## What is Asterisk?

Asterisk is an open source framework for building communications applications. Asterisk turns an ordinary computer into a communications server. Asterisk powers IP PBX systems, VoIP gateways, conference servers and more.

It is used by small businesses, large businesses, call centers, carriers and governments worldwide. Asterisk is free and open source. Asterisk is sponsored by Digium, the Asterisk Company.

Asterisk is "under the hood" in countless voice communications applications and is capable of interfacing with many traditional Telcom protocols, VoIP protocols, and codecs.

Asterisk provides a staggering list of capabilities and features including: IVR ACD Audio and Video Conferencing Voicemail Call Recording Fax termination CDR

## File Structure

The table below contains the default installation paths for Asterisk component files and libraries. This is not an exhaustive list, only the core components relative to this Quick Start Guide are listed:

**Table 1 Default Installation Paths**

| Path | Description |
|---|---|
| /etc/asterisk | Configuration files |
| /usr/sbin | Location of binary executable |
| /var/log/asterisk | message(error) logs and CDR |
| /usr/lib/asterisk/modules | Component module libraries |

**Default Ports**

| Protocol | Port number | Transport |
|---|---|---|
| SIP | 5060/5061 | TCP/UDP |
| IAX2 | 4569 | UDP |
| MGCP | 2727 | UDP |
| SCCP | 2000 | TCP |
| RTP | 10,00 – 20,000 | UDP |
| Manager | 5038 | TCP |
| H323 | 1720 | TCP |
| Dundi | 4520 | UDP |
| Unistim | 5000 | UDP |

## Requirements

Asterisk can run on multiple base architectures including embedded systems and there are no strict requirements on CPU speed or memory size. This document assumes the use of a standard x86 based processor.

Asterisk can run on a number of Operating Systems. Linux is the only officially supported OS, and it is recommended to use a 2.6.25 or higher kernel (although Asterisk will run on 2.4 kernels). A current and supported release of distributions such as CentOS or Debian is recommended.

An Internet connection is also required.

**Dependencies**

There are a number of packages that are required to be pre-installed on the host server to ensure that Asterisk will compile successfully.

Package like: build-essential, wget, libssl-dev, libncurses5-dev, libnewt-dev, libxml2-dev, linux-headers, libsqlite3-dev, uuid-dev, libjansson-dev, mc, vim

```
sudo apt-get update -y
sudo apt-get install -y <package_name>
```

**Installing Asterisk**

1. Log in to your Linux machine as the 'root' user (superuser). If you are using Ubuntu Linux log in as normal and prefix each command with 'sudo.

   ```
   #sudo -s
   ```

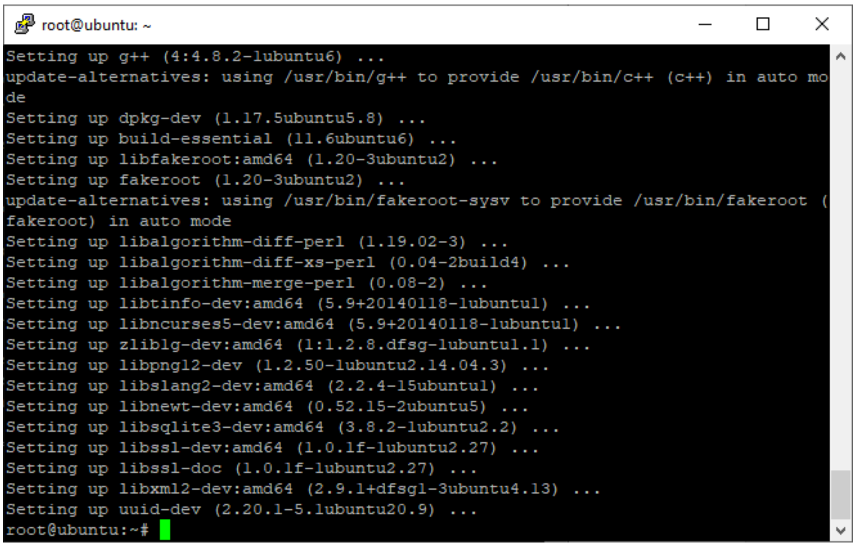
2. Open a Terminal Window or connect through SSH. Download the 'current' Asterisk source tarball to the host machine. This will download the latest (minor) version:

   ```
   #wget https://downloads.asterisk.org/pub/telephony/asterisk/asterisk-13-current.tar.gz
   ```


3. Install Required Packages for asterisk

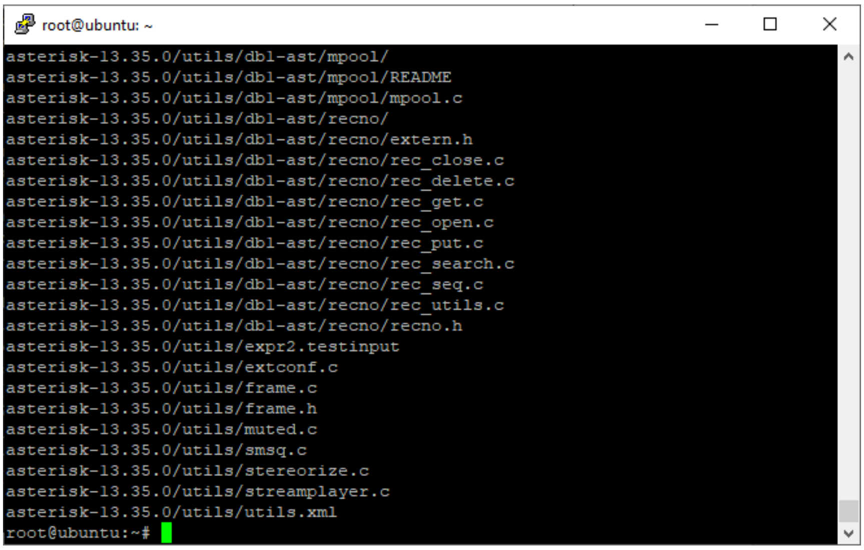
   ```
   #apt-get install build-essential wget libssl-dev libncurses5-dev libnewt-dev libxml2-dev linux-headers-$(uname -r) libsqlite3-dev uuid-dev

   sudo apt-get update -y
   sudo apt-get install -y libjansson-dev
   ```

   -$(uname -r) :- system will automatically replace the exact version number



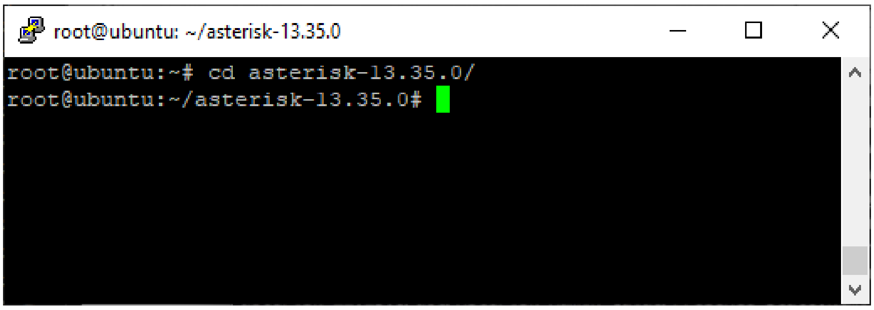

4. Unzip and extract all of the contained source files:

```
#tar zxvf asterisk-13-current.tar.gz
```




5. Enter the newly created source directory

```
# cd asterisk-13.35.0/
```



6. (Optional) and execute the '**install_prereq**' in the contrib/scripts subdirectory. This will not only install the required dependencies but also install all packages necessary to build all option Asterisk components.

   # ./contrib/scripts/install_prereq

7. Read "**Read Me"** file (below)

   **Read Me**

**# The Asterisk(R) Open Source PBX**
```text
      By Mark Spencer <markster@digium.com> and the Asterisk.org developer community.
      Copyright (C) 2001-2019 Digium, Inc. and other copyright holders.
```
**## SECURITY**

It is imperative that you read and fully understand the contents of the security information document before you attempt to configure and run an Asterisk server.

See [Important Security Considerations] for more information.

**## WHAT IS ASTERISK?**

Asterisk is an Open Source PBX and telephony toolkit. It is, in a sense, middleware between Internet and telephony channels on the bottom, and Internet and telephony applications at the top. However, Asterisk supports more telephony interfaces than just Internet telephony. Asterisk also has a vast amount of support for traditional PSTN telephony, as well.

For more information on the project itself, please visit the Asterisk [home page] and the official [wiki]. In addition, you'll find lots of information compiled by the Asterisk community at [voip-info.org].

There is a book on Asterisk published by O'Reilly under the Creative Commons License. It is available in book stores as well as in a downloadable version on the [asteriskdocs.org] web site.

**## SUPPORTED OPERATING SYSTEMS**

### Linux

The Asterisk Open Source PBX is developed and tested primarily on the GNU/Linux operating system, and is supported on every major GNU/Linux distribution.

### Others

Asterisk has also been 'ported' and reportedly runs properly on other operating systems as well, including Sun Solaris, Apple's Mac OS X, Cygwin, and the BSD variants.

**## GETTING STARTED**

First, be sure you've got supported hardware (but note that you don't need ANY special hardware, not even a sound card) to install and run Asterisk.

Supported telephony hardware includes:

* All Analog and Digital Interface cards from [Digium]
* QuickNet Internet PhoneJack and LineJack (http://www.quicknet.net)
* any full duplex sound card supported by ALSA, OSS, or PortAudio
* any ISDN card supported by mISDN on Linux
* The Xorcom Astribank channel bank
* VoiceTronix OpenLine products

**### UPGRADING FROM AN EARLIER VERSION**

If you are updating from a previous version of Asterisk, make sure you read the [UPGRADE.txt] file in the source directory. There are some files and configuration options that you will have to change, even though we made every effort possible to maintain backwards compatibility.

In order to discover new features to use, please check the configuration examples in the [configs] directory of the source code distribution. For a list of new features in this version of Asterisk, see the [CHANGES] file.

**### NEW INSTALLATIONS**

Ensure that your system contains a compatible compiler and development libraries. Asterisk requires either the GNU Compiler Collection (GCC) version 3.0 or higher, or a compiler that supports the C99 specification and some of the GCC language extensions. In addition, your system needs to have the C library headers available, and the headers and libraries for ncurses.

There are many modules that have additional dependencies. To see what libraries are being looked for, see `./configure --help`, or run `make menuselect` to view the dependencies for specific modules.

On many distributions, these dependencies are installed by packages with names like 'glibc-devel', 'ncurses-devel', 'openssl-devel' and 'zlib-devel' or similar.

So, let's proceed:

1. Read this file.

There are more documents than this one in the [doc] directory. You may also want to check the configuration files that contain examples and reference guides in the [configs] directory.

2. Run `./configure`

Execute the configure script to guess values for system-dependent variables used during compilation.

3. Run `make menuselect` _\[optional]_

This is needed if you want to select the modules that will be compiled and to check dependencies for various optional modules.

4. Run `make`

Assuming the build completes successfully:

5. Run `make install`

If this is your first time working with Asterisk, you may wish to install the sample PBX, with demonstration extensions, etc. If so, run:

6. Run `make samples`

Doing so will overwrite any existing configuration files you have installed.

7. Finally, you can launch Asterisk in the foreground mode (not a daemon) with:
```
   # asterisk -vvvc
```

You'll see a bunch of verbose messages fly by your screen as Asterisk initializes (that's the "very very verbose" mode). When it's ready, if you specified the "c" then you'll get a command line console, that looks like this:
```
        *CLI>
```

You can type "core show help" at any time to get help with the system. For help with a specific command, type "core show help <command>". To start the PBX using your sound card, you can type "console dial" to dial the PBX. Then you can use "console answer", "console hangup", and "console dial" to simulate the actions of a telephone. Remember that if you don't have a full duplex sound card (and Asterisk will tell you somewhere in its verbose messages if you do/don't) then it won't work right (not yet).

"man asterisk" at the Unix/Linux command prompt will give you detailedin formation on how to start and stop Asterisk, as well as all the command line options for starting Asterisk.

Feel free to look over the configuration files in `/etc/asterisk`, where you will find a lot of information about what you can do with Asterisk.

### ABOUT CONFIGURATION FILES

All Asterisk configuration files share a common format. Comments are delimited by ';' (since '#' of course, being a DTMF digit, may occur in many places). A configuration file is divided into sections whose names appear in []'s. Each section typically contains two types of statements, those of the form 'variable = value', and those of the form 'object => parameters'. Internally the use of '=' and '=>' is exactly the same, so they're used only to help make the configuration file easier to understand, and do not affect how it is actually parsed.

Entries of the form 'variable=value' set the value of some parameter in asterisk. For example, in [chan_dahdi.conf], one might specify:
```
      switchtype=national
```

In order to indicate to Asterisk that the switch they are connecting to is of the type "national". In general, the parameter will apply to instantiations which occur below its specification. For example, if the configuration file read:
```
      switchtype = national
      channel => 1-4
      channel => 10-12
      switchtype = dms100
      channel => 25-47
```

The "national" switchtype would be applied to channels one through four and channels 10 through 12, whereas the "dms100" switchtype would apply to channels 25 through 47.

The "object => parameters" instantiates an object with the given parameters. For example, the line "channel => 25-47" creates objects for the channels 25 through 47 of the card, obtaining the settings from the variables specified above.

### SPECIAL NOTE ON TIME

Those using SIP phones should be aware that Asterisk is sensitive to large jumps in time. Manually changing the system time using date(1) (or other similar commands) may cause SIP registrations and other internal processes to fail. If your system cannot keep accurate time by itself use [NTP] to keep the system clock synchronized to "real time". NTP is designed to keep the system clock synchronized by speeding up or slowing down the system clock until it is synchronized to "real time" rather than by jumping the time and causing discontinuities. Most Linux distributions include precompiled versions of NTP. Beware of some time synchronization methods that get the correct real time periodically and then manually set the system clock.

Apparent time changes due to daylight savings time are just that, apparent. The use of daylight savings time in a Linux system is purely a user interface issue and does not affect the operation of the Linux kernel or Asterisk. The system clock on Linux kernels operates on UTC. UTC does not use daylight savings time.

Also note that this issue is separate from the clocking of TDM channels, and is known to at least affect SIP registrations.

### FILE DESCRIPTORS

Depending on the size of your system and your configuration, Asterisk can consume a large number of file descriptors. In UNIX, file descriptors are used for more than just files on disk. File descriptors are also used for handling network communication (e.g. SIP, IAX2, or H.323 calls) and hardware access (e.g. analog and digital trunk hardware). Asterisk accesses many on-disk files for everything from configuration information to voicemail storage.

Most systems limit the number of file descriptors that Asterisk can have open at one time. This can limit the number of simultaneous calls that your system can handle. For example, if the limit is set at 1024 (a common default value) Asterisk can handle approximately 150 SIP calls simultaneously. To change the number of file descriptors follow the instructions for your system below:

#### PAM-BASED LINUX SYSTEM

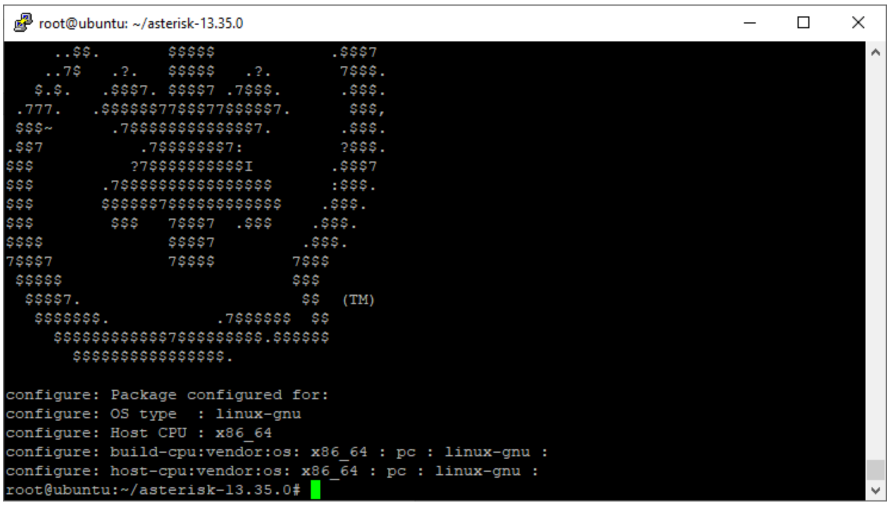
If your system uses PAM (Pluggable Authentication Modules) edit `/etc/security/limits.conf`. Add these lines to the bottom of the file:
```text
root            soft    nofile          4096
root            hard    nofile          8196
asterisk        soft    nofile          4096
asterisk        hard    nofile          8196
```

```

(adjust the numbers to taste). You may need to reboot the system for these changes to take effect.

#### GENERIC UNIX SYSTEM

If there are no instructions specifically adapted to your system above you can try adding the command `ulimit -n 8192` to the script that starts Asterisk.

## MORE INFORMATION

See the [doc] directory for more documentation on various features. Again, please read all the configuration samples that include documentation on the configuration options.

Finally, you may wish to visit the [support] site and join the [mailing list] if you're interested in getting more information.

Welcome to the growing worldwide community of Asterisk users!
```
   Mark Spencer, and the Asterisk.org development community
```
---

Asterisk is a trademark of Digium, Inc.

[home page]: https://www.asterisk.org
[support]: https://www.asterisk.org/support
[wiki]: https://wiki.asterisk.org/
[mailing list]: http://lists.digium.com/mailman/listinfo/asterisk-users
[chan_dahdi.conf]: configs/samples/chan_dahdi.conf.sample
[voip-info.org]: http://www.voip-info.org/wiki-Asterisk
[asteriskdocs.org]: http://www.asteriskdocs.org
[NTP]: http://www.ntp.org/
[Digium]: https://www.digium.com/
[UPGRADE.txt]: UPGRADE.txt
[CHANGES]: CHANGES
[configs]: configs
[doc]: doc
[Important Security Considerations]:
https://wiki.asterisk.org/wiki/display/AST/Important+Security+Considerations
```

8. Configure asterisk

```
# ./configure
```

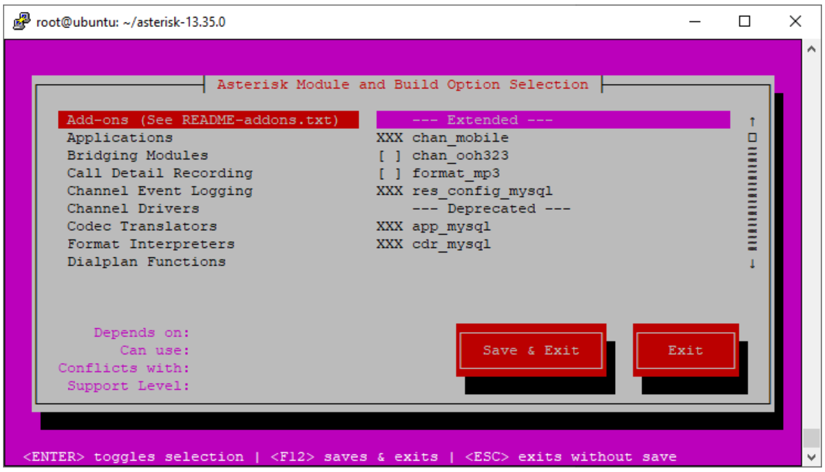
If there is an error, follow the instruction provided by asterisk and install required packages. Otherwise below screen will appear with asterisk logo.




9. Install **Menuselect**

This next step in the build process is to tell Asterisk which modules to compile and install, as well as set various compiler options. These settings are all controlled via a menu-driven system called **Menuselect**. To access the Menuselect system, type:

```
#make menuselect
```




On the left-hand side, you have a list of categories, such as **Applications**, **Channel Drivers**, and **PBX Modules**. On the right-hand side, you'll see a list of modules that correspond with the select category. At the bottom of the screen you'll see two buttons. You can use the **Tab** key to cycle between the various sections, and press the **Enter** key to select or unselect a particular module.

If you see **[*]** next to a module name, it signifies that the module has been selected. If you see ***XXX** next to a module name, it signifies that the select module cannot be built, as one of its dependencies is missing. In that case, you can look at the bottom of the screen for the line labeled **Depends upon**: for a description of the missing dependency.

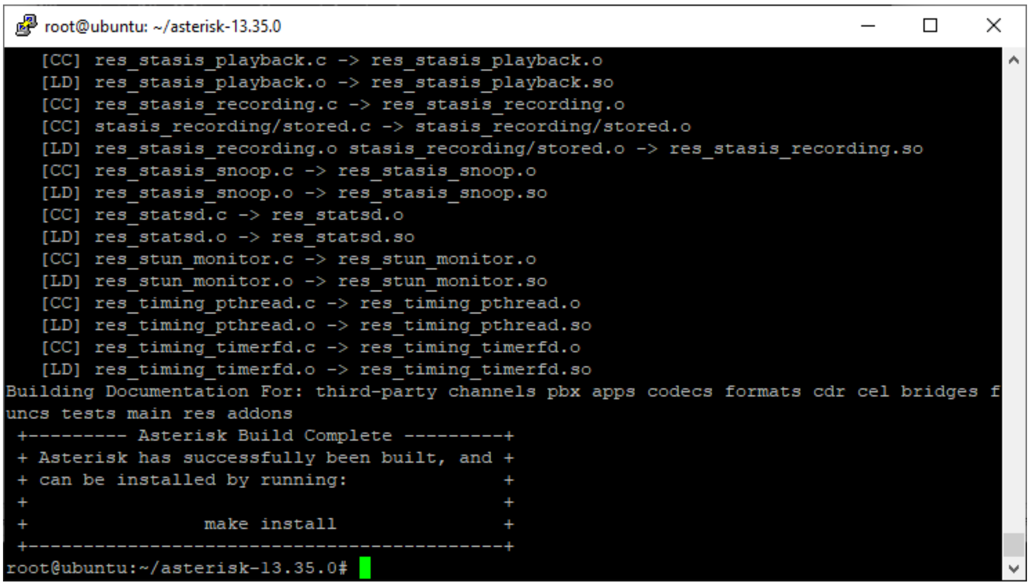
When you're first learning your way around Asterisk on a test system, you'll probably want to stick with the default settings in **Menuselect**. If you're building a production system, however, you may not wish to build all of the various modules, and instead only build the modules that your system is using.

