APPENDIX E

Asterisk Command-Line Interface Reference

To access the Asterisk command-line interface (CLI), pass the `-c` or `-r` argument to the Asterisk executable. In other words, type this from your shell prompt:

```
# asterisk -r
```

If you want the system to provide you with more information about what it is doing (an excellent idea, especially when you're new to Asterisk), you can add the argument `-v`, as many times as you'd like:

```
# asterisk -vvvvvvvvr
```

The more vs you include, the more vvvvvvvvvverbose the output will be.

The CLI allows you to interact with a running Asterisk server, and it will be very useful to you for troubleshooting and monitoring.

Since the CLI employs tabbed name completion, you can press the Tab key to see a list of possible commands. This makes the CLI very easy to use. Let's take a look at the commands.

!

`!`*`command`*

Executes a given shell command. If followed immediately by a carriage return, Asterisk starts an interactive shell. You can return to the Asterisk CLI by executing an `exit` command.

abort halt

`abort halt`

Cancels a requested Asterisk shutdown (betcha never get the chance, though!). This command is only for the very fast-fingered.

add

The `add` command contains many subcommands that allow you to add functionality to your Asterisk PBX without directly editing the configuration files.

When you add a new line to the dialplan your changes immediately become active, but changes made to the dialplan from the command line are not permanent until you save them (see `save dialplan`). All comments are stripped from the *extensions.conf* file upon a `save dialplan`. The `add` commands are useful for making temporary changes and for ad hoc testing, but we recommend that permanent changes to the dialplan be made directly to *extensions.conf* in */etc/asterisk/*.

add extension

`add extension` *`exten,priority,app,app_data`* `into` *`context`* `[replace]`

Adds a new extension into the specified context. If an extension with the same priority exists, and the optional `replace` argument is given, replaces the existing extension.

```
add extension 500,1,Dial,IAX2/guest@misery.digium.com/s@default into local
```

add ignorepat

`add ignorepat` *`pattern`* `into` *`context`*

Adds a new ignore pattern into the specified context.

```
add ignorepat 9 into local
```

add queue member

`add queue member` *`channel`* `to` *`queue`* `[penalty` *`penalty`*`]`

Allows you to add a channel to a specified queue, optionally specifying a penalty with the `penalty` option.

```
add queue member SIP/1000-d448 to customer_service penalty 10
```

agi

When you're running an AGI program, you can turn debugging on and off with the use of `agi debug` and `agi no debug`, respectively.

agi debug

```
agi debug
```

Turns on AGI debugging.

agi no debug

```
agi no debug
```

Turns off AGI debugging.

answer

```
answer
```

Answers an incoming call on the CONSOLE (OSS) channel. The OSS channel must be configured in *oss.conf* before the `answer` command is available.

database

The Asterisk database is a simple implementation based on Version 1 of the Berkeley database. You can add entries to the database, remove entries from the database, and view entries in the database with the following commands.[*]

database del

`database del` *`family key`*

Deletes an entry in the Asterisk database for a given family and key.

```
database del phones 1000/username
```

database deltree

`database deltree` *`family`* `[`*`keytree`*`]`

Deletes a family or a specific keytree within a family in the Asterisk database.

```
database deltree phones
```

[*] For more about the Asterisk database, see Chapter 6.

database get

`database get` *`family key`*

Retrieves an entry in the Asterisk database for a given family and key.

```
database get phones 1000/username
```

database put

`database put` *`family key value`*

Adds or updates an entry in the Asterisk database for a given family, key, and value.

```
database put phones 1000/username bob
```

database show

`database show [`*`family`*` [`*`key`*`]]`

Shows contents of database, or specific families, keys, and values.

```
database show phones
```

debug channel

`debug channel` *`channel_name`*

Allows a debug of a specific active channel. See also `show channels`.

```
debug channel SIP/1000-e54f
```

dial

`dial [`*`extension`*`[@`*`context`*`]]`

Dials a given extension (optionally, in the context specified) through the CONSOLE channel. This command is available only if *chan_oss.so* or *chan_alsa.so* is loaded in the *modules.conf* file.

```
dial 1000@phones
```

dont include

`dont include` *`context_to_be_removed`* `in` *`context`*

Removes a specified include from a context.

```
dont include local-extensions in incoming
```

dump agihtml

```
dump agihtml filename
```

Dumps a list of AGI commands in HTML format to the given filename. The file will be saved to the */tmp/* directory by default, but a full path may be specified.

