we take calls from when we aren't busy ,
- "*ACME INC*", a business, who has 2 phone numbers (`123-567-0001` and `123-567-1001`), that we take calls from during "business hours" ,
- "*Give2Me*", a charity, who has 1 phone number (`1-800-999-9999`), that we do not want to talk to directly, and

- "*Bad Duct Cleaning*", a telemarketer, who has 1 phone number (123-223-4890), that we never want to take a call from.

The AstDB entries for these callers might look like

AstDB Family/Key	***value***
CallerId/1232234890	"CHARITY"
CallerId/1234567890	"Mary"
CallerId/1235551000	"Joe"
CallerId/1235670001	"ACME_INC"
CallerId/1236671001	"ACME_INC"
CallerId/9998887654	"Mary"
CallerId/18009999999	"BLACKLIST"
CallHandling/ACME_INC	"FILTER"
CallHandling/BLACKLIST	"REFUSE"
CallHandling/CHARITY	"VMONLY"
CallHandling/Mary	"DIRECT"
CallHandling/Joe	"SCREEN"

Additionally, we have sound clips in **/var/lib/asterisk/sounds** with greetings for Mary and Joe:

Sound clip	***contains***
Receptionist/KnownCallers/Mary	a soundbyte greeting Mary ("Mary, Mary, quite contrary")
Receptionist/KnownCallers/Joe	a soundbyte greeting Joe ("friend Joe")

When the dialplan receives a call from CallerID 999-888-7654, it will

1. look for an AstDB CallerId/9998887654 entry.,
2. retrieve the value "Mary" from that entry,
3. look for an AstDB CallerHandling/Mary entry,
4. retrieve the value "DIRECT" from that entry ,
5. process the call according to the "DIRECT" rules , and
6. greet the caller by name using audio clip Receptionist/KnownCallers/Mary

When the dialplan receives a call from CallerID 1-800-999-9999, it will

1. look for an AstDB CallerId/18009999999 entry.
2. retrieve the value "BLACKLIST" from that entry,
3. look for an AstDB CallerHandling/BLACKLIST entry,
4. retrieve the value "REFUSE" from that entry ,
5. handle the call according to the "REFUSE" rules , and
6. greet the caller with the generic greeting in Receptionist/Greeting

When the dialplan receives a call from CallerID 555-555-1212, it will

1. look for an AstDB CallerId/5555551212 entry,
2. not finding one, handle the call according to the rules and schedules that it would use for a caller with a "FILTER" setting, and
3. greet the caller with the generic greeting in Receptionist/Greeting

When the dialplan receives a call from a blocked CallerID , it will

1. handle the call according to the rules it would use for a caller with a "REFUSE" setting, and
2. greet the caller with the generic greeting in Receptionist/Greeting

Settings/Hpolicy , Settings/Dpolicy, Settings/Ppolicy , Settings/Bpolicy , Settings/Upolicy

Once we derive a specific *TREATMENT* for an incoming CallerID (through the `CallerId` and `CallHandling` AstDB families), we need to retrieve dialplan-specific settings for that *TREATMENT*. At this time, we use hardcoded settings for "`REFUSE`" and "`VMONLY`" *TREATMENTs*, and look up the settings of the other *TREATMENTs* from the AstDB `Settings` family.

Treatment	***interpreted as***	***we use the values in***
"`MASTER`"	a Householder	`Settings/Hpolicy,`
"`DIRECT`"	a Direct line	`Settings/Dpolicy`
"`SCREEN`"	Personal friends	`Settings/Ppolicy`
"`FILTER`"	Business acquaintences	`Settings/Bpolicy`
"`VMOLNY`"	*voicemail-only callers*	*not applicable, hardcoded as NO_IVR=0, POLICY=0, ACCEPT_MSG=1*
"`REFUSE`"	*blacklisted callers*	*not applicable, hardcoded as NO_IVR=1, POLICY=0, ACCEPT_MSG=0*
anything else	Unknown callers	`Settings/Upolicy`

Each database entry contains three comma-separated numeric values representing three call-handling settings:

`NO_IVR`	dictates whether or not we play the Special Information Tone (SIT)	0 = suppress SIT 1 = play SIT,
`POLICY`	dictates whether we give the caller a brush-off, or pass the call through	0 = brush off 1 = pass through 2 = pass through or forward call
`ACCEPT_MSG`	dictates whether we offer to accept a voicemail or not	0 = don't offer voicemail 1 = offer to accept voicemail

For example:

```
asterisk*CLI> database show Settings Ppolicy
/Settings/Ppolicy                                 : 0,2,1
```

indicates that callers that we designate as "Personal Friends" ("`SCREEN`") under policy `Ppolicy`,

- ***will not*** hear the "Out of Service" Special Information Tone,
- ***will*** have their telephone calls advanced to the handset, or forwarded to the ECFN number, and
- ***will*** have the option of leaving a voicemail, should you not answer the call.

However

```
asterisk*CLI> database show Settings Bpolicy
/Settings/Bpolicy                                 : 0,1,0
```

indicates that callers that we designate as "Business acquaintences" ("`FILTER`") under policy `Bpolicy`,

- ***will not*** hear the "Out of Service" Special Information Tone,
- ***will*** have their telephone calls advanced to the handset, ***but not*** forwarded to the ECFN number, and
- ***will not*** have the option of leaving a voicemail, should you not answer the call.