When you are finished selecting the modules and options you'd like in **Menuselect**, press **F12** to save and exit, or highlight the **Save and Exit** button and press enter.

10. Compiling and Installing asterisk – **Make**

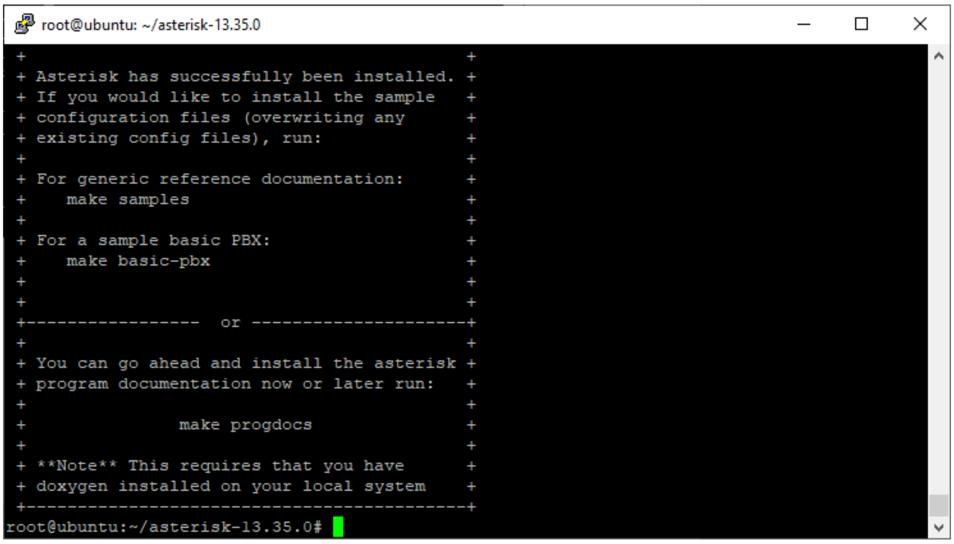
    The compiling step will take several minutes, and you'll see the various file names scroll by as they are being compiled. Once Asterisk has finished compiling, you'll see a message that looks like:

```
# make
```




11. As the message above suggests, our next step is to install the compiled Asterisk program and modules. To do this, use the **make install** command.

```
#make install
```



12. Installing sample files (Creating sample configuration for asterisk start)

    Configuring Asterisk (demo config) The command '**make samples**' created sample configuration files in the default directory **'/etc/asterisk/'**.

    ```
    #make samples
    ```

```
root@ubuntu: ~/asterisk-13.35.0
Installing file configs/samples/skinny.conf.sample
Installing file configs/samples/sla.conf.sample
Installing file configs/samples/smdi.conf.sample
Installing file configs/samples/sorcery.conf.sample
Installing file configs/samples/ss7.timers.sample
Installing file configs/samples/stasis.conf.sample
Installing file configs/samples/statsd.conf.sample
Installing file configs/samples/test_sorcery.conf.sample
Installing file configs/samples/udptl.conf.sample
Installing file configs/samples/unistim.conf.sample
Installing file configs/samples/users.conf.sample
Installing file configs/samples/voicemail.conf.sample
Installing file configs/samples/vpb.conf.sample
Installing file configs/samples/xmpp.conf.sample
Updating asterisk.conf
/usr/bin/install -c -d "/var/spool/asterisk/voicemail/default/1234/INBOX"
build_tools/make_sample_voicemail "//var/lib/asterisk" "//var/spool/asterisk"
Installing file phoneprov/000000000000.cfg
Installing file phoneprov/000000000000-directory.xml
Installing file phoneprov/000000000000-phone.cfg
Installing file phoneprov/polycom_line.xml
Installing file phoneprov/polycom.xml
Installing file phoneprov/snom-mac.xml
root@ubuntu:~/asterisk-13.35.0#
```

```
# cd asterisk/
# ls -la | mc -l
```

Total config file count: - 105 (more than)

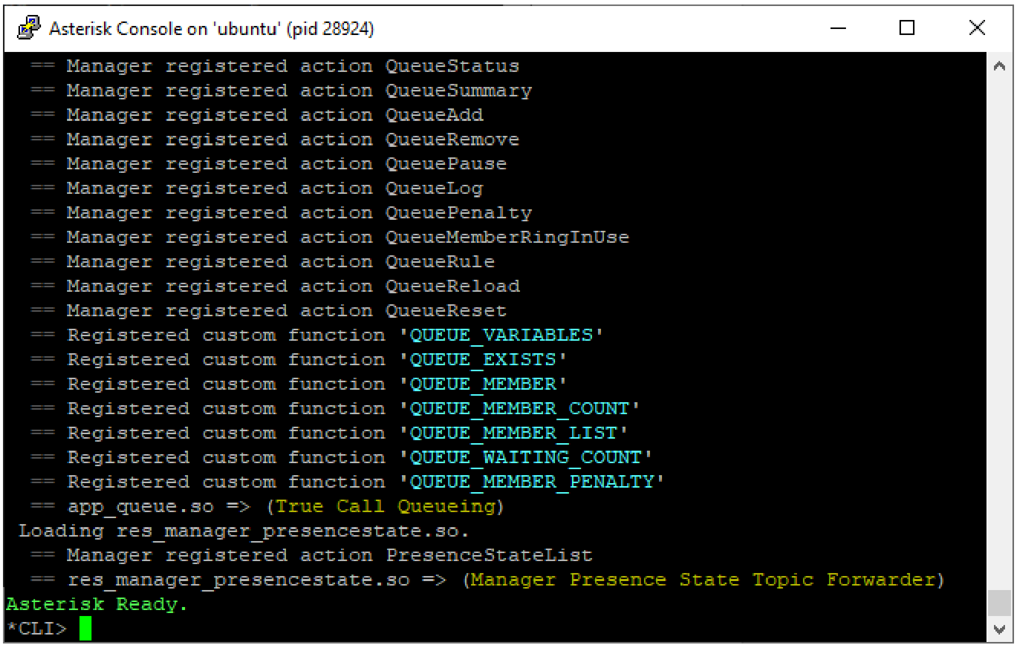
```
#cd /etc/asterisk and list #ls
```

```
root@ubuntu: /etc/asterisk
-rw-r--r--    1 root root  1089 Aug  4 19:32 res_pgsql.conf
-rw-r--r--    1 root root   713 Aug  4 19:32 res_pktccops.conf
-rw-r--r--    1 root root   833 Aug  4 19:32 res_snmp.conf
-rw-r--r--    1 root root  1350 Aug  4 19:32 res_stun_monitor.conf
-rw-r--r--    1 root root  5800 Aug  4 19:32 rtp.conf
-rw-r--r--    1 root root 17039 Aug  4 19:32 say.conf
-rw-r--r--    1 root root 93406 Aug  4 19:32 sip.conf
-rw-r--r--    1 root root   790 Aug  4 19:32 sip_notify.conf
-rw-r--r--    1 root root 10064 Aug  4 19:32 skinny.conf
-rw-r--r--    1 root root  7090 Aug  4 19:32 sla.conf
-rw-r--r--    1 root root  2669 Aug  4 19:32 smdi.conf
-rw-r--r--    1 root root  2501 Aug  4 19:32 sorcery.conf
-rw-r--r--    1 root root  1503 Aug  4 19:32 ss7.timers
-rw-r--r--    1 root root  4969 Aug  4 19:32 stasis.conf
-rw-r--r--    1 root root   370 Aug  4 19:32 statsd.conf
-rw-r--r--    1 root root  1384 Aug  4 19:32 telcordia-1.adsi
-rw-r--r--    1 root root   152 Aug  4 19:32 test_sorcery.conf
-rw-r--r--    1 root root   656 Aug  4 19:32 udptl.conf
-rw-r--r--    1 root root  5850 Aug  4 19:32 unistim.conf
-rw-r--r--    1 root root  2570 Aug  4 19:32 users.conf
-rw-r--r--    1 root root 26402 Aug  4 19:32 voicemail.conf
-rw-r--r--    1 root root  5938 Aug  4 19:32 vpb.conf
-rw-r--r--    1 root root  3734 Aug  4 19:32 xmpp.conf
root@ubuntu:/etc/asterisk#
```

13. Starting Asterisk

```
#asterisk -cvvv
```

Start Asterisk with a control console (-c) and level 3 verbosity (vvv).

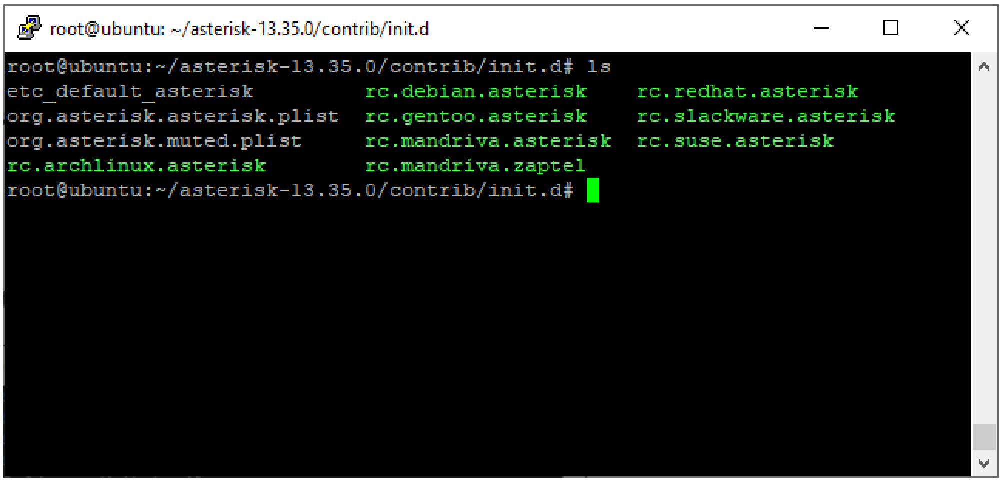



Press **ctrl + c** for previous widow

14. Starting script

There is a special folder called **/asterisk-13.35.0/contrib/init.d in asterisk source code.**

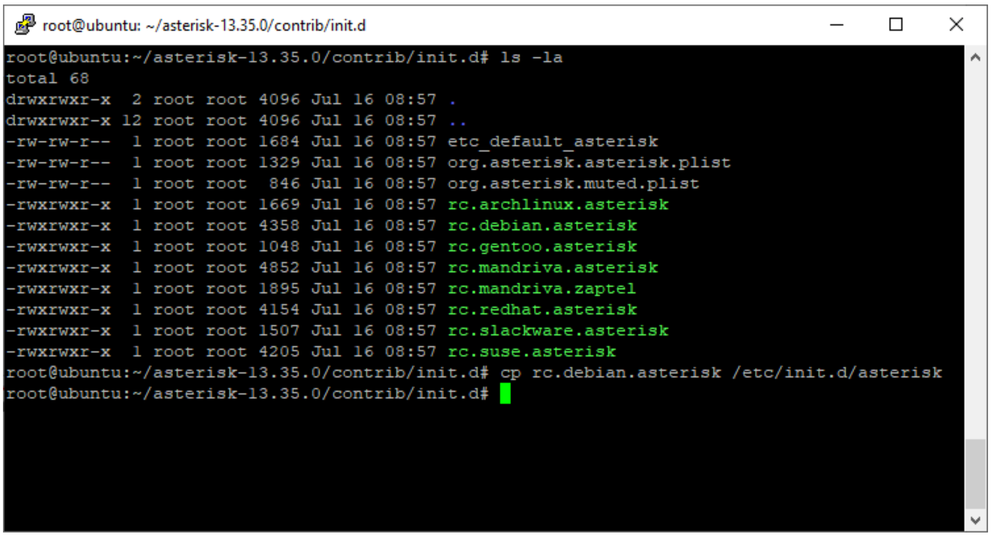
#cd /contrib/init.d




For ubuntu is based on Debian so we can use **rc.debian.asterisk** file (Choose appropriate file based on Operating System)

Copy **rc.debian.asterisk** file to **/etc/init.d/asterisk**

```
# cp rc.debian.asterisk /etc/init.d/asterisk
```

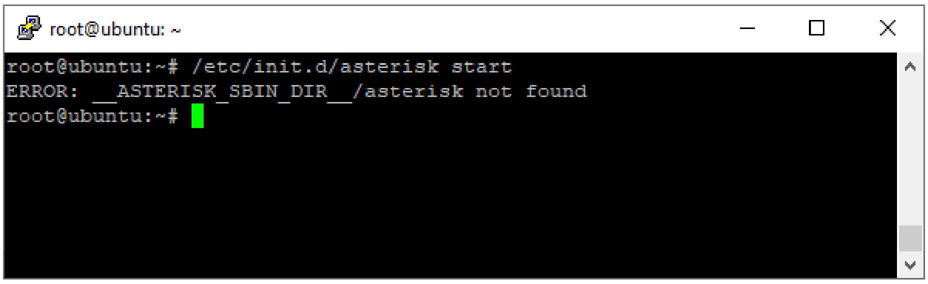



**To set start script in place in case of reboot**

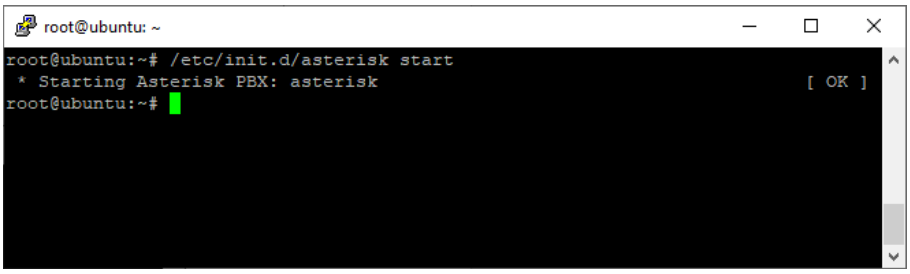
```
# update
# update-rc.d asterisk defaults
```

15. Start asterisk

```
# /etc/init.d/asterisk start
```



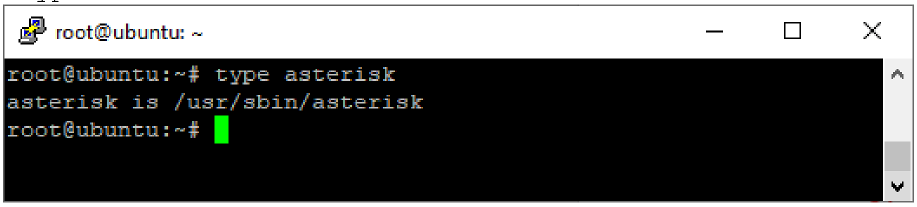

If You got **ERROR: __ASTERISK_SBIN_DIR__/asterisk not found**, we have to configure script to adapt ubuntu. Follow the steps 16 . Otherwise we will get below screen



16. Edit Asterisk script

First find out asterisk binary path using type command

```
#type asterisk
```

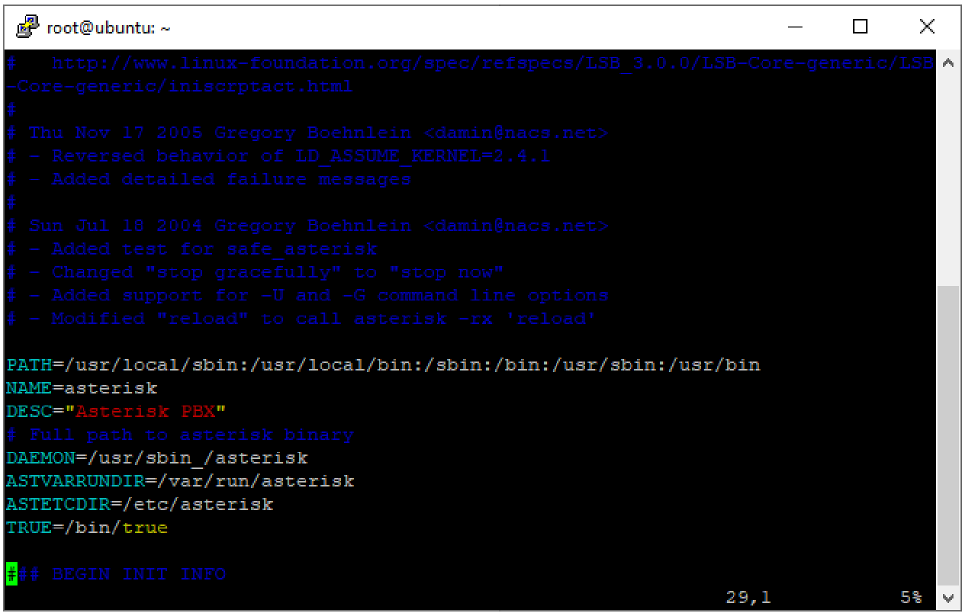



For editing

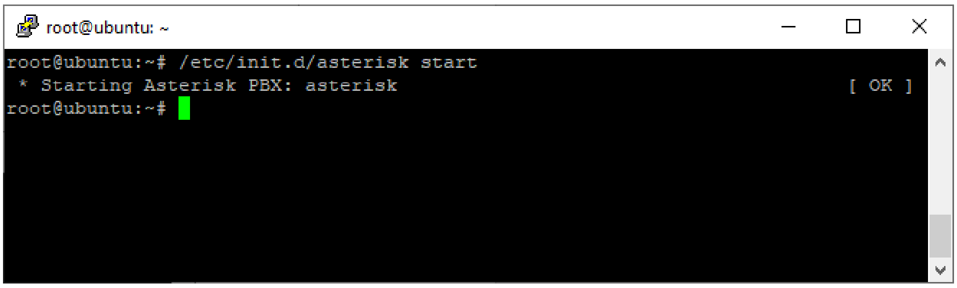
```
# vim /etc/init.d/asterisk
```

__ASTERISK_SBIN_DIR__ : - Location of binary executable
__ASTERISK_VARRUN_DIR__ : - Run Files
__ASTERISK_ETC_DIR__ : - Configuration Files

```
# Full path to asterisk binary
DAEMON=/usr/sbin/asterisk
ASTVARRUNDIR=/var/run/asterisk
ASTETCDIR=/etc/asterisk
TRUE=/bin/true
```

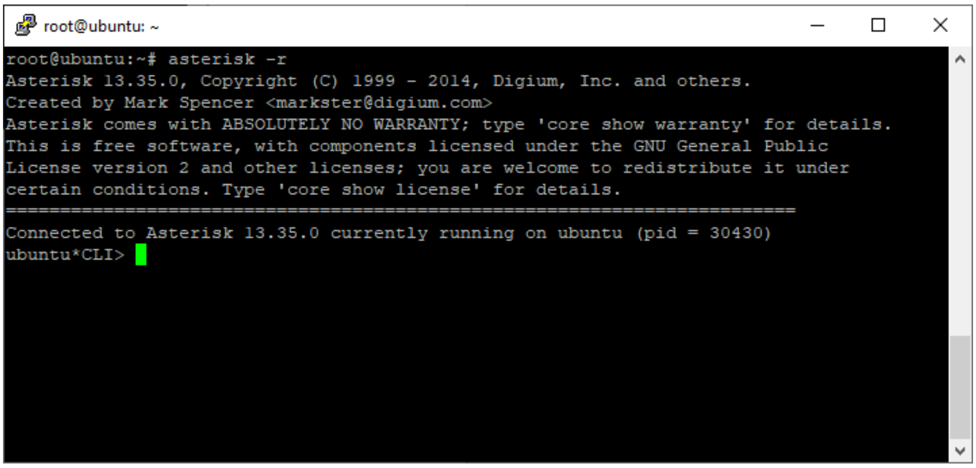



Run Command again `# /etc/init.d/asterisk start`



17. (Optional) to Open Asterisk Console. Type exit for existing asterisk console

```
# asterisk -r
```



### Changing Process User – Reducing Security Issues

To runing process as root is not an good idea,because the process can be accessed everyone. To avoid this problem , we have to change root to other user(accessing process from outside)

### To list all process

```
# ps aux
```

a = show processes for all users
u = display the process's user/owner
x = also show processes not attached to a terminal

```
root@ubuntu: ~
nirmal     2342  0.0  0.3 330112 15328 ?        Sl   17:50   0:00 /usr/lib/x86_64-linux
nirmal     2357  0.0  0.0  11424   776 ?        S    17:50   0:00 /bin/cat
nirmal     2373  0.0  0.5 507580 22540 ?        Sl   17:50   0:00 update-notifier
root       2390  0.0  0.0      0     0 ?        S    17:51   0:00 [kworker/1:2]
nirmal     2396  0.0  3.1 667588 127228 ?       SNl  17:51   0:03 /usr/bin/python3 /usr
nirmal     2423  0.0  0.2 459020  8380 ?        Sl   17:51   0:00 /usr/lib/x86_64-linux
nirmal     2438  0.0  0.7 657652 29932 ?        Sl   17:53   0:01 gnome-terminal
nirmal     2445  0.0  0.0  14836  1748 ?        S    17:53   0:00 gnome-pty-helper
nirmal     2446  0.0  0.1  26880  5444 pts/0    Ss   17:53   0:00 bash
root       2585  0.0  0.1  71252  4120 pts/0    S    17:57   0:00 sudo -s
root       2586  0.0  0.1  26864  5428 pts/0    S+   17:57   0:00 /bin/bash
root       3449  0.0  0.1  61392  5436 ?        Ss   17:58   0:00 /usr/sbin/sshd -D
nirmal     7835  0.0  0.1 124460  5376 ?        Sl   18:45   0:00 /usr/lib/gvfs/gvfsd-m
root      29025  0.1  0.0      0     0 ?        S    19:54   0:13 [kworker/0:1]
root      29198  0.0  0.0      0     0 ?        S    21:39   0:00 [kworker/1:0]
nirmal    29421  0.0  0.6 489124 24360 ?        Ssl  22:17   0:00 /usr/lib/unity/unity-
root      29451  0.0  0.0      0     0 ?        S    22:26   0:00 [kworker/u256:0]
root      30249  0.0  0.1 107252  6632 ?        Ss   23:04   0:00 sshd: nirmal [priv]
root      30266  0.1  0.0      0     0 ?        S    23:04   0:01 [kworker/0:2]
nirmal    30334  0.0  0.1 107252  4108 ?        S    23:04   0:00 sshd: nirmal@pts/2
nirmal    30335  0.0  0.1  26868  5368 pts/2    Ss   23:04   0:00 -bash
root      30355  0.0  0.1  71252  4076 pts/2    S    23:04   0:00 sudo -s
root      30356  0.0  0.1  26884  5588 pts/2    S    23:04   0:00 /bin/bash
root      30386  0.0  0.0  71252  3948 pts/2    S    23:08   0:00 sudo -s
root      30387  0.0  0.1  26884  5444 pts/2    S    23:08   0:00 /bin/bash
root      30410  0.0  0.0      0     0 ?        S    23:10   0:00 [kworker/u256:1]
root      30430  0.4  0.8 1558072 35480 ?       Ssl  23:10   0:02 /usr/sbin/asterisk
root      30501  0.0  0.0      0     0 ?        S    23:18   0:00 [kworker/u256:2]
root      30502  0.0  0.0  22656  2600 pts/2    R+   23:19   0:00 ps aux
root@ubuntu:~#
```