exit

```
exit
```

Closes the command-line interface, if you connected to the Asterisk console via the `–r` flag. You cannot use the `quit` and `exit` commands to shut down the PBX (as would be the case if the Asterisk were running in the foreground). To shut down the PBX rather than exiting the console, see the `stop` and `restart` commands.

extensions reload

```
extensions reload
```

Reloads the dialplan configuration from the *extensions.conf* file. In other words, it reloads only your dialplan; nothing else. This command is safe to run even when calls are active. Any new channels being created will be based on the newly reloaded dialplan.

hangup

```
hangup
```

Hangs up any currently active calls placed using the CONSOLE channel. This command is only available if *chan_oss.so* or *chan_alsa.so* is loaded in the *modules.conf* file.

help

```
help [command [subcommand]...]
```

Displays help for commands and command-line usage. A single question mark or tab will do the same.

```
    help show applications
```

iax2

Subsets of this command allow you to manage your IAX connections.

iax2 debug

```
iax2 debug
```

Enables IAX debugging.

iax2 no debug

```
iax2 no debug
```

Disables IAX debugging.

iax2 provision

```
iax2 provision host template [forced]
```

Used to configure an IAX device, such as Digium's IAXy. Provisions the given peer or IP address using a template matching either `template` or *‘ (if the template is not found). Templates are configured in the *iaxprov.conf* file, usually located in */etc/asterisk/*. If `forced` is specified, empty provisioning fields will be provisioned as empty fields.

```
iax2 provision 192.168.1.100 default
```

iax2 show cache

```
iax2 show cache
```

Displays currently cached IAX dialplan results. Related to the `switch =>` statement for remote dialplans. Remote dialplans are cached for a period of time (600 seconds); they then expire and must be requeried if used again.

iax2 show channels

```
iax2 show channels
```

Displays detailed information about active IAX channels.

iax2 show firmware

```
iax2 show firmware
```

Shows available IAX firmware.

iax2 show peer

```
iax2 show peer peer_name
```

Shows details on a specific IAX peer.

```
iax2 show peer iaxfwd
```

iax2 show peers

`iax2 show peers [registered] [like *pattern*]`

Lists all known IAX2 peers. The optional `registered` argument causes only peers with known addresses to be listed. The optional regular expression pattern is used to filter the peer list.

```
iax2 show peers registered like iax*
```

iax2 show provisioning

`iax2 show provisioning [*template*]`

Lists all known IAX provisioning templates, or the details of a specific template.

iax2 show registry

`iax2 show registry`

Lists details and status of all registration requests.

iax2 show stats

`iax2 show stats`

Displays statistics for the IAX channel driver.

iax2 show users

`iax2 show users [like *pattern*]`

Lists all known IAX2 users. The optional regular expression pattern is used to filter the user list.

```
iax2 show users like iax*
```

iax2 trunk debug

`iax2 trunk debug`

Requests the current status of IAX trunking. Trunking is enabled for a peer with `trunk=yes` in *iax.conf*.

include context

`include *context* in *context*`

Includes the specified context in another context.

```
include local-users in incoming
```

indication

The loadzone option in a channel configuration file configures the tone zone to use for a channel. A *tone zone* is a set of indications, as configured in *indications.conf*, that contains information about all the various sounds that are common to telephones in a particular country—dial tone, ringing cycles, busy tones, and so on. A loaded tone zone is applied to a Zaptel channel, which will behave according to the definition for its tone zone. The idea is to deliver familiar telephone sounds, wherever in the world the users might be. Individual channels can have different indication sets configured, which means that a single Asterisk system can provide familiar telephony behavior to people from different countries. The defaultzone is used if nothing is specified for the channel.

indication add

indication add *country indication* "*tonelist*"

Adds the given indication to the country. See also show indications.

```
indication add us dial "350+440"
```

indication remove

indication remove *country indication*

Removes the given indication from the country. See also show indications.

```
indication remove us dial
```

init keys

init keys

Initializes private RSA keys using the passcode specified by the user. Keys are generated with the use of the *astgenkey* script. Keys generated with the use of a passcode must be initialized with the –i flag when starting Asterisk, or with the init keys command from the CLI.

load

load *module_name*

Loads the specified module into Asterisk.

```
load chan_oss.so
```

local show channels

```
local show channels
```

Shows the status of Local channels.