Finally

```
asterisk*CLI> database show Settings Upolicy
/Settings/Upolicy                                 : 1,0,0
```

indicates that callers that we designate as "Unknown Callers" ("`CALLER`") under policy `Upolicy`,

- ***will*** hear the "Out of Service" Special Information Tone,

- ***will not*** have their telephone calls advanced to the handset or forwarded to the ECFN number, and
- ***will not*** have the option of leaving a voicemail.

Settings/ScheduleSlots

The dialplan uses multiple Settings/Pschedule and Settings/Bschedule entries to hold the Personal and Business call schedules. The Settings/ScheduleSlots entry holds a count of the number of Pschedule and Bschedule entries used for each of those schedules.

For example:

```
asterisk*CLI> database show Settings ScheduleSlots
/Settings/ScheduleSlots                          : 1,2
```

indicates that there is, at most, one "Personal hours" (Pschedule*) schedule entry, and, at most, two "Business hours" (Bschedule*) schedule entries.

Settings/Pschedule*

For those callers that we have established as having a "SCREEN" *TREATMENT*, we want the dialplan to pass their calls through, *except* for certain times of the day (such as lunch or dinner time, after bedtime, during our favourite TV shows, etc.). For these "SCREEN" callers, we check if the current time is within one of our blocked out times, and either ring them through (if *outside* of the blocked out time) or apologise to them (if *inside* of the blocked out time). This process involves looping through successive Settings/Pschedule entries (starting with Pschedule0, and stopping before Pschedule${PSLOTS}), testing them against the current time. If an entry matches, then we apologise to the caller (Receptionist/Not-accepting-calls) and do not pass the call through.

Each Settings/Pschedule* entry contains a comma-separated list of parameters:

ENABLED	*NULL* for no schedule 'A' for schedule present & enabled anything else for schedule present & disabled
STIME	time-of-day that this schedule applies to see **ExecIfTime()** time parameter for format
SDOW	day-of-week that this schedule applies to see **ExecIfTime()** dow parameter for format

For example:

```
asterisk*CLI> database show Settings Pschedule0
/Settings/Pschedule0                         :  A,22:00-08:30,*
```

which indicates that

1. the schedule entry is **ENABLED**, and the dialplan *will* check calls against this schedule,
2. the receptionist will *decline* calls because of Personal hours between 10:00 PM and 8:30 AM
3. this schedule will apply to every day of the week

Settings/Bschedule*

For those callers that we have established as having a "FILTER" *TREATMENT*, or have not established any *TREATMENT* (because we don't know them), we want the dialplan to pass their calls through only *during* certain times of the day (such as Monday-Friday, from 9:30AM to 4PM) . Additionally, these sorts of calls must also pass the same personal-time filtering that "SCREEN" *TREATMENT* affords (i.e. do not pass the call through during lunch). For "FILTER" *TREATMENT* callers, the dialplan first checks for compliance with "personal hours" before checking for compliance with "business hours".

If the call passes the "personal hours" check, the dialplan then determines if the current time is within one of our "business hours" allowed times, and either passes the call along (if it came in during the allowed time) or apologises to the caller (if outside the allowed time). This process involves looping through successive `Settings/Bschedule` entries (starting with `Bschedule0`, and stopping before `Bschedule${BSLOTS}`), testing each against the current time. If a call fails *all* tests, then we decline it (`Receptionist/Not-accepting-calls`), otherwise we pass it through.

Each `Settings/Bschedule*` entry contains a comma-separated list of parameters:

`ENABLED` *NULL* for no schedule
'A' for schedule present & enabled
anything else for schedule present & disabled

`STIME` time-of-day that this schedule applies to
see **`ExecIfTime()`** time parameter for format

`SDOW` day-of-week that this schedule applies to
see **`ExecIfTime()`** dow parameter for format

For example:

```
asterisk*CLI> database show Settings Bschedule0
/Settings/Bschedule0                              :  A,09:30-17:00,mon-fri
```

indicates that the first schedule entry

- is **ENABLED**, and the dialplan *will* check calls against this schedule, and
- the receptionist will *accept* calls between 9:30 AM and 5:00 PM, and
- this schedule will apply only between Monday and Friday.

However

```
asterisk*CLI> database show Settings Bschedule1
/Settings/Bschedule1                              :  D,*,*
```

indicates that the second schedule entry is **DISABLED**, and the dialplan will not check calls against this schedule.

Call Detail Recording

The `[Receptionist]` context updates the CDR `userfield` with values that summarize the dialplan’s handling of the inbound call. Note: Some calls will have multiple value summaries.

Value	***Indicates***
`/CallForward`	*seen on forwarded call when* Receptionist forwarded the call to the ECFN number
`/Accepted`	Receptionist accepted the call and it was picked up *or forwarded*
`/Declined`	Receptionist declined the call because of scheduling reasons *or because no one picked up the call*
`/Refused`	Receptionist declined the call because of policy reasons *(blacklisted via “REFUSE” or “VMONLY”)*
`/Voicemail`	*appended to userfield when* Caller left voicemail