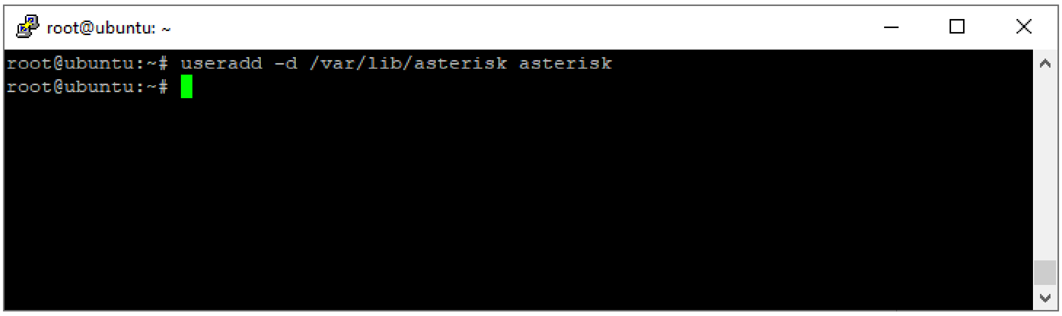
### First stop asterisk server

```
# /etc/init.d/asterisk stop
```

```
root@ubuntu: ~
root@ubuntu:~# /etc/init.d/asterisk stop
 * Stopping Asterisk PBX: asterisk                                    [ OK
root@ubuntu:~#
```

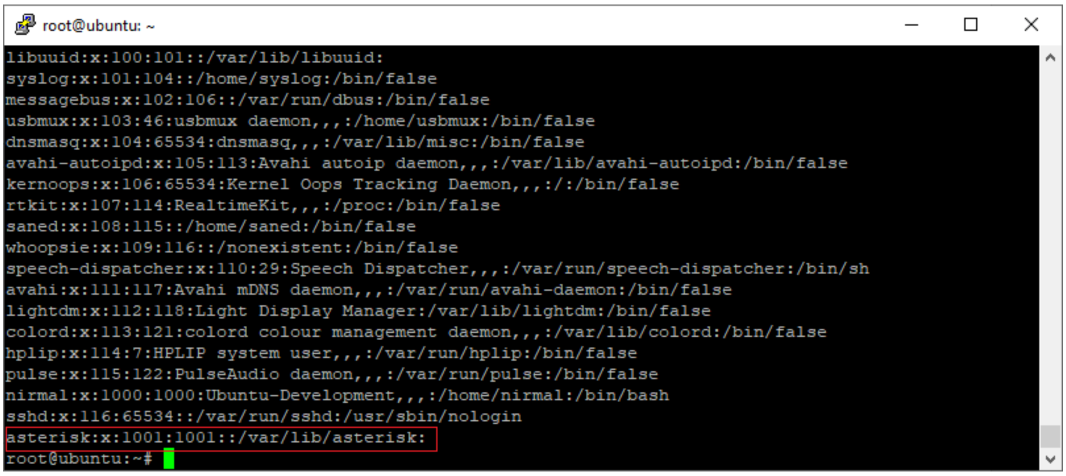
### Creating New User

```
# useradd -d /var/lib/asterisk asterisk
```




### To See password

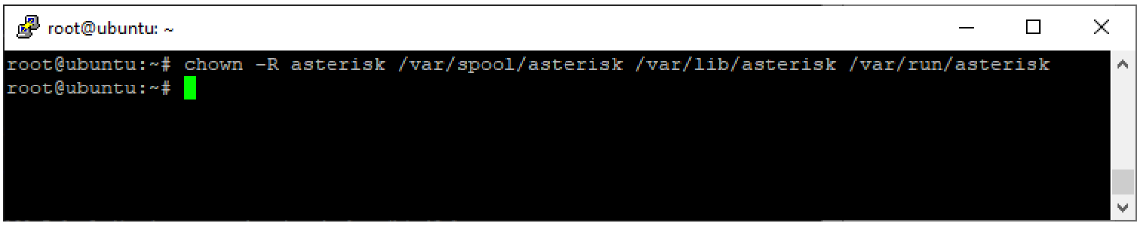
```
# cat /etc/passwd
```




### Access Rights

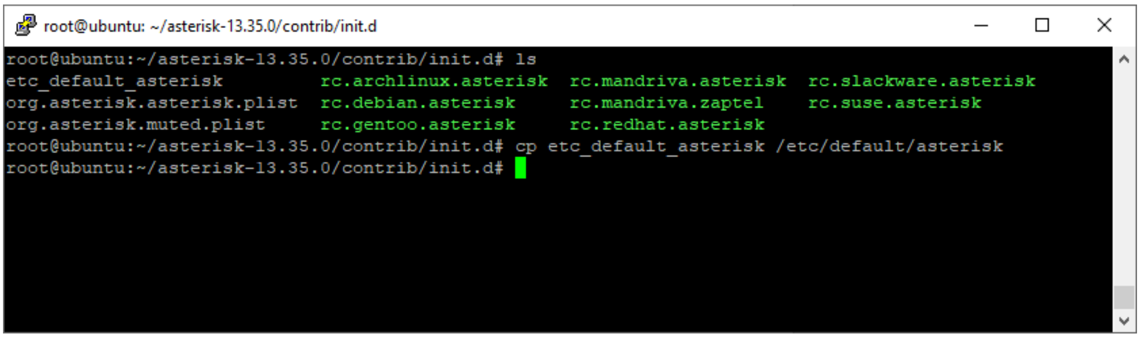
The new user need access right to some files. (Ownership Change)

```
# chown -R asterisk /var/spool/asterisk /var/lib/asterisk /var/run/asterisk
```



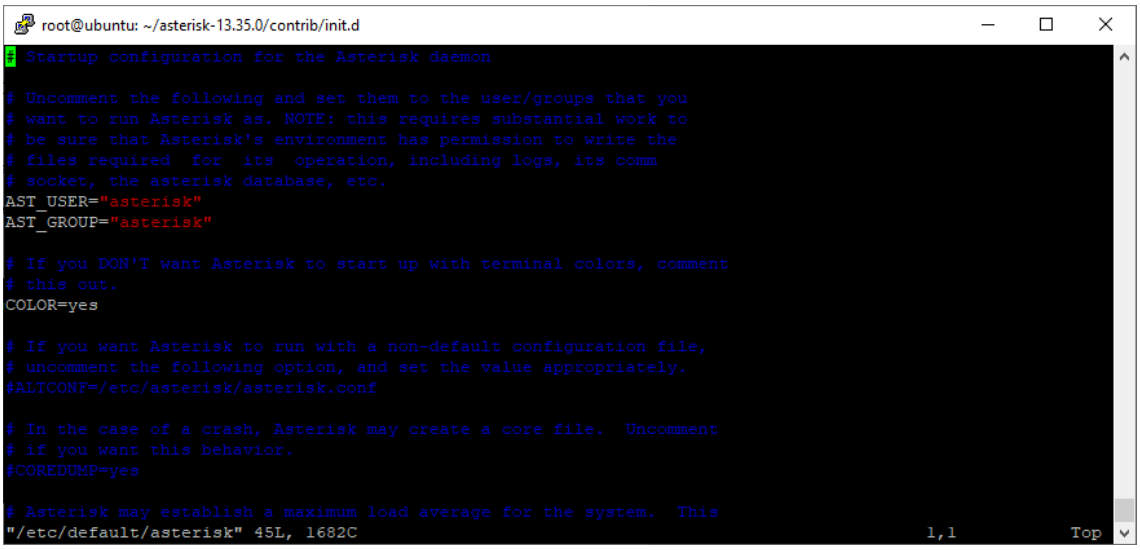
**Copy configuration (ect_default_asterisk ) files from /contrib/init.d to /etc/default/asterist** and enable AST_USER and AST_GROUP

```
# cd /asterisk-13.35.0/contrib/init.d

# cp ect_default_asterisk  /etc/default/asterisk
```




**Open copied file and enable AST_USER and AST_GROUP**

```
# vim /etc/default/asterisk

AST_USER="asterisk"
AST_GROUP="asterisk"
```

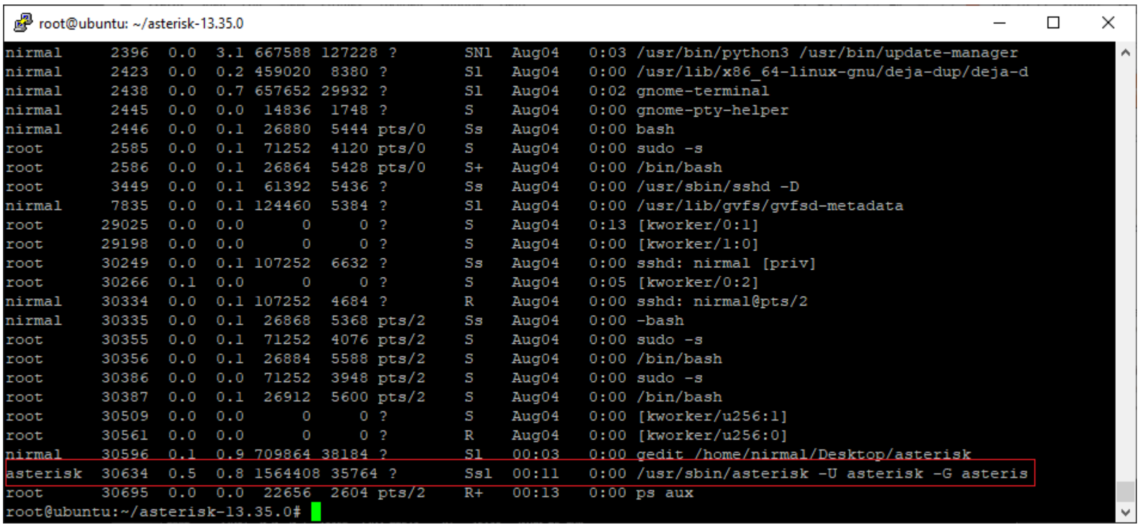



**Start Asterisk**

```
# /etc/init.d/asterisk start
```

**Check process list**

```
# ps aux
```




**Privilege Escalations with Dialplan Functions [Issue: - "Privilege escalation protection disabled"]**

Dialplan functions within Asterisk are incredibly powerful, which is wonderful for building applications using Asterisk. Dialplan functions can be 'read' or 'written'. But during the read or write execution, certain diaplan functions do much more.

For example, reading the **SHELL ()** function can execute arbitrary commands on the system Asterisk is running on. Writing to the **FILE ()** function can change any file that Asterisk has write access to.

From the context of executing the dialplan defined in **extensions.conf**, this is not a problem. From other contexts, however, it could be a problem.

Channel variables can be get or set via external mechanisms (like AMI or ARI), during which time dialplan functions are evaluated. These external protocols have permission levels associated with them, so the fact that executing a read on a certain function could effect a change on the system results in a **privilege escalation**.

In order to avoid this security issue, Asterisk can now inhibit the execution of privilege escalating functions from external protocols. These functions will continue to execute normally when invoked from the dialplan. For legacy configurations where the less secure behavior is desired, a new flag called **live_dangerously** has been added to **asterisk.conf**. When set to yes, Asterisk will allow privilege escalating functions to execute, even from external protocols.

For Asterisk 11 and earlier, in order to maintain backward compatibility, **live_dangerously** defaults to yes. In Asterisk 12, that default was changed to no.

```
# vim /etc/asterisk/asterisk.conf
```

```
root@ubuntu: ~/asterisk-13.35.0
[directories](!)
astetcdir => /etc/asterisk
astmoddir => /usr/lib/asterisk/modules
astvarlibdir => /var/lib/asterisk
astdbdir => /var/lib/asterisk
astkeydir => /var/lib/asterisk
astdatadir => /var/lib/asterisk
astagidir => /var/lib/asterisk/agi-bin
astspooldir => /var/spool/asterisk
astrundir => /var/run/asterisk
astlogdir => /var/log/asterisk
astsbindir => /usr/sbin

[options]
;verbose = 3
;debug = 3
;alwaysfork = yes                ; Same as -F at startup.
;nofork = yes                    ; Same as -f at startup.
;quiet = yes                     ; Same as -q at startup.
;timestamp = yes                 ; Same as -T at startup.
;execincludes = yes              ; Support #exec in config files.
;console = yes                   ; Run as console (same as -c at startup).
;highpriority = yes              ; Run realtime priority (same as -p at
                                 ; startup).
"/etc/asterisk/asterisk.conf" 123L, 5421C                    1,1           Top
```

Enable parameter **live_dangerously**

```
root@ubuntu: ~/asterisk-13.35.0
                                 ; gosub - Invoke the stdexten using a gosub as
                                 ;         documented in extensions.conf.sample.
                                 ; Default gosub.
live_dangerously = no            ; Enable the execution of 'dangerous' dialplan
                                 ; functions from external sources (AMI,
                                 ; etc.) These functions (such as SHELL) are
                                 ; considered dangerous because they can allow
                                 ; privilege escalation.
                                 ; Default no
;entityid=00:11:22:33:44:55      ; Entity ID.
                                 ; This is in the form of a MAC address.
                                 ; It should be universally unique.
                                 ; It must be unique between servers communicating
                                 ; with a protocol that uses this value.
                                 ; This is currently is used by DUNDi and
                                 ; Exchanging Device and Mailbox State
                                 ; using protocols: XMPP, Corosync and PJSIP.
;rtp_pt_dynamic = 96             ; Normally the Dynamic RTP Payload Type numbers
                                 ; are 96-127, which allow 32 formats. When you
                                 ; use more and receive the message "No Dynamic
                                 ; RTP mapping available", extend the dynamic
                                 ; range by going for 35 (or 0) instead of 96.
                                 ; This allows 29 (or 64) more formats. 96 is the
                                 ; default because any number below might be
-- INSERT --                                                 94,22         90%
```

Stop and start asterisk server

```
# /etc/init.d/asterisk stop

# /etc/init.d/asterisk start
```

```
root@ubuntu: ~/asterisk-13.35.0
root@ubuntu:~/asterisk-13.35.0# /etc/init.d/asterisk stop
 * Stopping Asterisk PBX: asterisk                                    [ OK ]
root@ubuntu:~/asterisk-13.35.0# /etc/init.d/asterisk start
 * Starting Asterisk PBX: asterisk                                    [ OK ]
root@ubuntu:~/asterisk-13.35.0#
```

**The Asterisk CLI** (Asterisk Command Line interface)

Connecting to the Asterisk CLI There are many options that you can apply following the 'asterisk' command at the Linux terminal. A few of the most common and useful are listed and described below. You can see a detailed list of all the valid options by running 'asterisk –h'.

```
root@ubuntu: ~/asterisk-13.35.0
Valid Options:
   -V              Display version number and exit
   -C <configfile> Use an alternate configuration file
   -G <group>      Run as a group other than the caller
   -U <user>       Run as a user other than the caller
   -c              Provide console CLI
   -d              Enable extra debugging
   -f              Do not fork
   -F              Always fork
   -g              Dump core in case of a crash
   -h              This help screen
   -i              Initialize crypto keys at startup
   -L <load>       Limit the maximum load average before rejecting new calls
   -M <value>      Limit the maximum number of calls to the specified value
   -m              Mute debugging and console output on the console
   -n              Disable console colorization
   -p              Run as pseudo-realtime thread
   -q              Quiet mode (suppress output)
   -r              Connect to Asterisk on this machine
   -R              Same as -r, except attempt to reconnect if disconnected
   -s <socket>     Connect to Asterisk via socket <socket> (only valid with -r)
   -t              Record soundfiles in /var/tmp and move them where they
                   belong after they are done
   -T              Display the time in [Mmm dd hh:mm:ss] format for each line
                   of output to the CLI
   -v              Increase verbosity (multiple v's = more verbose)
   -x <cmd>        Execute command <cmd> (implies -r)
   -X              Execute includes by default (allows #exec in asterisk.conf)
   -W              Adjust terminal colors to compensate for a light background

root@ubuntu:~/asterisk-13.35.0#
```

### asterisk –r

If you've started Asterisk using a script or by running 'asterisk' at the Linux terminal, you can then connect to that running instance of asterisk with the '-r' option. You will be presented license and warranty information, followed by the CLI prompt:

```
root@ubuntu: ~
root@ubuntu:~# asterisk -r
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 30782)
ubuntu*CLI>
```

### asterisk –c

Starts Asterisk in console mode. This assumes you have not already started asterisk as a background daemon process by running 'asterisk' (or a script). You will immediately be connected to the Asterisk CLI. Run 'core stop now' at the CLI to be end the process and return to the Linux prompt.


**asterisk –x**

This will issue a valid CLI command to Asterisk and provide the standard output to the Terminal. This should be immediately followed by the CLI command in quotes e.g. '**asterisk –x "sip show peers**"'

**Helpful CLI Commands**

**core show help**: - lists valid CLI commands.

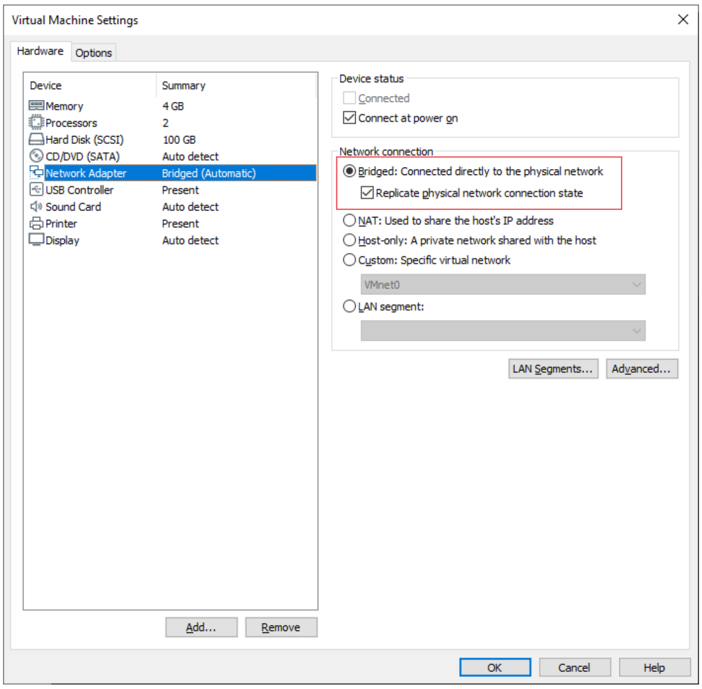
**core restart now**: - Immediately restarts Asterisk. You will exit the CLI and be returned to the Linux prompt.

**core stop now**: - Immediately stops Asterisk. You will exit the CLI and be returned to the Linux prompt.

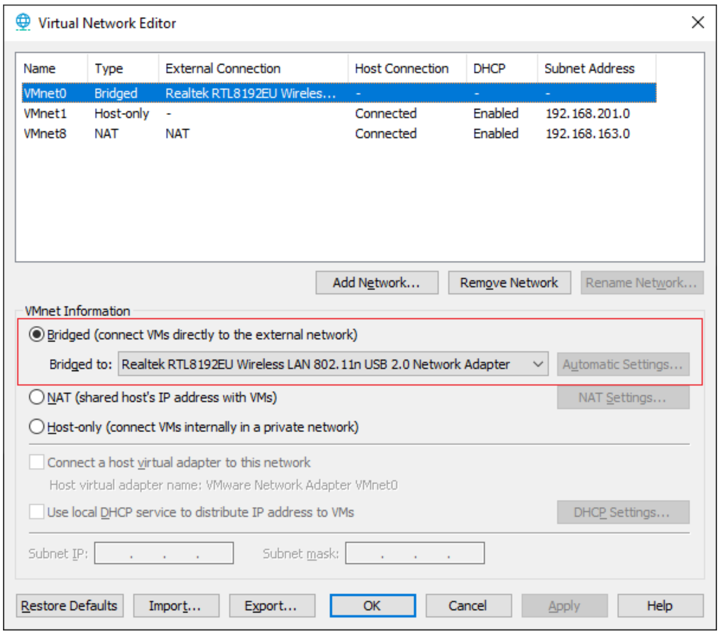
**sip show peers**: - Lists all configured SIP devices. The output includes the account name used for a given device and its IP address.

**dialplan show**: - Displays all of the active (in memory) dialplan. This includes, but is not limited to, the configuration contained in '/etc/asterisk/extensions.conf'.

## Changing Network Setting "NAT" to "Bridged"




If not connected, Go to your Virtual Adapter Editor, select Bridged, and then select Bridged to, choose your Network Adapter.