logger

In the *logger.conf* file, you can specify the various levels of detail the system will record in its logs. The following commands allow you to reload and rotate those files. Logs are typically stored in the */var/log/asterisk/* directory.

logger reload

```
logger reload
```

Reloads the log files. Required after making a change to the *logger.conf* configuration file.

logger rotate

```
logger rotate
```

Rotates and reopens the log files. When rotating, the old file is renamed to include a `.n`, where `n` is the highest numbered *logfile.n* + 1. If *logfile.n* does not exist, the file is renamed to *logfile.0*.

meetme

The `meetme` command can be used for a variety of purposes, including listing all active conferences, the number of parties in a conference, the number of marked users, the active length of a conference, and whether a conference was created dynamically or statically.

> A timing interface must be loaded in order for this command to be available.

The following `meetme` subcommands can be used from the console to control active conferences.

meetme kick

```
meetme kick confno [user_number | all]
```

Kicks (i.e., removes) one or all participants from an active conference.

```
meetme kick 100 all
```

meetme list

```
meetme list confno
```

Lists the associated channel names of conference participants and monitors status.

```
meetme list 100
```

meetme lock

```
meetme lock confno
```

Locks a conference from allowing any joins.

```
meetme lock 100
```

> As the number of users in a conference grows, so does the load on the CPU, as it has to mix all of the incoming streams into one, and then transmit the result back out to all the participants. If you have advertised a public conference and it suddenly becomes too popular, you may want to lock out any further participants in order to preserve sound quality.

meetme mute

```
meetme mute confno user_number
```

Mutes a user in the conference.

```
meetme mute 100 1
```

meetme unlock

```
meetme unlock confno
```

Unlocks a conference, allowing channels to join the active conference.

```
meetme unlock 100
```

meetme unmute

`meetme unmute` *`confno user_number`*

Unmutes a user in the conference who is muted.

```
meetme unmute 100 1
```

pri

If you are running the ISDN-PRI protocol on any of your T1 spans, the following commands will help you with troubleshooting.

pri debug

`pri debug`

Turns on PRI debugging.

pri intense debug span

`pri intense debug span` *`span`*

Enables very verbose debugging information for the D-channel of your PRI. This information is invaluable when troubleshooting PRI connections to non-Asterisk systems (such as the PSTN).

```
pri intense debug span 1
```

pri no debug

`pri no debug`

Turns off PRI debugging.

pri show debug

`pri show debug [`*`span`*`]`

Displays the status of PRI debugging and intense debugging for all spans or, optionally, a single defined span.

pri show span

```
pri show span span
```

Displays extended information about a PRI span.

```
pri show span 1
```

quit

See `exit`.

reload

```
reload [module ...]
```

Reloads configuration files for all listed modules that support reloading (or for all supported modules, if none are specified).

```
reload res_crypto.so
```

remove

The `remove` command contains many subcommands that allow you to remove functionality from your Asterisk PBX without directly editing the configuration files.

This function can be used for ad hoc testing, but if you want to make the changes permanent, it is recommended that you edit the various configuration files directly, from */etc/asterisk/*.

remove extension

```
remove extension exten@context [priority]
```

Removes a whole extension from a context. If the priority is specified, removes that priority only within the given extension. Subsequent priorities within the extension will be renumbered if you use the `n` priority-naming scheme.[*]

```
remove extension 500@default 3
```

[*] If you have explicitly numbered your priorities, you will create a gap in your extension. This can easily be corrected by adding a `NoOp( )` command in the removed priority (e.g., `add extension 500,3,Noop into default`).

remove ignorepat

```
remove ignorepat pattern from context
```

Removes the ignore pattern from the given context.

```
remove ignorepat 9 from local
```

remove queue member

```
remove queue member channel from queue
```

Drops the active channel from the given queue. Queue members are the active channels within a queue.

```
remove queue member SIP/1000-d448 from customer_service
```

restart

When a restart is performed, all channels are cleared (i.e., hung up) and all modules are reloaded. You can also instruct Asterisk to restart only when there no longer any active channels, thus preventing calls from being dropped.

restart gracefully

```
restart gracefully
```

Causes Asterisk to stop accepting new calls and perform a cold restart when all active calls have ended.

restart now

```
restart now
```

Causes Asterisk to immediately hang up all calls and perform a cold restart.