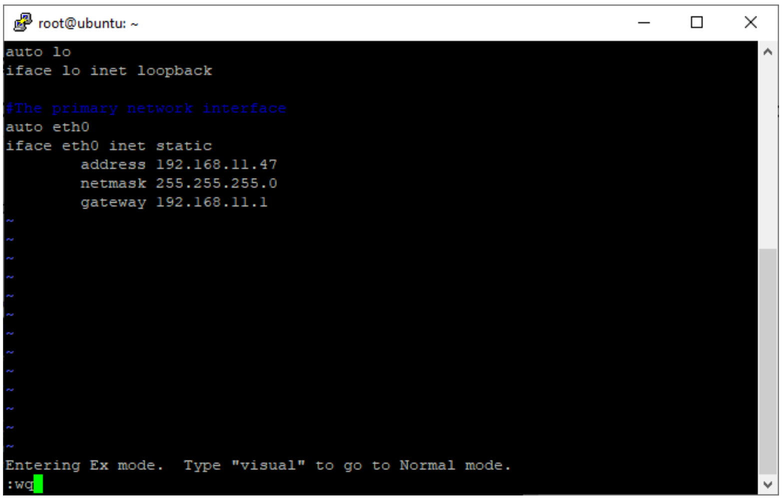


## Asterisk PBX Network Setup

After Changing network settings **NAT** to **Bridged** and go to **/etc/network/interfaces** and modify as below

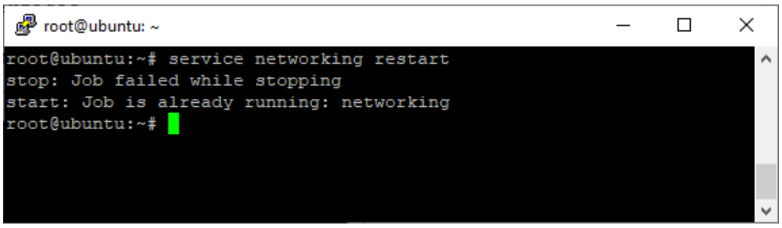
```
# vim /etc/network/interfaces
```

Add **address**, **netmask** and **gateway**



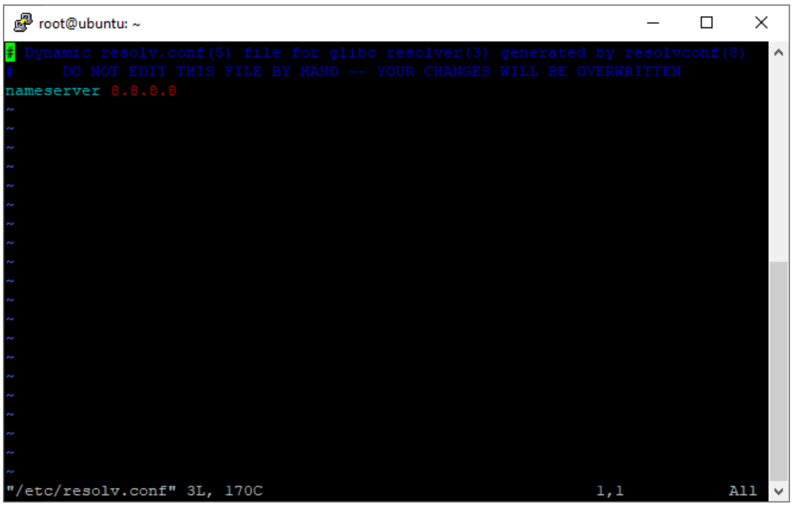

Restart Network Service

```
# service networking restart
```




Go to /etc/resolv.conf add nameserver (if not)

```
#sudo vim /etc/resolv.conf
```



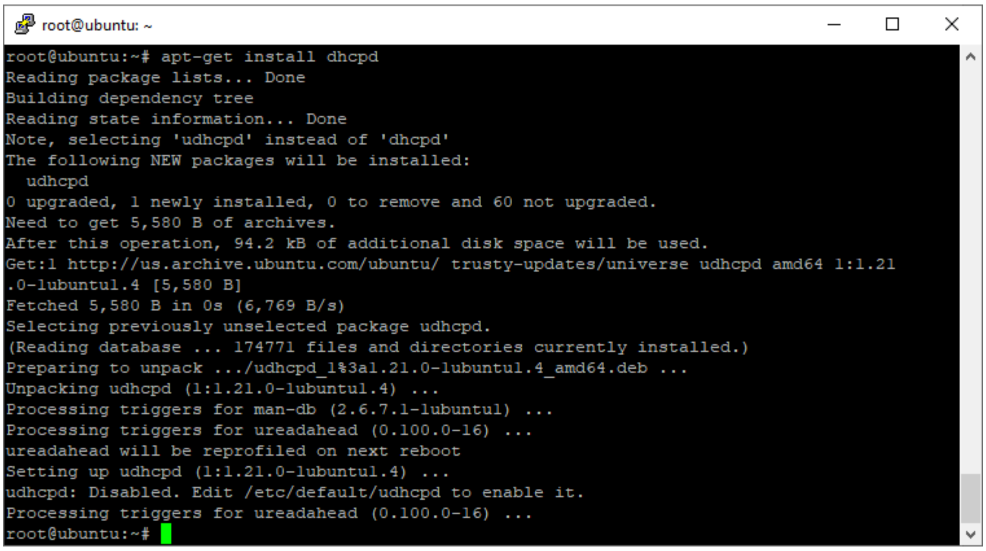
**Setting up DHCP Server**

A **DHCP Server** is a network server that automatically provides and assigns IP addresses, default gateways and other network parameters to client devices. It relies on the standard protocol known as Dynamic Host Configuration Protocol or DHCP to respond to broadcast queries by clients.

A DHCP server automatically sends the required network parameters for clients to properly communicate on the network. Without it, the network administrator has to manually set up every client that joins the network, which can be cumbersome, especially in large networks. DHCP servers usually assign each client with a unique dynamic IP address, which changes when the client’s lease for that IP address has expired.

1. Install DHCP Server

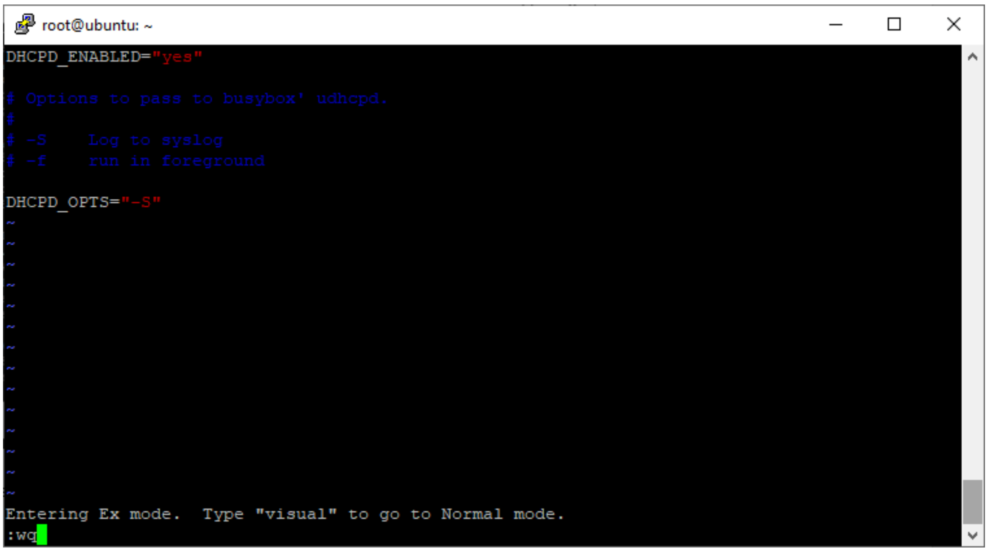
```
# apt-get install dhcpd
```



```
root@ubuntu: ~
root@ubuntu:~# apt-get install dhcpd
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'udhcpd' instead of 'dhcpd'
The following NEW packages will be installed:
  udhcpd
0 upgraded, 1 newly installed, 0 to remove and 60 not upgraded.
Need to get 5,580 B of archives.
After this operation, 94.2 kB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu/ trusty-updates/universe udhcpd amd64 1:1.21
.0-1ubuntu1.4 [5,580 B]
Fetched 5,580 B in 0s (6,769 B/s)
Selecting previously unselected package udhcpd.
(Reading database ... 174771 files and directories currently installed.)
Preparing to unpack .../udhcpd_1%3a1.21.0-1ubuntu1.4_amd64.deb ...
Unpacking udhcpd (1:1.21.0-1ubuntu1.4) ...
Processing triggers for man-db (2.6.7.1-1ubuntu1) ...
Processing triggers for ureadahead (0.100.0-16) ...
ureadahead will be reprofiled on next reboot
Setting up udhcpd (1:1.21.0-1ubuntu1.4) ...
udhcpd: Disabled. Edit /etc/default/udhcpd to enable it.
Processing triggers for ureadahead (0.100.0-16) ...
root@ubuntu:~#
```

2. Currently disabled, Edit /etc/default/udhcpd to enable it. (DHCPD_ENABLED –“Yes”)

```
# vi /etc/default/udhcpd
```



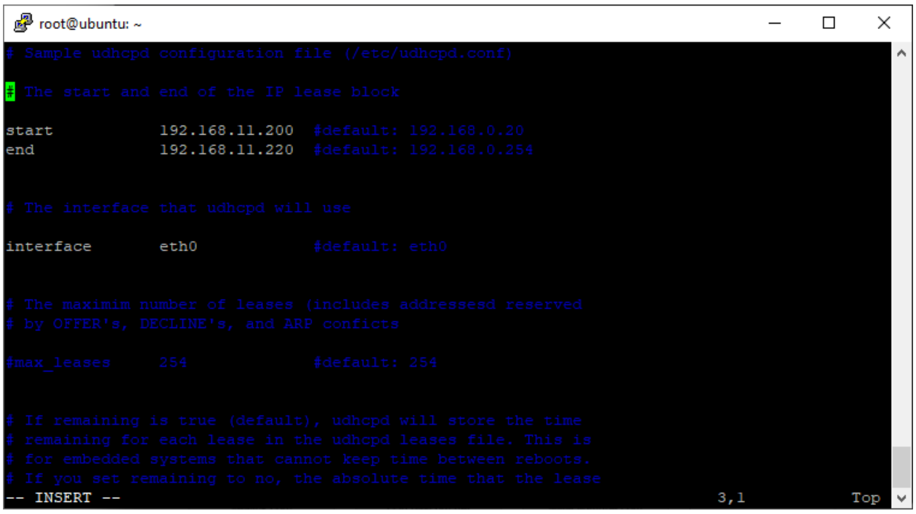
```
root@ubuntu: ~
DHCPD_ENABLED="yes"

# Options to pass to busybox' udhcpd.
#
# -S    Log to syslog
# -f    run in foreground

DHCPD_OPTS="-S"
~
~
~
~
~
~
~
~
~
~
~
~
~
~
Entering Ex mode.  Type "visual" to go to Normal mode.
:wq
```

3. Configuring DHCP Server

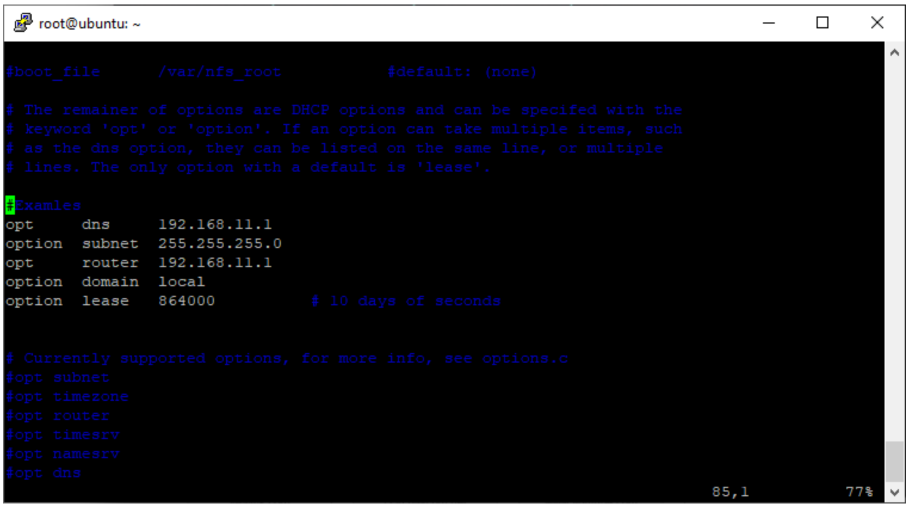
   Open /etc/udhcpd.conf and define range for (start and end) telephone and edit DNS server and Gateways

```
# vi /etc/udhcpd.conf
```

   Define range start and end ip address

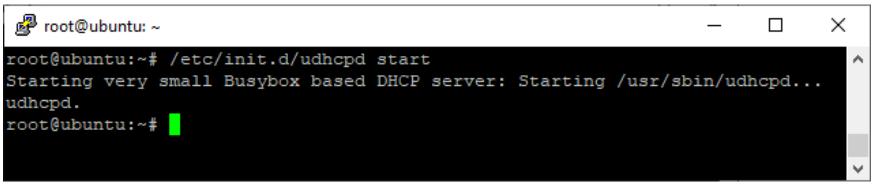



   Change DNS ,router (gateway) and remove win and dns




4. Start DHCP server

```
# /etc/init.d/udhcpd start
```



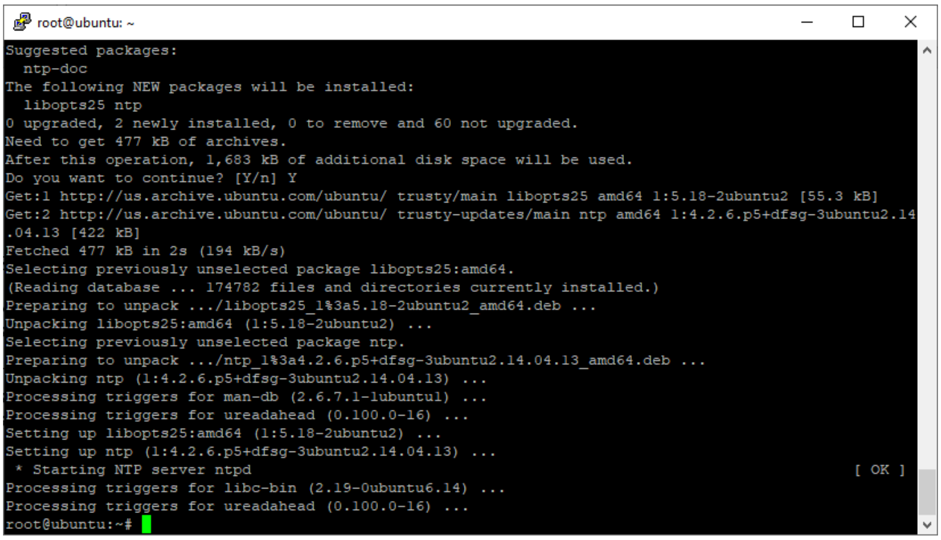
**Setting up NTP Server (ntp)**

NTP stands for Network Time Protocol, and it is an Internet protocol used to synchronize the clocks of computers to same time reference.

The Network Time Protocol (NTP) is a networking protocol for clock synchronization between computer systems over packet-switched, variable-latency data networks.

1. Install ntp server

   `# apt-get install ntp`



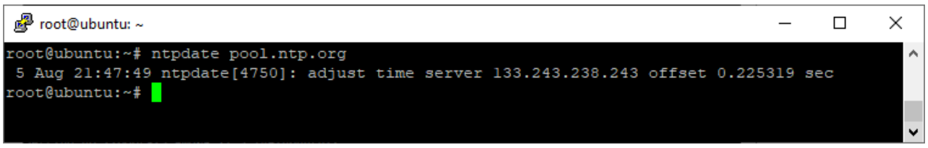
```
root@ubuntu: ~
Suggested packages:
  ntp-doc
The following NEW packages will be installed:
  libopts25 ntp
0 upgraded, 2 newly installed, 0 to remove and 60 not upgraded.
Need to get 477 kB of archives.
After this operation, 1,683 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://us.archive.ubuntu.com/ubuntu/ trusty/main libopts25 amd64 1:5.18-2ubuntu2 [55.3 kB]
Get:2 http://us.archive.ubuntu.com/ubuntu/ trusty-updates/main ntp amd64 1:4.2.6.p5+dfsg-3ubuntu2.14
.04.13 [422 kB]
Fetched 477 kB in 2s (194 kB/s)
Selecting previously unselected package libopts25:amd64.
(Reading database ... 174782 files and directories currently installed.)
Preparing to unpack .../libopts25_1%3a5.18-2ubuntu2_amd64.deb ...
Unpacking libopts25:amd64 (1:5.18-2ubuntu2) ...
Selecting previously unselected package ntp.
Preparing to unpack .../ntp_1%3a4.2.6.p5+dfsg-3ubuntu2.14.04.13_amd64.deb ...
Unpacking ntp (1:4.2.6.p5+dfsg-3ubuntu2.14.04.13) ...
Processing triggers for man-db (2.6.7.1-1ubuntu1) ...
Processing triggers for ureadahead (0.100.0-16) ...
Setting up libopts25:amd64 (1:5.18-2ubuntu2) ...
Setting up ntp (1:4.2.6.p5+dfsg-3ubuntu2.14.04.13) ...
 * Starting NTP server ntpd                                              [ OK ]
Processing triggers for libc-bin (2.19-0ubuntu6.14) ...
Processing triggers for ureadahead (0.100.0-16) ...
root@ubuntu:~#
```

2. Stop ntp server, update date and start again

   `# /etc/init.d/ntp stop`

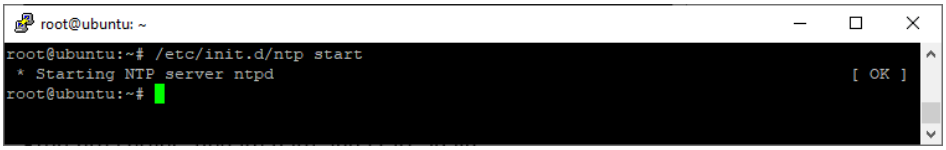
```
root@ubuntu: ~
root@ubuntu:~# /etc/init.d/ntp stop
 * Stopping NTP server ntpd                                              [ OK ]
root@ubuntu:~#
```

   `# ntpdate pool.ntp.org`



```
root@ubuntu: ~
root@ubuntu:~# ntpdate pool.ntp.org
 5 Aug 21:47:49 ntpdate[4750]: adjust time server 133.243.238.243 offset 0.225319 sec
root@ubuntu:~#
```

   `# /etc/init.d/ntp stop`



```
root@ubuntu: ~
root@ubuntu:~# /etc/init.d/ntp start
 * Starting NTP server ntpd                                              [ OK ]
root@ubuntu:~#
```

## Asterisk PBX SIP Phone Peers

A SIP server is the main component of an IP PBX, and mainly deals with the management of all SIP calls in the network. A SIP server is also referred to as a SIP Proxy or a Registrar.

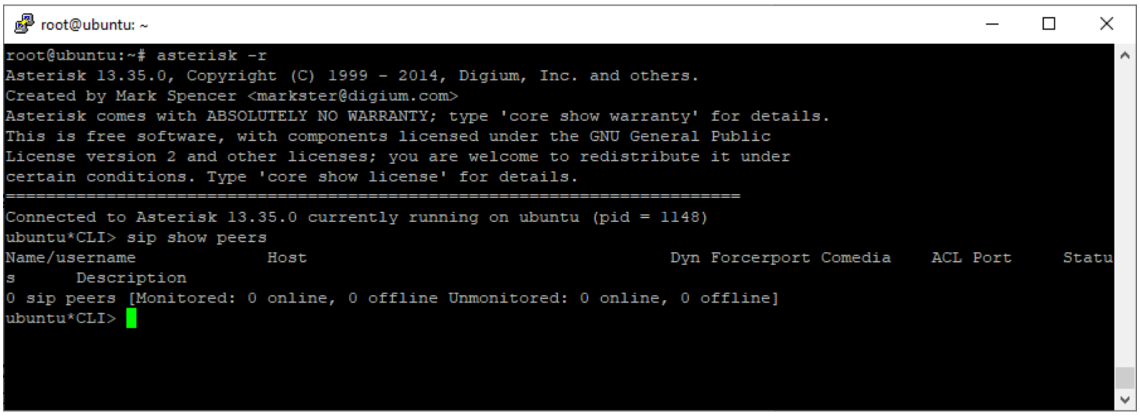
The Asterisk SIP Settings Module is used to configure the default settings used for SIP calls. Since most VOIP calls are sent using SIP, these settings can be very important to the operation of your PBX. Because this module sets the default settings, most of these settings can be overridden for a particular extension in the Extensions Module or for a particular trunk in the Trunks Module.

Every configuration about peers are generally done in **sip.conf** under **/etc/asterisk**

1. First, check available sip peers, go to asterisk console

```
# asterisk -r
# sip show peers
```

Show all SIP peers (including friends)

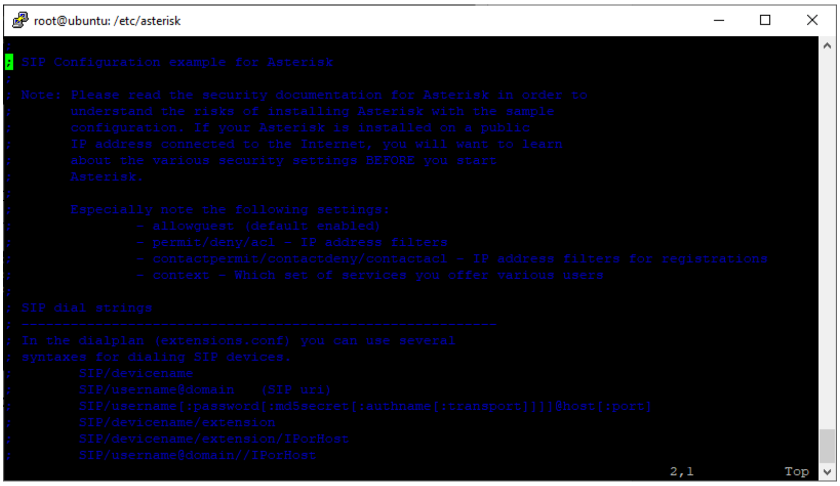



2. Go back, configure peers in sip.conf. Before making changes create backup

```
# cd /etc/asterisk
# cp sip.conf sip.conf.orig
```


3. Open and edit sip.conf

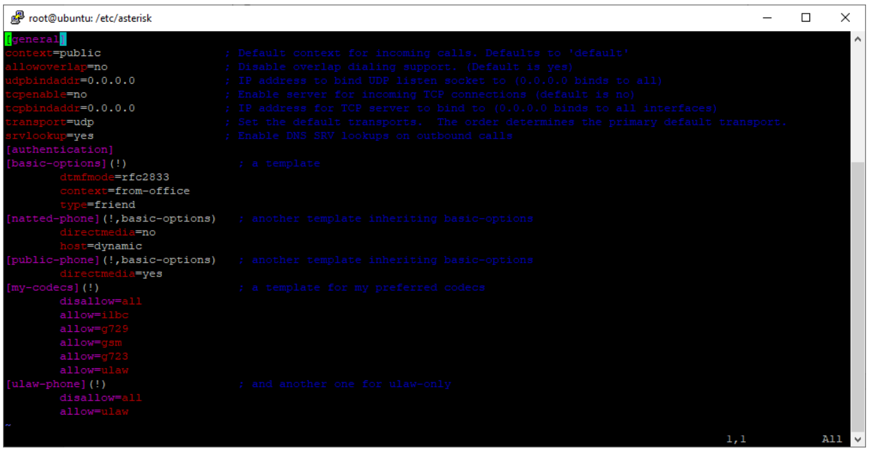
```
#vi sip.conf
```



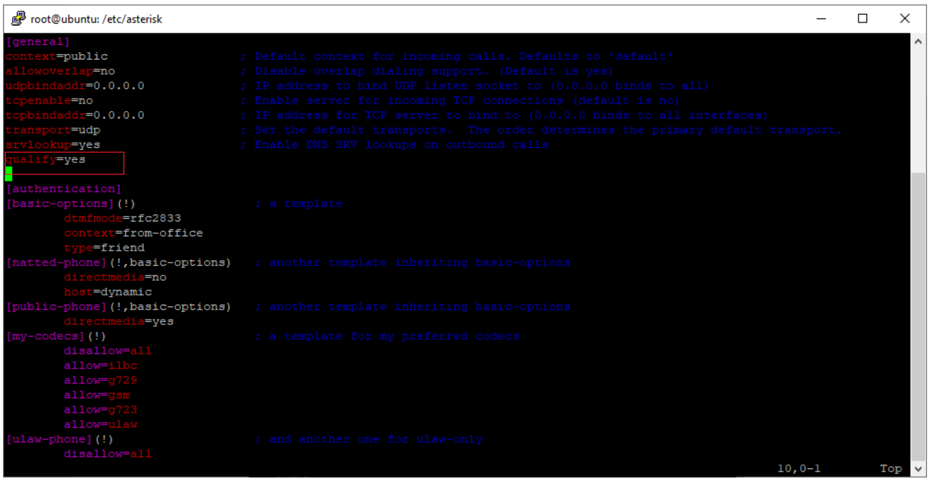

This file contain configuration and its documantations(so it is big files). To delete unwated document and comments do below steps

```
:g/^\s*;/d
:g/^$/d
```

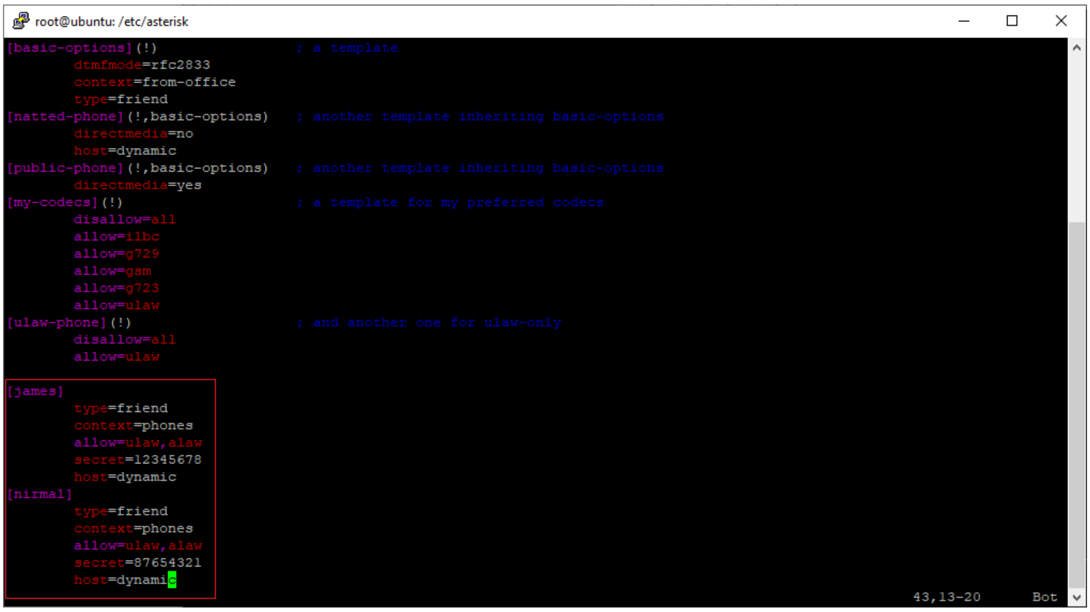
Commands in vi editor




4. Add **qualify=yes** parameter in general section (to avoid idle tone whils calling)



5. Add New peers



```
root@ubuntu: /etc/asterisk
[basic-options](!)                 ; a template
        dtmfmode=rfc2833
        context=from-office
        type=friend
[natted-phone](!,basic-options)    ; another template inheriting basic-options
        directmedia=no
        host=dynamic
[public-phone](!,basic-options)    ; another template inheriting basic-options
        directmedia=yes
[my-codecs](!)                     ; a template for my preferred codecs
        disallow=all
        allow=ilbc
        allow=g729
        allow=gsm
        allow=g723
        allow=ulaw
[ulaw-phone](!)                    ; and another one for ulaw-only
        disallow=all
        allow=ulaw

[james]
        type=friend
        context=phones
        allow=ulaw,alaw
        secret=12345678
        host=dynamic
[nirmal]
        type=friend
        context=phones
        allow=ulaw,alaw
        secret=87654321
        host=dynamic
                                                          43,13-20      Bot
```

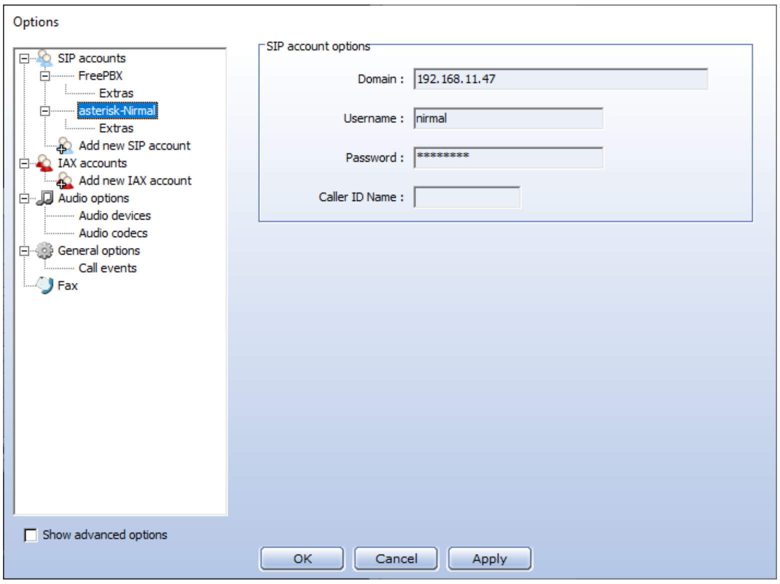
6. Go to asterisk console and check peers

```
# asterisk -rvvv
# sip reload
# sip show peers
```

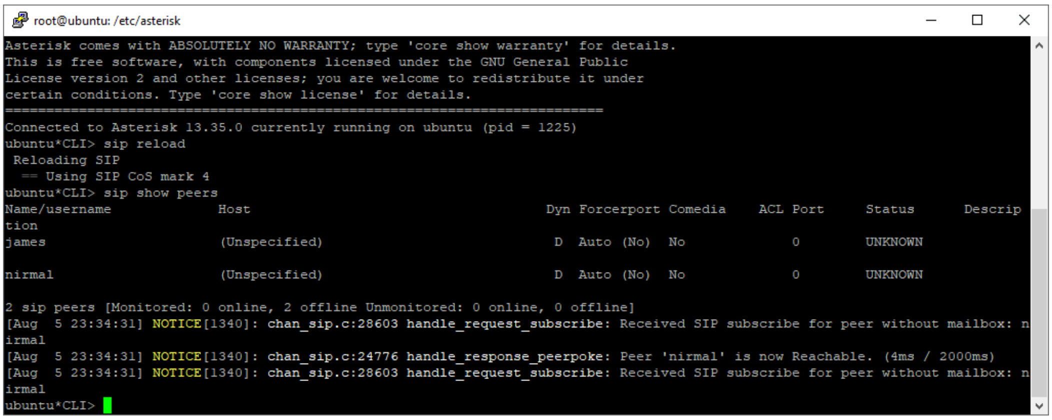
```
root@ubuntu: /etc/asterisk
root@ubuntu:/etc/asterisk# asterisk -rvvv
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1225)
ubuntu*CLI> sip reload
 Reloading SIP
  == Using SIP CoS mark 4
ubuntu*CLI> sip show peers
Name/username             Host                                    Dyn Forcerport Comedia    ACL Port     Status      Descrip
tion
james                     (Unspecified)                            D  Auto (No)  No             0        UNKNOWN

nirmal                    (Unspecified)                            D  Auto (No)  No             0        UNKNOWN

2 sip peers [Monitored: 0 online, 2 offline Unmonitored: 0 online, 0 offline]
ubuntu*CLI>
```

7. Register your phone (using Softphone)

Download Zoiper Free (any) and add new SIP account
https://www.zoiper.com/en/voip-softphone/download/

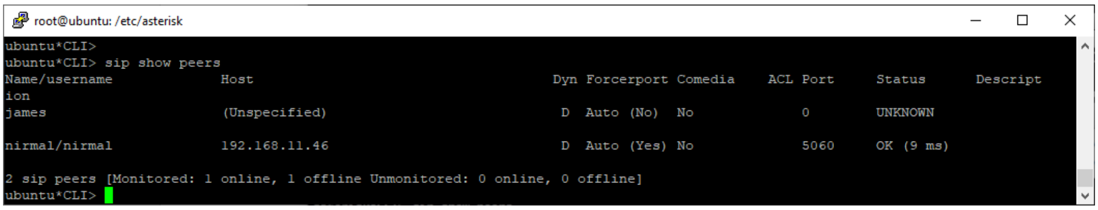



You will get console like this (while registering)




8. Check Registers peers. After registering port and host assigned automatically

```
# sip show peers
```



## Asterisk Dialplan

The dialplan is essentially a scripting language specific to Asterisk and one of the primary ways of instructing Asterisk on how to behave. It ties everything together, allowing you to route and manipulate calls in a programmatic way. The pages in this section will describe what the elements of dialplan are and how to use them in your configuration.

The Asterisk dialplan is found in the extensions.conf file in the configuration directory, typically **/etc/asterisk**.

If you modify the dialplan, you can use the Asterisk CLI command "**dialplan reload**" to load the new dialplan without disrupting service in your PBX.

### Context

The Asterisk dialplan is divided into sections, and each section is called a context. Any dialplan must begin with a **[general]** context where global configuration entries reside, but the subsequent contexts can have any name. Contexts are the means by which actual physical devices (usually telephones, but not always; for example, SIP or Zap devices) are bound to the dialplan. The configuration for every device, be it a softphone, hard phone or outgoing trunk, must specify the default context for that device.

Syntax

Contexts are defined by a name inside square brackets ("[" and "]"). Ideally the name is relevant and helps to describe the intended use for the context. This name will also be used to refer to the context elsewhere, be it in other contexts or in other Asterisk configuration files. All lines following a context name are considered part of that context, until the next context name is encountered:

```
[general]

[internal-phones]
Rules, instructions, etc.
```

### Extension

Individual entries in **extensions.conf** are called extensions. Extensions are interpreted by Asterisk every time a call is initiated, but extensions.conf is only read into Asterisk at start time. You can also refresh the dialplan during operation from the CLI (Command Line Interface) by entering the command **reload now** (which reloads all the configurations) or **extensions reload** (which reloads only the dialplan).

Syntax

- An extension consists of the following parts:
- Extension (Name or number)
- Priority (a kind of program line number)
- Application - an instruction which tells Asterisk what it should do with the call.

```
exten => Extension, Priority, Application
```

### Priority

A typical extension is composed of a multiple entry. Each entry has a priority so that Asterisk knows in what order it should execute the entries.

If you have ever worked with early versions of BASIC, you might be familiar with line numbers; priorities work in much the same way, but with one important distinction.

They are always executed in numerical order from smallest to largest, but there can be no gaps! If Asterisk executes an entry of priority n, then it will look for the next entry at n + 1. If it cannot find an entry at n + 1, it stops executing without displaying an error in the CLI.

### Application

"Application" is perhaps too expansive a term but it is the convention in Asterisk when referring to dialplan commands.

### Create sample Context

1. Backup and define context

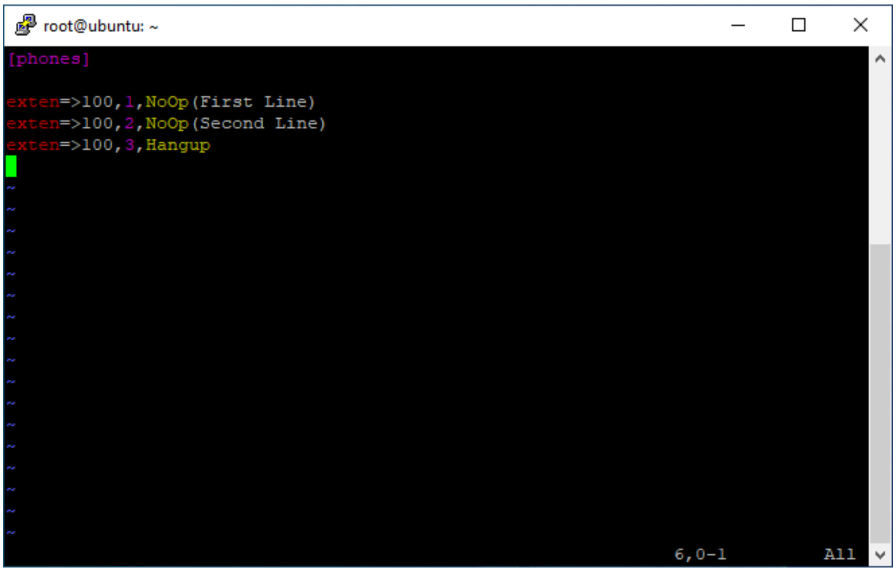
```
# cd /etc/asterisk
# cp extensions.conf extensions.conf.orig
```

2. Empty extensions.conf , because of it is too big and it contrains all documentation. Currently I don’t want this documentation contents

```
# echo “”> extensions.conf
```

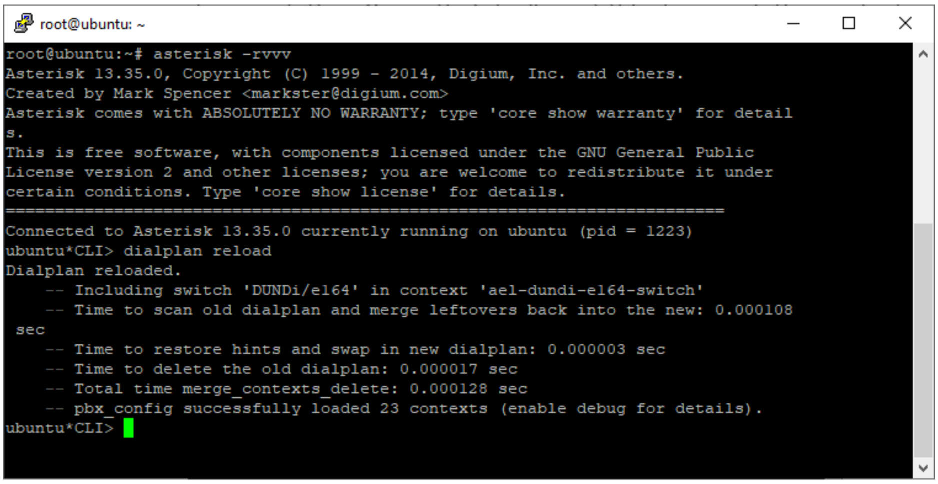
3. Open and define context

```
# vi extensions.conf
```

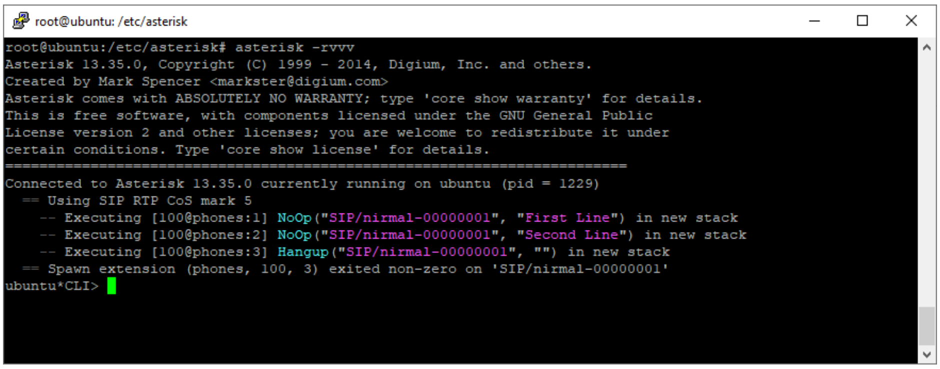



4. Open Asterisk console and Reload

```
# asterisk -rvvv
# dialpan reload
```




5. Open a Softphone, dial 100 and see Asterisk console



**Asterisk First Dial – Ring Telephone**

1. Applications (like NoOp, Hangup). Go to asterisk CLI

   `# asterisk -rvvv`

   To list all applications available

   `# core show applications`

```
root@ubuntu: /etc/asterisk
         StopPlayTones: Stop playing a tone list.
                System: Execute a system command.
            TestClient: Execute Interface Test Client.
            TestServer: Execute Interface Test Server.
              Transfer: Transfer caller to remote extension.
               TryExec: Executes dialplan application, always returning.
             TrySystem: Try executing a system command.
        UnpauseMonitor: Unpause monitoring of a channel.
    UnpauseQueueMember: Unpauses a queue member.
             UserEvent: Send an arbitrary user-defined event to parties interested in a channel (AMI users and relev
ant res_stasis applications).
               Verbose: Send arbitrary text to verbose output.
        VMAuthenticate: Authenticate with Voicemail passwords.
             VMSayName: Play the name of a voicemail user
             VoiceMail: Leave a Voicemail message.
         VoiceMailMain: Check Voicemail messages.
      VoiceMailPlayMsg: Play a single voice mail msg from a mailbox by msg id.
                  Wait: Waits for some time.
             WaitExten: Waits for an extension to be entered.
          WaitForNoise: Waits for a specified amount of noise.
           WaitForRing: Wait for Ring Application.
        WaitForSilence: Waits for a specified amount of silence.
             WaitUntil: Wait (sleep) until the current time is the given epoch.
                 While: Start a while loop.
            Zapateller: Block telemarketers with SIT.
    -= 169 Applications Registered =-
ubuntu*CLI>
```

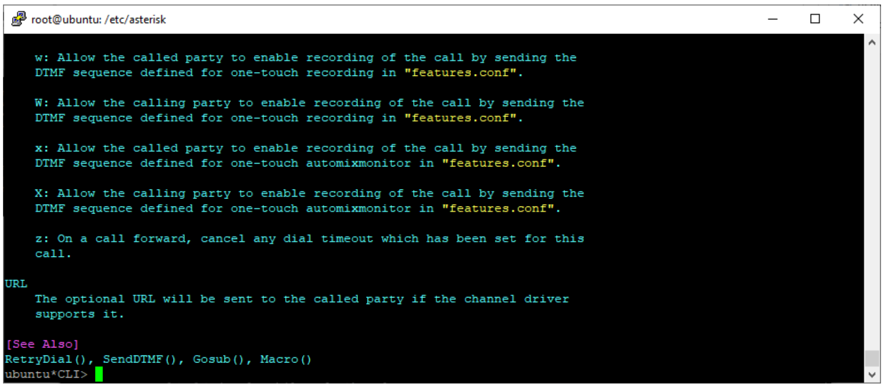
   `# core show applications like dial` (Searching)

```
root@ubuntu: /etc/asterisk
root@ubuntu:/etc/asterisk# asterisk -rvvv
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1229)
ubuntu*CLI> core show applications like dial
    -= Matching Asterisk Applications =-
                  Dial: Attempt to connect to another device or endpoint and bridge the call.
             RetryDial: Place a call, retrying on failure allowing an optional exit extension.
    -= 2 Applications Matching =-
ubuntu*CLI>
```

   `# core show applications describing dial` (Searching)

```
root@ubuntu: /etc/asterisk
SE.
             MP3Player: Play an MP3 file or M3U playlist file or stream.
                  MSet: Set channel variable(s) or function value(s).
             Originate: Originate a call.
                  Park: Park yourself.
       ParkAndAnnounce: Park and Announce.
            ParkedCall: Retrieve a parked call.
                Pickup: Directed extension call pickup.
             PlayTones: Play a tone list.
                 Queue: Queue a call for a call queue.
        RaiseException: Handle an exceptional condition.
             RetryDial: Place a call, retrying on failure allowing an optional exit extension.
              SendText: Send a Text Message on a channel.
                   Set: Set channel variable or function value.
          SIPAddHeader: Add a SIP header to the outbound call.
  SpeechActivateGrammar: Activate a grammar.
      SpeechBackground: Play a sound file and wait for speech to be recognized.
               TryExec: Executes dialplan application, always returning.
        VMAuthenticate: Authenticate with Voicemail passwords.
             VoiceMail: Leave a Voicemail message.
         VoiceMailMain: Check Voicemail messages.
    -= 42 Applications Matching =-
ubuntu*CLI>
```

`# core show application dial` (Opening application)