restart when convenient

```
restart when convenient
```

Causes Asterisk to perform a cold restart when all active calls have ended. New calls are accepted, and only when all calls have completed is the restart performed. Use this command very carefully, as you have no way of knowing when the conditions for the restart will be met. On a busy system, the restart might not occur until well after you've forgotten you requested it. The best practice on a busy system is to execute restarts manually.

save dialplan

```
save dialplan
```

Saves the current dialplan from the command line to the *extensions.conf* file. It is important to remember that all comments are stripped from the dialplan upon saving. It is recommended that permanent changes to the dialplan be made directly in the *extensions.conf* file and then reloaded (see `extensions reload`) to preserve comments.

set

The `set` command is used to control the amount of debugging information on the console. If connecting to a remote Asterisk console, be aware that changes made to the level of debugging have global scope—that is, they affect all consoles. Also be sure to lower the debugging level before exiting if you are logging to a text file (see `logger`).

set debug

set debug *level*

Sets the level of core debug messages to be displayed. `0` means no messages are displayed. Equivalent to `-d[d[d...]]` on startup.

```
set debug 10
```

set verbose

set verbose *level*

Sets the verbosity level on the console. A setting of `0` means that no information on calling activity will be displayed. If you request `10`, you'll be seeing a lot of activity indeed (especially on a busy system). This command has the exact same effect as the `-v[v[v...]]` flags you provide on startup.

```
set verbose 10
```

show

The `show` subcommands are used to display all kinds of information about your system.

show agents

```
show agents
```

Provides summary information about agents configured in *agents.conf*.

show agi

```
show agi [topic]
```

Displays usage information on the given command, when called with a topic as an argument. If called without a topic, provides a list of AGI commands.

```
show agi channel status
```

show application

```
show application application [application [application [...]]]
```

Displays extended information about one (or, optionally, more than one) given application.

```
show application dial
```

show applications

```
show applications
```

Lists brief explanations of all currently available applications.

show channel

```
show channel channel
```

Displays extended information about the given channel.

```
show channel SIP/1000-3d43
```

show channels

```
show channels [concise]
```

Lists the currently defined channels and some information about them. If `concise` is specified, the format is abridged and presented in a more easily machine-parsable format.

show dialplan

```
show dialplan [context]
```

Shows the current state of the dialplan as loaded into memory. If a context name is appended to the end of the command, only that context will be shown. The `show dialplan` command is useful for verifying the order of pattern matching as well.

```
show dialplan incoming
```

> If you type `show dialplan` and then press the Tab key a few times, you'll be presented with a list of all the contexts in your dialplan. On the Asterisk CLI, the Tab key can yield all kinds of neat information. If in doubt, press Tab.

show indications

```
show indications [country [...]]
```

Displays a condensed list of countries, or optionally a detailed list of indications for one or more countries. See also `indications add` and `indications remove`.

```
show indications us
```

show keys

```
show keys
```

Lists the encryption keys on your system. Keys are stored in */var/lib/asterisk/keys/* and are loaded with the *res_crypto.so* module.

show manager command

```
show manager command command
```

Shows extended information about a Manager command. See also `show manager commands`.

```
show manager command setvar
```

show manager commands

```
show manager commands
```

Lists all available Manager commands and their privilege levels, and gives a brief synopsis of each.

show manager connected

```
show manager connected
```

Lists all currently connected Manager agents. Manager agents are configured in *manager.conf*.

show modules

```
show modules
```

Lists currently loaded modules, gives a brief description of each, and shows the module use count.

show parkedcalls

```
show parkedcalls
```

Lists currently parked calls.

show queue

`show queue` *`queue`*

Provides extended information about a particular queue.

```
    show queue customer_service
```

show queues

```
show queues
```

Provides extended information about all queues.

show translation

```
show translation
```

Displays a table of all codecs and their relative translation times between formats (provided in milliseconds). The higher the number, the more work is required to transcode between those formats. If the formats are native (i.e., the same), no transcoding is required—Asterisk simply routes the packets, which requires very little processing time.

show uptime

```
show uptime
```

Displays Asterisk's total uptime and the time since the last reload.

show version

```
show version
```

Displays the currently installed version of Asterisk. The version is controlled through the *.version* file in the Asterisk sources. When updating the Asterisk source code, be sure to perform a `make update` to update this value. The correct version is required when submitting a bug report to the bug tracker (located at *http://bugs.digium.com*—be sure to read the bug submission guidelines *before* submitting bugs!).