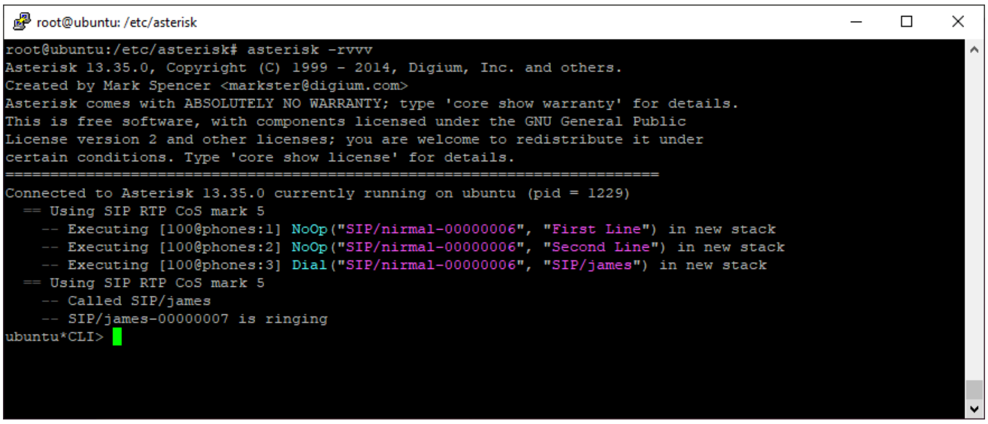



2. Open extension.conf and parameter exten => 100,1,Dial(SIP/james). James is a peer name created before

```
# cd /etc/asterisk
# vi extensions.conf
```


3. Open Asterisk CLI , reload dialplan and dial 100

```
# asterisk -rvvv
# dialplan reload
```



**To solve issue :-** ERROR[1292]: cdr_csv.c:315 csv_log: Unable to re-open master file /var/log/asterisk//cdr-csv//Master.csv : Permission denied

This file is used to write call details.

```
root@ubuntu: /etc/asterisk
  == Using SIP RTP CoS mark 5
    -- Executing [100@phones:1] NoOp("SIP/nirmal-00000006", "First Line") in new stack
    -- Executing [100@phones:2] NoOp("SIP/nirmal-00000006", "Second Line") in new stack
    -- Executing [100@phones:3] Dial("SIP/nirmal-00000006", "SIP/james") in new stack
  == Using SIP RTP CoS mark 5
    -- Called SIP/james
    -- SIP/james-00000007 is ringing
    -- SIP/james-00000007 is ringing
    -- SIP/james-00000007 answered SIP/nirmal-00000006
    -- Channel SIP/james-00000007 joined 'simple_bridge' basic-bridge <7ba3f8f7-0344-495c-ad31-dc27577932b7
>
    -- Channel SIP/nirmal-00000006 joined 'simple_bridge' basic-bridge <7ba3f8f7-0344-495c-ad31-dc27577932b
7>
    -- Started music on hold, class 'default', on channel 'SIP/james-00000007'
    -- Stopped music on hold on SIP/james-00000007
    -- Channel SIP/james-00000007 left 'native_rtp' basic-bridge <7ba3f8f7-0344-495c-ad31-dc27577932b7>
    -- Channel SIP/nirmal-00000006 left 'native_rtp' basic-bridge <7ba3f8f7-0344-495c-ad31-dc27577932b7>
  == Spawn extension (phones, 100, 3) exited non-zero on 'SIP/nirmal-00000006'
[Aug  6 21:51:15] ERROR[1292]: cdr_csv.c:315 csv_log: Unable to re-open master file /var/log/asterisk//cdr-
csv//Master.csv : Permission denied
ubuntu*CLI>
```

Problem : - File is not exist (create one file)

To Create file

```
# touch /var/log/asterisk//cdr-csv//Master.csv
```

Changing Permission

```
# chown asterisk /var/log/asterisk//cdr-csv//Master.csv
```

```
root@ubuntu: /etc/asterisk
root@ubuntu:/etc/asterisk# touch /var/log/asterisk//cdr-csv//Master.csv
root@ubuntu:/etc/asterisk# chown asterisk /var/log/asterisk//cdr-csv//Master.csv
root@ubuntu:/etc/asterisk#
```

To check dial 100 again

```
root@ubuntu: /etc/asterisk
root@ubuntu:/etc/asterisk# chown asterisk /var/log/asterisk//cdr-csv//Master.csv
root@ubuntu:/etc/asterisk# asterisk -rvvv
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1229)
  == Using SIP RTP CoS mark 5
    -- Executing [100@phones:1] NoOp("SIP/nirmal-00000008", "First Line") in new stack
    -- Executing [100@phones:2] NoOp("SIP/nirmal-00000008", "Second Line") in new stack
    -- Executing [100@phones:3] Dial("SIP/nirmal-00000008", "SIP/james") in new stack
  == Using SIP RTP CoS mark 5
    -- Called SIP/james
    -- SIP/james-00000009 is ringing
    -- SIP/james-00000009 answered SIP/nirmal-00000008
    -- Channel SIP/james-00000009 joined 'simple_bridge' basic-bridge <cacbd851-9d1e-46e1-a9c6-59900535125c>
    -- Channel SIP/nirmal-00000008 joined 'simple_bridge' basic-bridge <cacbd851-9d1e-46e1-a9c6-59900535125c>
    -- Started music on hold, class 'default', on channel 'SIP/james-00000009'
ubuntu*CLI>
```

## Replacing hardcoded values in extension.conf

Replace line number

```
exten=>100,1,NoOp(First Line)
exten=>100,2,NoOp(Second Line)
exten=>100,3,Dial(SIP/james)
exten=>100,4,Hangup
```

number replace with "n" and number start with 1

```
exten=>100,1,NoOp(First Line)
exten=>100,n,NoOp(Second Line)
exten=>100,n,Dial(SIP/james)
exten=>100,n,Hangup
```


## Use same extension

```
exten =>100,1,NoOp(First Line)
same  =>100,n,NoOp(Second Line)
same  =>100,n,Dial(SIP/james)
same  =>100,n,Hangup
```


## Calling Both Direction

## Applications in the Dialplan

This section is a comprehensive description of the applications available for use in the dialplan **(/etc/asterisk/extensions.conf**). To use an application, the module to which it belongs must be loaded; this is configured in the **[modules]** section of **/etc/asterisk/modules.conf** with **autoload=yes** or explicitly with **load => app_applicationname.so.**

You can see which modules and applications are available in Asterisk with by entering **show applications** or **show application application_name** (Asterisk 1.2) and **core show applications** oder **core show application application_name** (Asterisk 1.4).

Take care not to confuse **applications** or **commands** with **functions**. When required, functions are called within commands in the dialplan. "Application" is perhaps too expansive a term but it is the convention in Asterisk when referring to dialplan commands.

**Important:** Be sure to separate parameters with the "," (comma) or "|" (pipe) depending on the version of Asterisk. In this book we use the "," primarily.

To list all installed application

```
#core show applications
```

### Answer, Playback, and Hangup Applications

As its name suggests, the **Answer()** application answers an incoming call. The **Answer()** application takes a delay (in milliseconds) as its first parameter. Adding a short delay is often useful for ensuring that the remote end pointing has time to begin processing audio before you play a sound prompt. Otherwise, you may not hear the very beginning of the prompt.

### Knowing When to Answer a Call

When you're first learning your way around the Asterisk dialplan, it may be a bit confusing knowing when to use the Answer() application, and when not to.

If Asterisk is simply going to pass the call off to another device using the **Dial()** application, you probably don't want to call the answer the call first. If, on the other hand, you want Asterisk to play sound prompts or gather input from the caller, it's probably a good idea to call the **Answer()** application before doing anything else.

The **Playback()** application loads a sound prompt from disk and plays it to the caller, ignoring any touch tone input from the caller. The first parameter to the dialplan application is the filename of the sound prompt you wish to play, without a file extension. If the channel has not already been answered, **Playback()** will answer the call before playing back the sound prompt, unless you pass **noanswer** as the second parameter.

To avoid the first few milliseconds of a prompt from being cut off you can play a second of silence. For example, if the prompt you wanted to play was hello-world which would look like

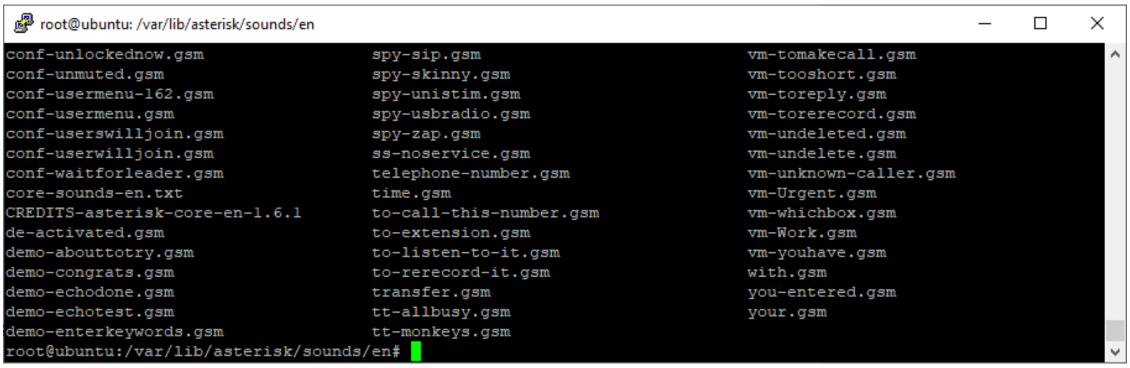
### File Playback

The playback file has same name with different encoding. But asterisk pick right files intelligently

**Implement Playback  Application**

1. Check File Location (tt start files are for testing pupose)

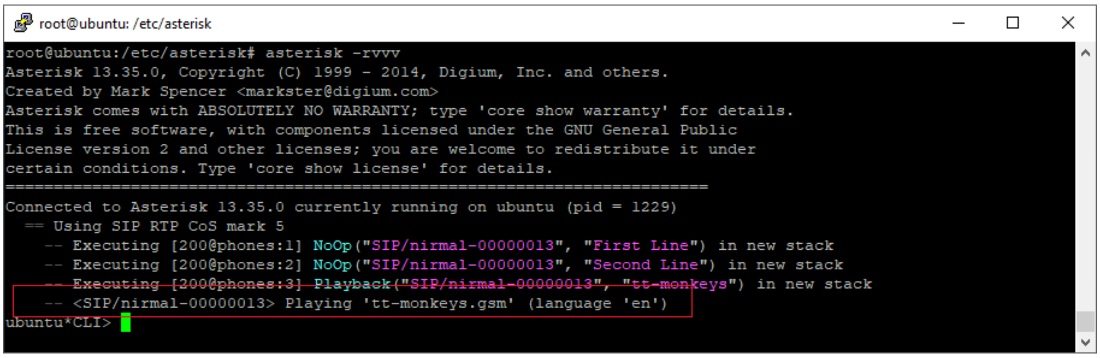
```
# cd /var/lib/asterisk/sounds/en
# ls
```




2. Add parameter in extensions.conf

```
# vi /etc/asterisk/extensions.conf
```


3. Go asterisk console, Reload dialplan and dial 200 through softphone



**Incoming Calls Simulation –** (Call From Outside)

1. Go to dialpan config file :- extensions.conf

   #vi /etc/asterisk/extensions.conf

```
root@ubuntu: /etc/asterisk
[phones]

exten=>100,1,NoOp(Call For Nirmal)
same=>n,Dial(SIP/nirmal)
same=>n,Hangup

exten=>200,1,NoOp(Call For James)
same=>n,Dial(SIP/james)
same=>n,Hangup

~
~
~
~
~
"/etc/asterisk/extensions.conf" 10L, 160C                1,1          All
```

2. Go to asrerisk console , reload dialplan and Check both number are working or not using softphone

   # asterisk -rvvv
   # dialplan reload

```
root@ubuntu: ~
root@ubuntu:~# asterisk -rvvv
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for detail
s.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1225)
ubuntu*CLI> dialplan reload
Dialplan reloaded.
    -- Including switch 'DUNDi/e164' in context 'ael-dundi-e164-switch'
    -- Time to scan old dialplan and merge leftovers back into the new: 0.000160
 sec
    -- Time to restore hints and swap in new dialplan: 0.000001 sec
    -- Time to delete the old dialplan: 0.000017 sec
    -- Total time merge_contexts_delete: 0.000178 sec
    -- pbx_config successfully loaded 23 contexts (enable debug for details).
ubuntu*CLI>
```

```
  == Using SIP RTP CoS mark 5
    -- Executing [200@phones:1] NoOp("SIP/nirmal-00000004", "Call For James") in new stack
    -- Executing [200@phones:2] Dial("SIP/nirmal-00000004", "SIP/james") in new stack
  == Using SIP RTP CoS mark 5
    -- Called SIP/james
    -- SIP/james-00000005 is ringing
    -- Got SIP response 486 "Busy Here" back from 192.168.11.46:5060
    -- SIP/james-00000005 is busy
  == Everyone is busy/congested at this time (1:1/0/0)
    -- Executing [200@phones:3] Hangup("SIP/nirmal-00000004", "") in new stack
  == Spawn extension (phones, 200, 3) exited non-zero on 'SIP/nirmal-00000004'
```

```
  == Using SIP RTP CoS mark 5
    -- Executing [100@phones:1] NoOp("SIP/james-00000006", "Call For Nirmal") in new stack
    -- Executing [100@phones:2] Dial("SIP/james-00000006", "SIP/nirmal") in new stack
  == Using SIP RTP CoS mark 5
    -- Called SIP/nirmal
    -- SIP/nirmal-00000007 is ringing
    -- Got SIP response 486 "Busy Here" back from 192.168.11.46:5060
    -- SIP/nirmal-00000007 is busy
  == Everyone is busy/congested at this time (1:1/0/0)
    -- Executing [100@phones:3] Hangup("SIP/james-00000006", "") in new stack
  == Spawn extension (phones, 100, 3) exited non-zero on 'SIP/james-00000006'
```

3. Create **New Peer** for Outside Calls in Sip.conf

# vi /etc/asterisk/sip.conf

```
root@ubuntu: ~
        allow=g723
        allow=ulaw
[ulaw-phone](!)                   ; and another one for ulaw-only
        disallow=all
        allow=ulaw

[james]
        type=friend
        context=phones
        allow=ulaw,alaw
        secret=12345678
        host=dynamic
[nirmal]
        type=friend
        context=phones
        allow=ulaw,alaw
        secret=87654321
        host=dynamic

[outside]
        type=friend
        context=incoming
        allow=ulaw,alaw
        secret=12345678
        host=dynamic
                                                                          44,0-1         Bot
```

4. Go to asterisk console and reload sip (sip reload command)

```
root@ubuntu:~# asterisk -rvvv
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1225)
ubuntu*CLI> sip reload
 Reloading SIP
  == Using SIP CoS mark 4
ubuntu*CLI>
```

5. Register newly created phone

```
ubuntu*CLI> sip show peers
Name/username             Host                                    Dyn Forceport Comedia    ACL Port     Status
  Description
james/james               192.168.11.46                            D  Auto (No)  No            5060     OK (17 ms)

nirmal/nirmal             192.168.11.46                            D  Auto (No)  No            5060     OK (12 ms)

outside/outside           192.168.11.46                            D  Auto (No)  No            59782    OK (1 ms)

3 sip peers [Monitored: 3 online, 0 offline Unmonitored: 0 online, 0 offline]
ubuntu*CLI>
```

6. Add new context in extension.conf

# vi /etc/asterisk/extensions.conf

```
root@ubuntu: ~
[incoming]

exten=> 991123123,1,Goto(phones,100,1)

[phones]

exten=>100,1,NoOp(Call For Nirmal)
same=>n,Dial(SIP/nirmal)
same=>n,Hangup

exten=>200,1,NoOp(Call For James)
same=>n,Dial(SIP/james)
same=>n,Hangup

~
~
                                                                          1,1            All
```

7. Reload dialplan and call specified number

```
root@ubuntu: ~
root@ubuntu:~# asterisk -rvvv
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1225)
  == Using SIP RTP CoS mark 5
    -- Executing [991123123@incoming:1] Goto("SIP/outside-0000000e", "phones,100,1") in new stack
    -- Goto (phones,100,1)
    -- Executing [100@phones:1] NoOp("SIP/outside-0000000e", "Call For Nirmal") in new stack
    -- Executing [100@phones:2] Dial("SIP/outside-0000000e", "SIP/nirmal") in new stack
  == Using SIP RTP CoS mark 5
    -- Called SIP/nirmal
    -- SIP/nirmal-0000000f is ringing
    -- SIP/nirmal-0000000f answered SIP/outside-0000000e
    -- Channel SIP/nirmal-0000000f joined 'simple_bridge' basic-bridge <fc75f3c8-2076-4d4b-b8f7-8331b8797569>
    -- Channel SIP/outside-0000000e joined 'simple_bridge' basic-bridge <fc75f3c8-2076-4d4b-b8f7-8331b8797569>
ubuntu*CLI>
```

**<u>Outgoing Calls Configuration –</u>** (Calling Outside)

1. Go to dialpan config file :- extensions.conf

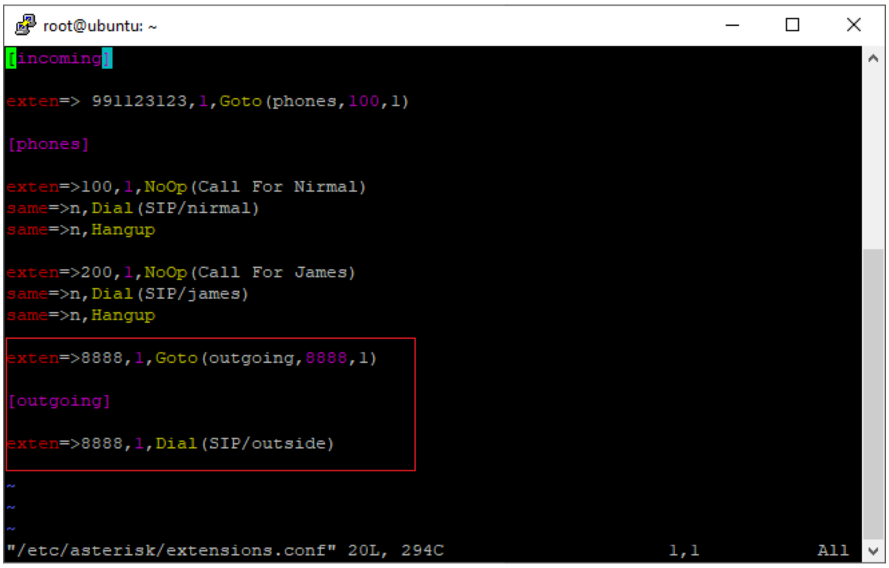
```
#vi /etc/asterisk/extensions.conf
```

2. Edit extensions.conf add below context

```
[outgoing]
exten=>8888,1,Dial(SIP/outside)
```

And goto [Phones] section and add

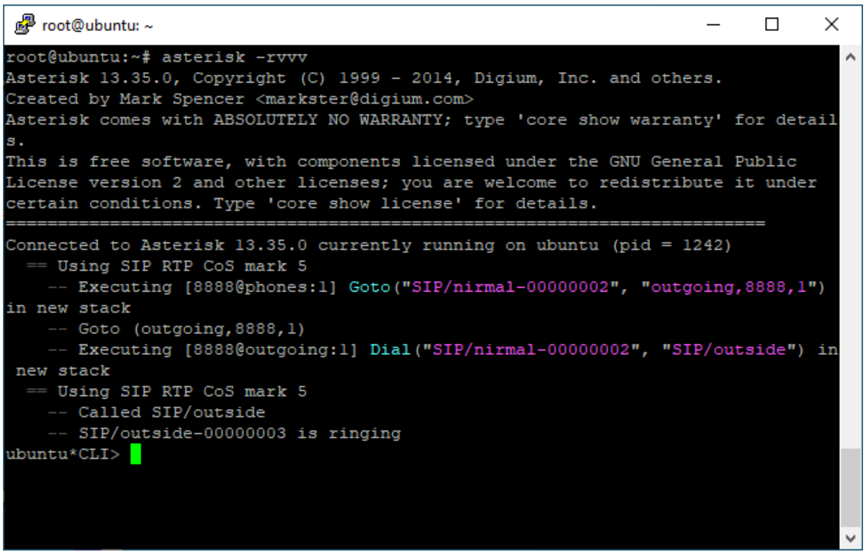
```
exten => 8888,1,Goto(outgoing,8888,1)
```




3. Go to asrerisk console , reload dialplan

```
# asterisk -rvvv
# dialplan reload
```

4. Dial 8888



## Regular Expressions

A regular expression (shortened as **regex** or **regexp** also referred to as rational expression) is a sequence of characters that define a search pattern. Usually such patterns are used by string-searching algorithms for "find" or "find and replace" operations on strings, or for input validation. It is a technique developed in theoretical computer science and formal language theory.

### Pattern Matching

With what we know so far, we need to write a separate extension for each telephone number. As the system expands, this leads to unwieldy and error-prone dialplans. Say that, for our example, we need numbers 100 to 109 to play the "hello world" sound file. Our extensions.conf would look like this:

```
[general]

[widgets]
exten => 100,1,Answer()
exten => 100,2,Playback(hello-world)
exten => 100,3,Hangup()

exten => 101,1,Answer()
exten => 101,2,Playback(hello-world)
exten => 101,3,Hangup()

exten => 102,1,Answer()
exten => 102,2,Playback(hello-world)
exten => 102,3,Hangup()
..
```

If we use a pattern, the same dialplan becomes instantly more compact and elegant:

```
[general]

[widgets]
exten => _10X,1,Answer()
exten => _10X,2,Playback(hello-world)
exten => _10X,3,Hangup()
```

The '_10X' extension describes the number range from 100 to 109.

### Syntax

Dialplan patterns always begin with the underscore (_) character:

```
exten => _Pattern, Priority, Application
```

### An Asterisk dialplan pattern can have the following elements:

### [abc]
The digits a, b and c. For example, to match 34, 37, and 38:

```
exten => _3[478],1,NoOp(Test)
```

**[a-b]**
Any digit in the range a to b. For example, to match any number between 31 and 35:

```
exten => _3[1-5],1,NoOp(Test)
```

(e.g. [25-8] is also acceptable for the digits 2,5,6,7,8)

**X**
Any digit from 0 to 9. For example, to match any number between 300 and 399:

```
exten => _3XX,1,NoOp(Test)
```

**Z**
Any digit from 1 to 9. For example, to match any number between 31 and 39:

```
exten => _3Z,1,NoOp(Test)
```

**N**
Any digit from 2 to 9. For example, to match any number between 32 and 39:

```
exten => _3N,1,NoOp(Test)
```

**.**
Any number of digits of any kind. For example, to match all numbers that begin with 011:

```
exten => _011.,1,NoOp(Test)
```

Warning: Don't use the '_.' pattern! This will also include special extensions such as i, t and h. Use _X. or _X if you need broad pattern matching.

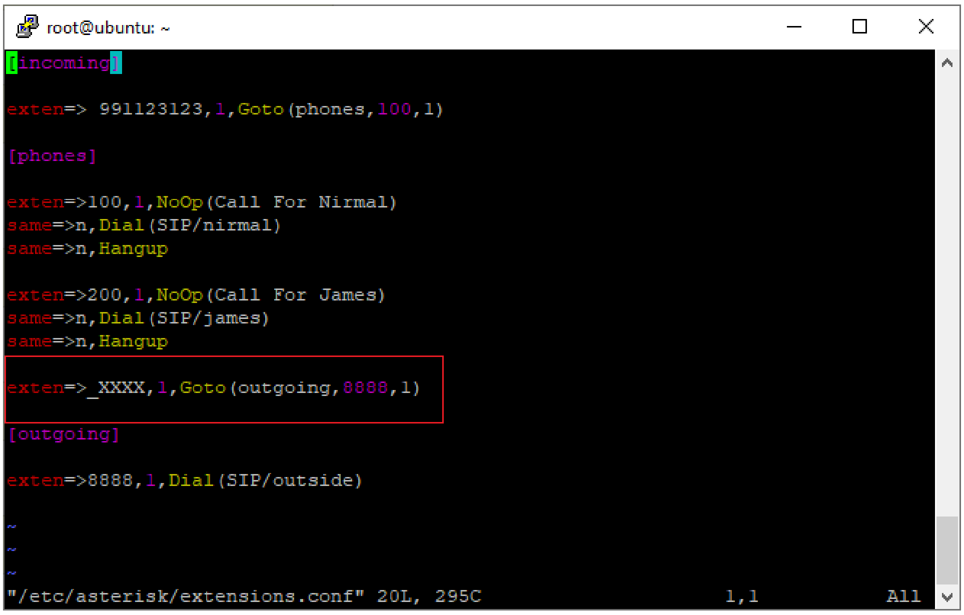
**!**
This special 'wildcard' character will match as soon as the number dialled is unambiguous; i.e. when the number being dialled cannot match any other extension in the context. Once a match is made, the outgoing line is picked up and dialing proceeds in real-time with direct feedback (this is known as 'overlap dialing').

**Implementing Regular Expression or Pattern Matching**

1. Go to dialpan config file :- extensions.conf

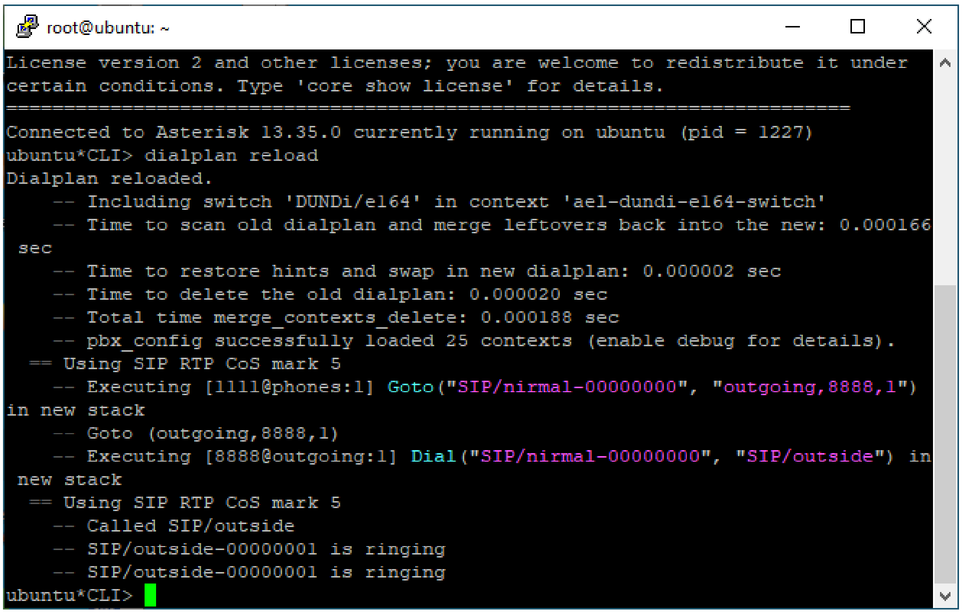
   #vi /etc/asterisk/extensions.conf

2. Edit extensions.conf ,Repalce 8888 with _XXXX



```
root@ubuntu: ~
[incoming]

exten=> 991123123,1,Goto(phones,100,1)

[phones]

exten=>100,1,NoOp(Call For Nirmal)
same=>n,Dial(SIP/nirmal)
same=>n,Hangup

exten=>200,1,NoOp(Call For James)
same=>n,Dial(SIP/james)
same=>n,Hangup

exten=>_XXXX,1,Goto(outgoing,8888,1)

[outgoing]

exten=>8888,1,Dial(SIP/outside)

~
~
~
"/etc/asterisk/extensions.conf" 20L, 295C                    1,1          All
```

5. Go to asrerisk console , reload dialplan and Dial any 4 digit number



```
root@ubuntu: ~
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1227)
ubuntu*CLI> dialplan reload
Dialplan reloaded.
    -- Including switch 'DUNDi/e164' in context 'ael-dundi-e164-switch'
    -- Time to scan old dialplan and merge leftovers back into the new: 0.000166
 sec
    -- Time to restore hints and swap in new dialplan: 0.000002 sec
    -- Time to delete the old dialplan: 0.000020 sec
    -- Total time merge_contexts_delete: 0.000188 sec
    -- pbx_config successfully loaded 25 contexts (enable debug for details).
  == Using SIP RTP CoS mark 5
    -- Executing [1111@phones:1] Goto("SIP/nirmal-00000000", "outgoing,8888,1")
in new stack
    -- Goto (outgoing,8888,1)
    -- Executing [8888@outgoing:1] Dial("SIP/nirmal-00000000", "SIP/outside") in
 new stack
  == Using SIP RTP CoS mark 5
    -- Called SIP/outside
    -- SIP/outside-00000001 is ringing
    -- SIP/outside-00000001 is ringing
ubuntu*CLI>
```

## Asterisk Variable

A variable is a placeholder for an actual value. Exactly what that value is depends on the kind of variable. In Asterisk, variables can contain numbers, letters and strings (sequences of letters and numbers). Variables are useful because they let us create rules for call flow that apply in changing circumstances and make it easier to accommodate future changes in the telephone application or system.

In Asterisk, variables have varying scope. There are local variables (called channel variables in Asterisk), which can only set values for the current, active channel, and global variables, which set values for all channels. We should already be familiar with some of the variables Asterisk sets from our exposure to them as configuration parameters in the Asterisk configuration files (such as **sip.conf**, for example). We also have the freedom to define our own variables and use them in configuration files

### Expanding variables in an extension

The value of a variable can be obtained using the syntax **${VARIABLENAME}**. There are variables that are automatically set by Asterisk. For example, the called number is always stored in the Asterisk system variable **${EXTEN}**. Using patterns and variables, it is often possible to dramatically compress a long dialplan.

Before:

```
exten => 100,1,Dial(SIP/100)
exten => 101,1,Dial(SIP/101)
exten => 102,1,Dial(SIP/102)
exten => 103,1,Dial(SIP/103)
exten => 104,1,Dial(SIP/104)
exten => 105,1,Dial(SIP/105)
exten => 106,1,Dial(SIP/106)
exten => 107,1,Dial(SIP/107)
exten => 108,1,Dial(SIP/108)
exten => 109,1,Dial(SIP/109)
```

After:

```
exten => _10X,1,Dial(SIP/${EXTEN})
```

### General considerations

Variable names needn't be in all uppercase as in our examples, nor are user-defined variables case-sensitive. It is a good idea to use uppercase variable names nonetheless because it makes the variables easier to identify and the dialplan code easier to read. The primary disadvantage of this is that it means you cannot distinguish variable names based on case. For example, ${FOO} is considered the same as ${foo}.

**Important:** Asterisk system variables such as ${EXTEN} must always be uppercase.

### String variables

String variables (meaning variables that contain text and not numbers) should be defined using double quotes, though Asterisk will still accept them without double quotes - the following two entries are functionally identical:

```
exten => 1234,1,Set(FRUIT=Apple)
exten => 1234,2,Set(FRUIT="Apple")
```

If the string contains commas or spaces, you must use double quotes:

```
exten => 1234,1,Set(FRUITTYPES="Apple, Pear, etc.")
```

This is why it is a good idea to get into the habit of using them for any string variables you define.

### Reserved characters

Sometimes a variable will contain reserved characters (characters that have special functions and are interpreted differently). For example, if you want to variable to contain the underscore character ("_") you must use an "escape" character to tell the dialplan interpreter that it should ignore the reserved character. The following characters must be escaped when used in a variable:

```
[ ] $ " \
```

The escape character in extensions.conf is "\" (backslash):

Example: `exten => 1234,1,Set(AMOUNT="\$10.00")`

Similarly, if you want to use the backslash character in a variable, you must escape it:

```
exten => 1234,1,Set(ROOMNUMBER="48\\10")
```

### Integers

If a variable contains an integer, it can have no more than 18 digits. Anything larger will cause an error which will be recorded in the log file.

### Defining global variables in extensions.conf

Global variables are defined at the beginning of extensions.conf. You must place them in the special [globals] context, which follows [general]. Example:

```
[general]

[globals]
      RINGTIME=90

[from-intern]
      exten => _XXX,1,Dial(SIP/${EXTEN},${RINGTIME})
      exten => _XXX,n,VoiceMail(${EXTEN})
```

**Defining variables with Set()**

Set() is used to define a variable inside an extension.

Syntax

```
Set(<variable1>=<value1>[,<variable2>=<value2>][,<option>])
```

Setting option g makes the variable global; without it, the variable is treated as a local channel variable.

Example:

```
; Set a global variable:
exten => 10,1,Set(RINGTIME=90,g)

; Set a local channel variable:
exten => 10,2,Set(FAVORITEFRUIT="Apple")

; Set two channel variables at once:
exten => 10,3,Set(VAR1=10,VAR2=23)

; Print variables to the CLI
exten => 10,4,NoOp(RINGTIME = ${RINGTIME})
exten => 10,5,NoOp(FAVORITEFRUIT = ${FAVORITEFRUIT})
exten => 10,6,NoOp(VAR1 = ${VAR1})
exten => 10,7,NoOp(VAR2 = ${VAR2})
```

**Inheritance of channel variables**

If new channels are spawned while a conversation is in progress, they will have their own channel variables.

**Single-level inheritance**

Sometimes you want to have a channel variable persist into the spawned channel. You can do this by prefixing the variable with an "_" (underscore) character. When the variable is inherited by the spawned channel, Asterisk automatically removes the prefix. This ensures that the variable is inherited only once.

Example:

```
exten => 1234,1,Set(_CAKE="Marble cake")
```

**Multi-level inheritance**

If you need unlimited inheritance of a channel variable, you can do this by prefixing the variable with two "_" (underscore) characters. Variables prefixed in this way will always be inherited by spawned channels.

Asterisk makes no distinction between variable names that are preceded with an underscore and those that are not. In the example below, a variable with multi-level inheritance ("__CAKE") is rendered uninheritable by the subsequent entry:

```
exten => 1234,1,Set(__CAKE="Marble cake")
exten => 1234,n,Set(CAKE="Marble cake")
```

Example:

```
exten => 1234,1,Set(__CAKE="Sponge cake")
```

When calling an inherited variable, it doesn't matter if it is called with a prefix or not. These entries will give the same output in the CLI:

```
exten => 1234,1,NoOp(${__CAKE})
exten => 1234,n,NoOp(${CAKE})
```

**System channel variables**

The following list describes the more important system channel variables. These variables may be read but not overwritten by entries in extensions.conf, as they are pre-defined by Asterisk.

Some of the "variables" described here are not really variables but in fact built-in functions. In practice, they often play a similar role, so they are listed here for convenience.

**${ANSWEREDTIME}**: The total elapsed time for the active connection (in other words, the number of seconds since the conversation started).
**${BLINDTRANSFER}**: The name of the channel on the other side of a blind transfer.
**${CHANNEL}**: Name of the current channel.
**${CONTEXT}**: Name of the current context.
${EPOCH}: Current Unix time (total number of seconds elapsed since the beginning of the Unix "epoch", which began at midnight UTC, January 1st, 1970)
**${EXTEN}**: Currently dialed extension.
**${ENV(VARIABLENAME)}**: Environment variable VARIABLENAME
**${HANGUPCAUSE}**: Cause of connection hang-up.
**${INVALID_EXTEN}**: Used in the i extension and contains the dialed extension.
**${PRIORITY}:** Current priority in the current extension.
**${TRANSFER_CONTEXT}:** Context of a transferred call.
**${UNIQUEID}:** The unique ID for the current connection.
**${SYSTEMNAME}:** The system name as defined by **systemname** in **/etc/asterisk/asterisk.conf**.

**Manipulating variables**

Variables are most useful when we can change their contents at execution time. This gives us the flexibility to impart complex and powerful behavior to our Asterisk system.

**Substring**

In general, a string consists of a sequence of individual characters. The size of a string is determined by the number of characters contained in it.

For example, the string "apple tree" has 10 characters (we must include the space). Any string can be broken into substrings. For example, "apple", "tree", "app" and "le tre" are all valid substrings of "apple tree". In theory, a string can be of any length; this entire book could be contained in a single string, though it would be impractical. Manipulation of strings is an important technique in programming applications. Asterisk lets you manipulate strings and substrings using the : (colon) character. Using the : character, you can extract a specified portion of an existing string variable.

**Syntax**

```
${VARIABLENAME[:start[:length]]}
```

Examples

Many telephone systems require that a prefix digit be dialed in order to get an outside line (In North America, this is usually "9"). The target number, however, cannot include this prefix digit. If we dial 9-1-202-7075000, we can store the actual outside number in the ${OUTGOINGNUMBER} using the following dialplan entry.

```
exten => _0X.,1,Set(OUTGOINGNUMBER=${EXTEN:1})
```

If the length option is omitted, the rest of the string is taken automatically.
What if we only need the last seven digits of the dialed number? I this case we use a negative number for the start parameter. The following entry would store 7075000 from our example above in the variable ${LOCALNUMBER}.[14]

```
exten => _0X.,1,Set(LOCALNUMBER=${EXTEN:-7})
```

We can also capture just the area code:

```
exten => _0X.,1,Set(AREACODE=${EXTEN:2:3})
```

Here, then, is how we might extract useful information from a dialed number:

```
exten => _9X.,1,Set(AREACODE=${EXTEN:2:3})
exten => _9X.,n,Set(LOCALNUMBER=${EXTEN:5})
```

${EXTEN} Used to store dialed number

${EXTEN:1} Removing unwanted number Sterting position and -1 for ending (e.g:- 01111 result is 1111

```
${EXTEN:1:3} (${EXTEN:<offset>:<length>}
```

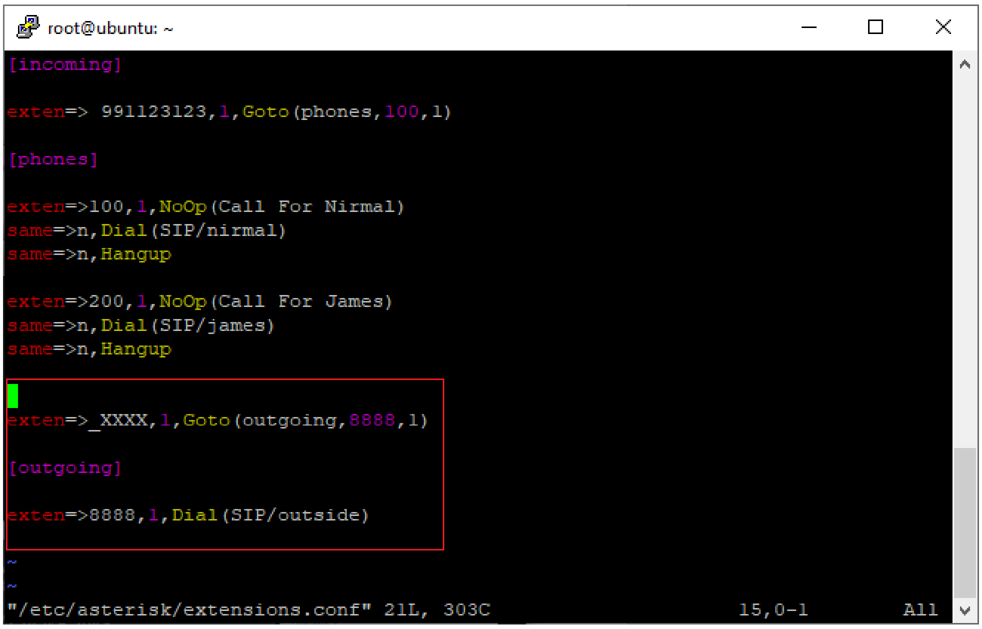
Result :- 111

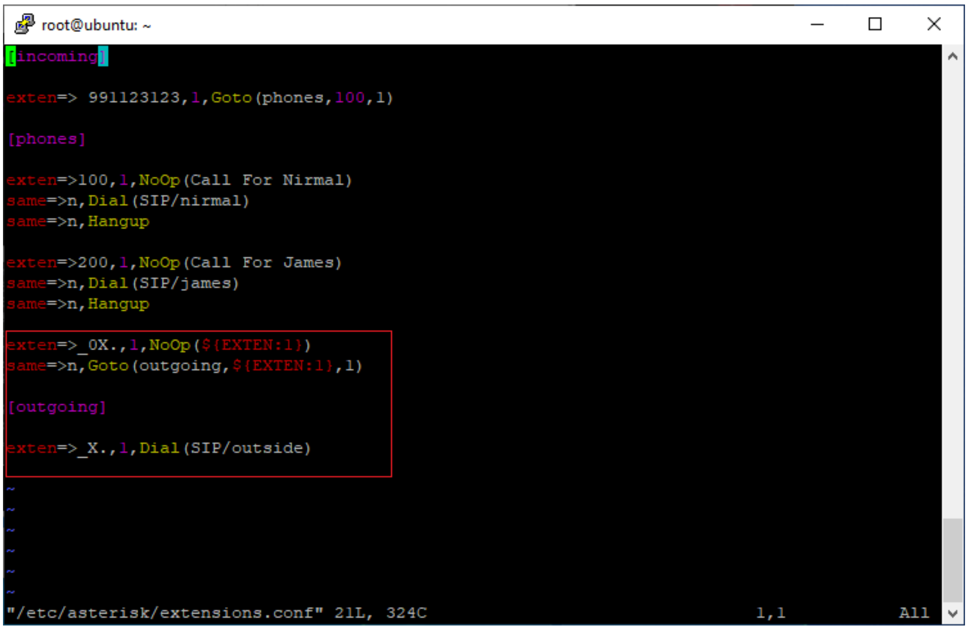
### Implementing variables

1. Go to dialpan config file :- extensions.conf

   `#vi /etc/asterisk/extensions.conf`

2. Replace hardcoded value with asterisk variable



3. Reload dial plan and Test (dial any number start with zero for e.g. 022223)

```
root@ubuntu: ~
root@ubuntu:~# vi /etc/asterisk/extensions.conf
root@ubuntu:~# asterisk -rvvv
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1227)
ubuntu*CLI> dialplan reload
Dialplan reloaded.
    -- Including switch 'DUNDi/e164' in context 'ael-dundi-e164-switch'
    -- Time to scan old dialplan and merge leftovers back into the new: 0.000134 sec
    -- Time to restore hints and swap in new dialplan: 0.000002 sec
    -- Time to delete the old dialplan: 0.000022 sec
    -- Total time merge_contexts_delete: 0.000158 sec
    -- pbx_config successfully loaded 25 contexts (enable debug for details).
  == Using SIP RTP CoS mark 5
    -- Executing [022223@phones:1] NoOp("SIP/nirmal-00000006", "22223") in new stack
    -- Executing [022223@phones:2] Goto("SIP/nirmal-00000006", "outgoing,22223,1") in new s
tack
    -- Goto (outgoing,22223,1)
    -- Executing [22223@outgoing:1] Dial("SIP/nirmal-00000006", "SIP/outside") in new stack
  == Using SIP RTP CoS mark 5
    -- Called SIP/outside
    -- SIP/outside-00000007 is ringing
ubuntu*CLI>
```

## Time Conditions

Setting up time conditions for incoming calls

1. Go to dialpan config file :- extensions.conf

   `#vi /etc/asterisk/extensions.conf`

2. Edit [incoming] context

```
root@ubuntu: ~
[incoming]

exten=>991123123,1,GotoIfTime(8:00-9:00,mon-fri,*,*?phones,100,1)
exten=>991123123,n,Playback(tt-monkeys)
exten=>991123123,n,Hangup


[phones]

exten=>100,1,NoOp(Call For Nirmal)
same=>n,Dial(SIP/nirmal)
same=>n,Hangup

exten=>200,1,NoOp(Call For James)
same=>n,Dial(SIP/james)
same=>n,Hangup

exten=>_0X.,1,NoOp(${EXTEN:1})
same=>n,Goto(outgoing,${EXTEN:1},1)

[outgoing]

exten=>_X.,1,Dial(SIP/outside)
"/etc/asterisk/extensions.conf" 24L, 418C                    1,1          Top
```

3. Dial 991123123

```
root@ubuntu: ~
root@ubuntu:~# asterisk -rvvv
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1227)
ubuntu*CLI> dialplan reload
Dialplan reloaded.
    -- Including switch 'DUNDi/e164' in context 'ael-dundi-e164-switch'
    -- Time to scan old dialplan and merge leftovers back into the new: 0.000120 sec
    -- Time to restore hints and swap in new dialplan: 0.000002 sec
    -- Time to delete the old dialplan: 0.000022 sec
    -- Total time merge_contexts_delete: 0.000144 sec
    -- pbx_config successfully loaded 25 contexts (enable debug for details).
  == Using SIP RTP CoS mark 5
    -- Executing [991123123@incoming:1] GotoIfTime("SIP/outside-0000000e", "8:00-9:00,mon-f
ri,*,*?phones,100,1") in new stack
    -- Executing [991123123@incoming:2] Playback("SIP/outside-0000000e", "tt-monkeys") in n
ew stack
    -- <SIP/outside-0000000e> Playing 'tt-monkeys.gsm' (language 'en')
    -- Executing [991123123@incoming:3] Hangup("SIP/outside-0000000e", "") in new stack
  == Spawn extension (incoming, 991123123, 3) exited non-zero on 'SIP/outside-0000000e'
ubuntu*CLI>
```

If Success

```
    -- Executing [991123123@incoming:1] GotoIfTime("SIP/outside-00000010", "8:00-17:00,mon-
fri,*,*?phones,100,1") in new stack
    -- Goto (phones,100,1)
    -- Executing [100@phones:1] NoOp("SIP/outside-00000010", "Call For Nirmal") in new stac
k
    -- Executing [100@phones:2] Dial("SIP/outside-00000010", "SIP/nirmal") in new stack
  == Using SIP RTP CoS mark 5
    -- Called SIP/nirmal
    -- SIP/nirmal-00000011 is ringing
ubuntu*CLI>
```

## Subroutines

Subroutines in Asterisk are defined similarly to standard dialplan contexts and are referred to as **Macros** and **Gosub**. They are invoked via the Macro and Gosub applications.