show voicemail users

```
show voicemail users [for vm_context]
```

Displays the voicemail context, mailbox number, voicemail zone, and number of new messages for all voicemail users configured in *voicemail.conf*. Optionally, displays information for a specific voicemail context.

```
show voicemail users for default
```

show voicemail zones

```
show voicemail zones
```

Displays the currently configured voicemail zones and their associated time zones and message formats.

sip

The subsets of the `sip` command allow you to manage your SIP connections.

sip debug

```
sip debug
```

Turns on SIP debugging. This will be very verbose.

sip debug ip

```
sip [no] debug ip dotted_ip_notation
```

Debugs (or disables debugging of) SIP messages from a specific IP address. This is useful when trying to debug messages coming from a peer who is not yet registered with you or is not configured in *sip.conf*.

```
sip debug ip 192.168.1.100
```

sip debug peer

```
sip [no] debug peer peer_name
```

Debugs (or disables debugging of) SIP messages from an individual peer, referenced by the peer name configured in *sip.conf*. Debugging information can be displayed for a dynamic host only if that host is registered with you. If you are trying to debug a registration issue, see `sip debug ip`.

```
sip debug peer john
```

sip history

```
sip [no] history
```

Enables or disables SIP history recording. See also `sip show history`.

sip no debug

```
sip no debug
```

Turns off SIP debugging.

sip reload

```
sip reload
```

Reloads the SIP channel module. This is the equivalent of performing a `reload chan_sip.so`. Reloading the SIP channel is required to load changes to *sip.conf* and *sip_notify.conf* into memory. Active SIP channels are not dropped during a `sip reload`.

sip show channel

```
sip show channel channel
```

Displays extended information about an active SIP channel. See also `sip show channels`.

```
sip show channel 00036bdd-39
```

sip show channels

```
sip show channels
```

Displays a list of all active SIP channels. The value in the Call ID column is used by the `sip show channel` command to display extended information about an individual channel. See also `sip show channel`.

sip show history

sip show history *channel*

Provides a detailed log history for a given SIP channel.

```
sip show history 00036bdd-39
```

sip show peer

sip show peer *peer_name*

Displays detailed information about a peer configured in *sip.conf*.

```
sip show peer john
```

sip show peers

sip show peers

Lists and displays the status of all SIP peers.

sip show registry

sip show registry

Lists and displays the status of all peers with whom you are registered.

sip show user

sip show user *user_name*

Displays detailed information about a user in *sip.conf*.

```
sip show user 1000
```

sip show users

sip show users

Displays a listing of all users configured in *sip.conf*.

soft hangup

soft hangup *channel*

Requests a hangup on a given channel.

```
soft hangup SIP/1000-4248
```

stop

Asterisk has various ways of controlling how and when it stops the system. The options are similar to the `restart` commands. You can instruct Asterisk to stop only when there no longer any active channels, thus preventing calls from being dropped.

stop gracefully

```
stop gracefully
```

Stops the system when all currently active calls have completed, and does not accept new calls.

stop now

```
stop now
```

Stops immediately, terminating all active calls.

stop when convenient

```
stop when convenient
```

Stops the system when all currently active calls have completed. New calls are accepted, and the system will stop only when there are no longer any active calls. Using this command is not a good idea, since you have no real way of knowing when the necessary condition for stopping the system will occur.

unload

unload [-f | -h] *module_name*

Unloads the specified module from Asterisk. The `–f` option causes the module to be unloaded even if it is in use (which may cause a crash), and the `-h` option causes the module to be unloaded even if the module says it cannot be, which will almost always cause a crash.

```
    unload app_math.so
```

zap

The Zaptel interfaces allow Asterisk to interact via a physical medium, either analog or digital. This may include telephones, analog PSTN connections, or digital circuits such as T-1/E-1 circuits.

zap destroy channel

```
zap destroy channel channel_number
```

Immediately removes a channel, whether or not it is in use.

```
zap destroy channel 1
```

zap show cadences

```
zap show cadences
```

Displays the configuration of the various ring cadences (ring tones) Asterisk has configured for an analog circuit (FXS).

zap show channel

```
zap show channel channel_number
```

Displays extended information about a particular Zaptel channel.

```
zap show channel 1
```

zap show channels

```
zap show channels
```

Lists all Zaptel channels and their associated extensions, languages, and default Music on Hold classes.