Macro is a dialplan application that facilitates code-reuse within the dialplan. That is, a macro, once defined can be called from almost anywhere else within the dialplan using the Macro application or else via flags and arguments for other applications that allow calling macros.

Gosub allows you to execute a specific block (context or section) of dialplan as well as pass and return information via arguments to/from the scope of the block. Whereas Macro has issues with nesting, Gosub does not and Gosub should be used wherever you would have used a Macro.

## Implemeting GoSub

1. Go to dialpan config file :- extensions.conf

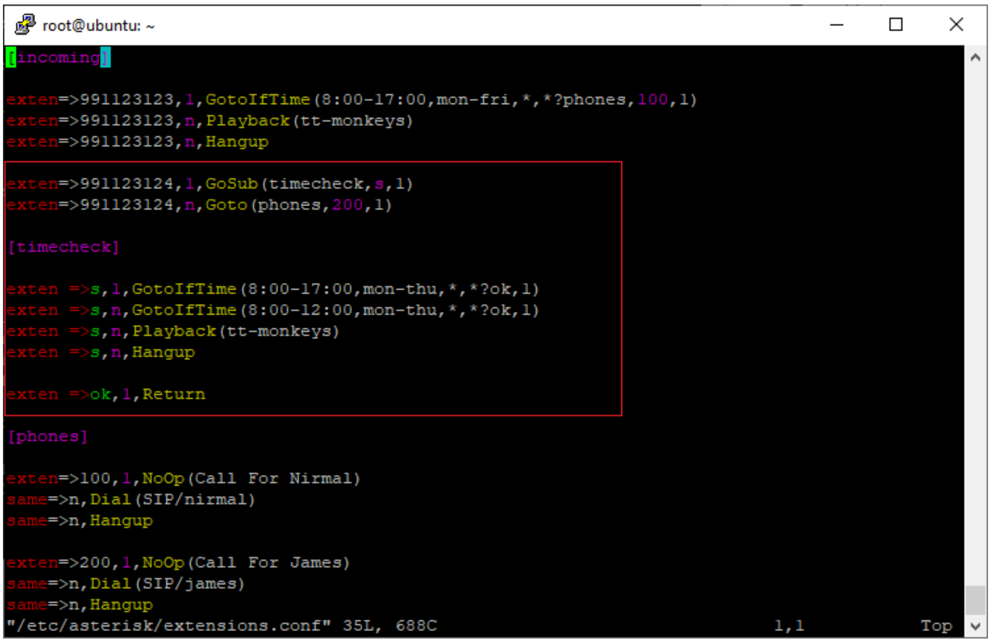
```
#vi /etc/asterisk/extensions.conf
```

2. Add New context called [timecheck] (Subroutine name)

```
[timecheck]

        exten =>s,1,GotoIfTime(8:00-17:00,mon-thu,*,*?ok,1)
        exten =>s,n,GotoIfTime(8:00-12:00,mon-thu,*,*?ok,1)
        exten =>s,1,Playback(tt-monkeys)
        exten =>s,1,Hangup

        exten =>ok,1,Return
```

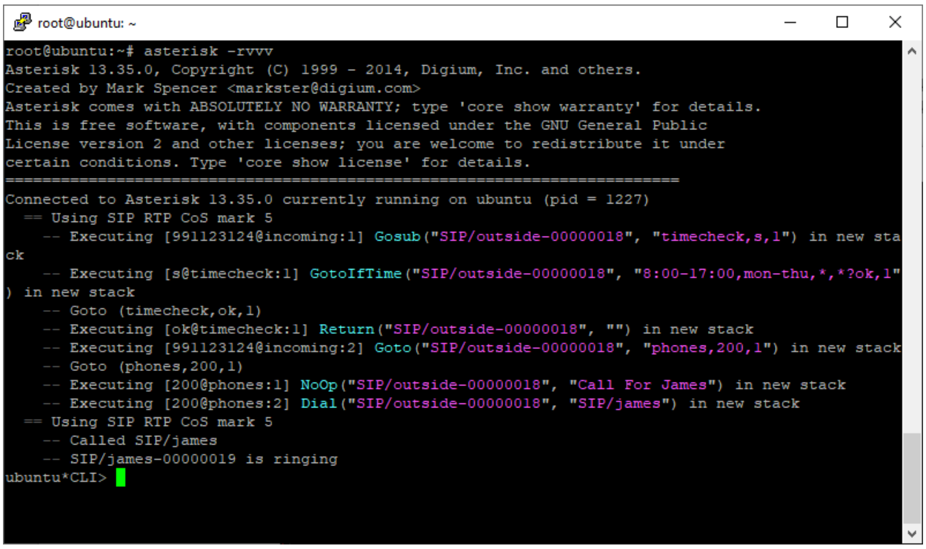
In `[incoming]` Context

```
        exten=>991123124,1,GoSub(timecheck,s,1)
        exten=>991123124,n,GoTo(phones,200,1)
```



```
root@ubuntu: ~
[incoming]

exten=>991123123,1,GotoIfTime(8:00-17:00,mon-fri,*,*?phones,100,1)
exten=>991123123,n,Playback(tt-monkeys)
exten=>991123123,n,Hangup

exten=>991123124,1,GoSub(timecheck,s,1)
exten=>991123124,n,Goto(phones,200,1)

[timecheck]

exten =>s,1,GotoIfTime(8:00-17:00,mon-thu,*,*?ok,1)
exten =>s,n,GotoIfTime(8:00-12:00,mon-thu,*,*?ok,1)
exten =>s,n,Playback(tt-monkeys)
exten =>s,n,Hangup

exten =>ok,1,Return

[phones]

exten=>100,1,NoOp(Call For Nirmal)
same=>n,Dial(SIP/nirmal)
same=>n,Hangup

exten=>200,1,NoOp(Call For James)
same=>n,Dial(SIP/james)
same=>n,Hangup
"/etc/asterisk/extensions.conf" 35L, 688C                    1,1           Top
```

3. Reload dial plan and Test (Dial 991123124)



```
root@ubuntu: ~
root@ubuntu:~# asterisk -rvvv
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1227)
  == Using SIP RTP CoS mark 5
    -- Executing [991123124@incoming:1] Gosub("SIP/outside-00000018", "timecheck,s,1") in new sta
ck
    -- Executing [s@timecheck:1] GotoIfTime("SIP/outside-00000018", "8:00-17:00,mon-thu,*,*?ok,1"
) in new stack
    -- Goto (timecheck,ok,1)
    -- Executing [ok@timecheck:1] Return("SIP/outside-00000018", "") in new stack
    -- Executing [991123124@incoming:2] Goto("SIP/outside-00000018", "phones,200,1") in new stack
    -- Goto (phones,200,1)
    -- Executing [200@phones:1] NoOp("SIP/outside-00000018", "Call For James") in new stack
    -- Executing [200@phones:2] Dial("SIP/outside-00000018", "SIP/james") in new stack
  == Using SIP RTP CoS mark 5
    -- Called SIP/james
    -- SIP/james-00000019 is ringing
ubuntu*CLI>
```

## Voicemail

This is different from an Interactive Voice Response System (IVR). There are two voicemail applications we can use in the dialplan (extensions.conf):

- **VoiceMail():** This application sends a caller to the voicemail system, where she will be asked to leave a message.
- **VoiceMailMain():** This application lets recipients check their voice messages and record new voicemail prompts.

### VoiceMail()

The caller is prompted to leave a voice message. The VoiceMail() command is always called from the dialplan (extensions.conf). For example:

```
exten => 2000,2,VoiceMail(2000,u)
```

Syntax

```
VoiceMail(mailbox[@context][,u|b|s])
```

**Mailbox**: This is the mailbox number. This does not have to be the same as the extension the caller dialed; nevertheless, this is a sensible practice, particularly in larger installations.

**@context:** Mailboxes may be implemented in a specific context. If no context is provided, the [default] context is used.

If the caller presses "0" while listening to the prompt, the application will jump to extension "o" (the small letter o) in the specified context.

If the caller presses "*" while listening to the prompt, the application will jump to extension "a" (the small letter a) in the specified context

**[u|b|s]**

**u**: causes the "unavailable" message to be played. The pathname for this message is /var/lib/asterisk/sounds/vm-isunavail.gsm

**b**: causes the "busy" to be played. The pathname for this message is /var/lib/asterisk/sounds/vm-rec-busy.gsm.

**s**: suppresses playback of the "unavailable" or "busy" notifications, plays a beep, and begins recording.

If there is no mailbox configured in voicemail.conf for the given number but there is a n+101 priority, Asterisk jumps to this priority and continues executing there.

**VoiceMailMain()**

Let's users listen to their voicemail messages and record prompts. The VoiceMailMain() command is always called from the dialplan (extensions.conf). For example:

```
exten => 300,1,VoiceMailMain()
```

Syntax

```
VoiceMailMain([mailbox][@context][,s|p|g(#)])
```

**mailbox**
This is the mailbox number. If no mailbox number is provided, Asterisk prompts for it.

**@context**
specifies the voicemail context (in voicemail.conf) for the mailbox.

**[s|p|g#]**

**s:**Disables the password requirement.
**p:** The user is asked for a mailbox number. The number entered is attached as a suffix to the contents of [mailbox]; for example, if the user enters 123, [mailbox]123 is called. This lets you easily configure mailbox groups.
**g(#):** Adjusts the gain (in decibels) when recording voicemail prompts.

**IVR**

A complete description of the voice menus for VoiceMailMain() is difficult because they depend on the installed prompts. The main functions are described below.

- 1 Play messages
    - 3 Advanced options
        - 1 Reply
        - 2 Call back
        - 3 Envelope
        - 4 Outgoing call
    - 4 Play previous message
    - 5 Repeat the current message
    - 6 Play the next message
    - 7 Delete the current message
    - 8 Forward the message to another mailbox
    - 9 Save the message in a folder

| | |
|---|---|
| | * Help; during message playback, rewind<br># Exit; during message playback, skip forward |
| 2 | Change folders |
| 0 | Mailbox options |
| | 1 Record your unavailable message<br>2 Record your busy message<br>3 Record your name<br>4 Record your temporary message<br>Recording options<br>1 Accept<br>2 Review<br>3 Re-record |
| * | Help |
| # | Exit |

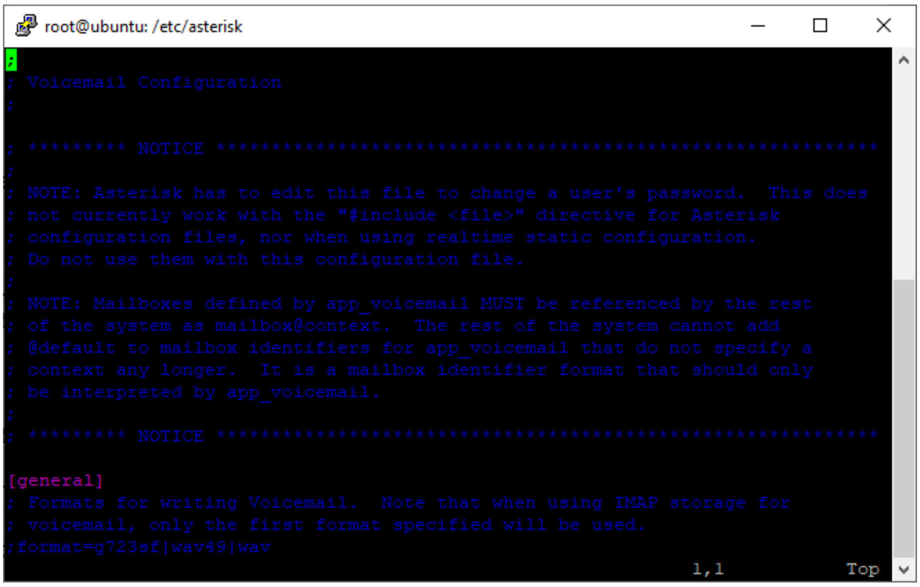
## Voice mail configuration - voicemail.conf

1. Copy voicemail.conf

```
# cd /etc/asterisk
#cp voicemail.conf voicemail.conf.org
```

2. Open voicemail.conf file and remove unwanted lines

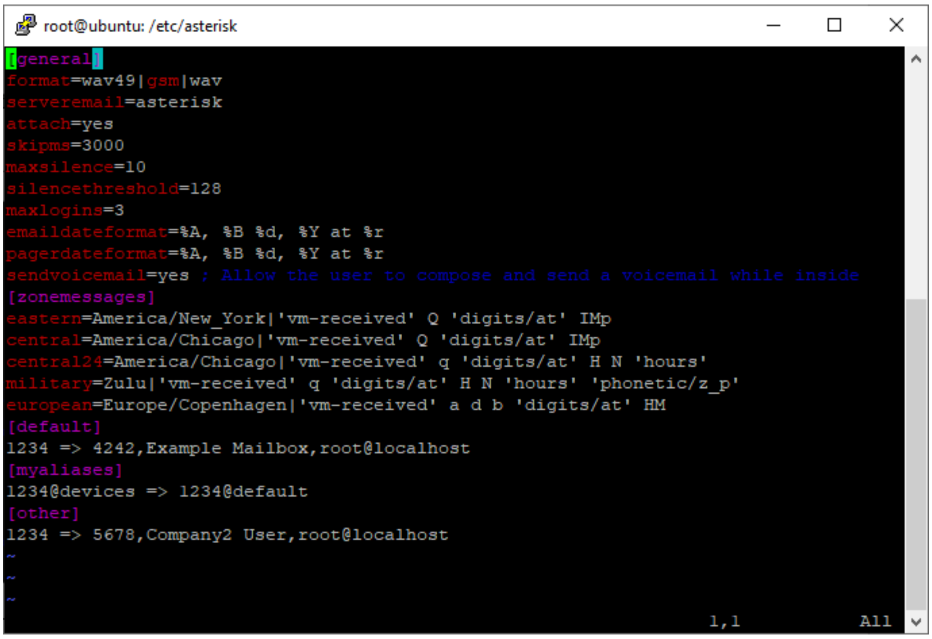
```
#vi voicemail.conf
```



root@ubuntu: /etc/asterisk

```
;
; Voicemail Configuration
;

; ********* NOTICE ************************************************************
;
; NOTE: Asterisk has to edit this file to change a user's password.  This does
; not currently work with the "#include <file>" directive for Asterisk
; configuration files, nor when using realtime static configuration.
; Do not use them with this configuration file.
;
; NOTE: Mailboxes defined by app_voicemail MUST be referenced by the rest
; of the system as mailbox@context.  The rest of the system cannot add
; @default to mailbox identifiers for app_voicemail that do not specify a
; context any longer.  It is a mailbox identifier format that should only
; be interpreted by app_voicemail.
;
; ********* NOTICE ************************************************************

[general]
; Formats for writing Voicemail.  Note that when using IMAP storage for
; voicemail, only the first format specified will be used.
;format=g723sf|wav49|wav
                                                            1,1           Top
```

Use (:g/^\s*;/d and :g/^$/d)



root@ubuntu: /etc/asterisk

```
[general]
format=wav49|gsm|wav
serveremail=asterisk
attach=yes
skipms=3000
maxsilence=10
silencethreshold=128
maxlogins=3
emaildateformat=%A, %B %d, %Y at %r
pagerdateformat=%A, %B %d, %Y at %r
sendvoicemail=yes ; Allow the user to compose and send a voicemail while inside
[zonemessages]
eastern=America/New_York|'vm-received' Q 'digits/at' IMp
central=America/Chicago|'vm-received' Q 'digits/at' IMp
central24=America/Chicago|'vm-received' q 'digits/at' H N 'hours'
military=Zulu|'vm-received' q 'digits/at' H N 'hours' 'phonetic/z_p'
european=Europe/Copenhagen|'vm-received' a d b 'digits/at' HM
[default]
1234 => 4242,Example Mailbox,root@localhost
[myaliases]
1234@devices => 1234@default
[other]
1234 => 5678,Company2 User,root@localhost
~
~
~
                                                            1,1           All
```

3. Add voice mail context in [default] (check extension number in extensions.conf)

```
100 => 1234,james,james@example.com
200 => 1234,nirmal,nirmal@example.com
```

```
root@ubuntu: ~
[general]
format=wav49|gsm|wav
serveremail=asterisk
attach=yes
skipms=3000
maxsilence=10
silencethreshold=128
maxlogins=3
emaildateformat=%A, %B %d, %Y at %r
pagerdateformat=%A, %B %d, %Y at %r
sendvoicemail=yes ; Allow the user to compose and send a voicemail while inside
[zonemessages]
eastern=America/New_York|'vm-received' Q 'digits/at' IMp
central=America/Chicago|'vm-received' Q 'digits/at' IMp
central24=America/Chicago|'vm-received' q 'digits/at' H N 'hours'
military=Zulu|'vm-received' q 'digits/at' H N 'hours' 'phonetic/z_p'
european=Europe/Copenhagen|'vm-received' a d b 'digits/at' HM
[default]
200 => 1234,james,james@example.com
100 => 1234,nirmal,nirmal@example.com

~
~
"/etc/asterisk/voicemail.conf" 21L, 685C                      1,1           All
```

4. Go to asterisk console and reload voice mail

# voicemail reload

```
root@ubuntu: /etc/asterisk
root@ubuntu:/etc/asterisk# asterisk -rvvv
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1238)
ubuntu*CLI> voicemail reload
Reloading voicemail configuration...
ubuntu*CLI>
```

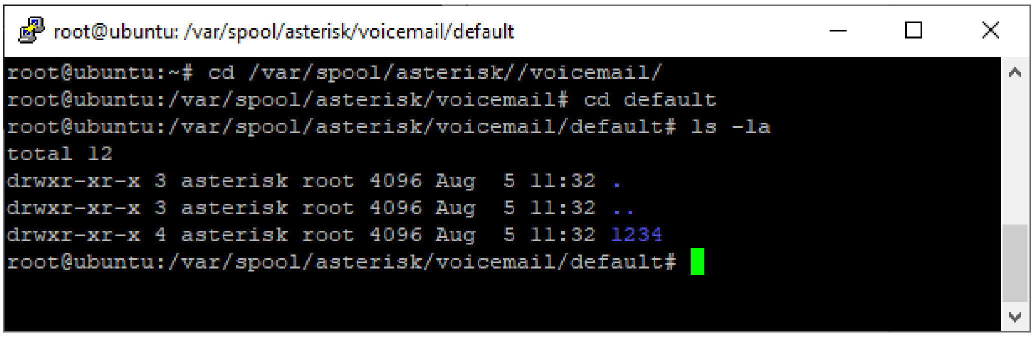
To check

#voicemail show users

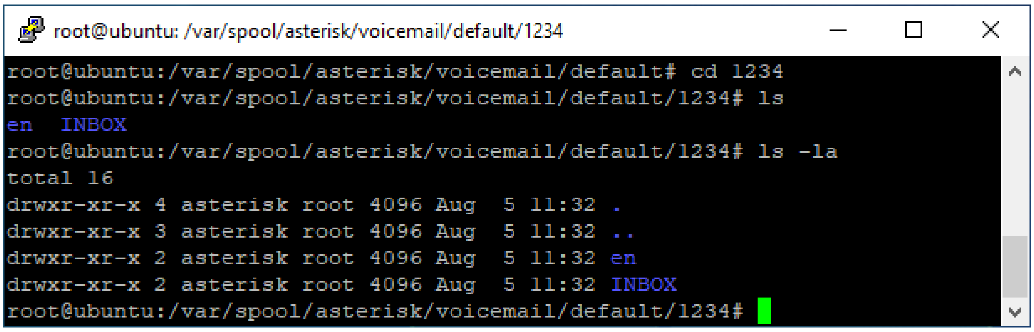
```
root@ubuntu: /etc/asterisk
root@ubuntu:/etc/asterisk# asterisk -rvvv
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1238)
ubuntu*CLI> voicemail reload
Reloading voicemail configuration...
ubuntu*CLI> voicemail show users
Context    Mbox  User                     Zone       NewMsg
default    200   james                                    0
default    100   nirmal                                   0
2 voicemail users configured.
ubuntu*CLI>
```

5. To check voice mail. All files are stored in /var/spool/asterisk/voicemail/

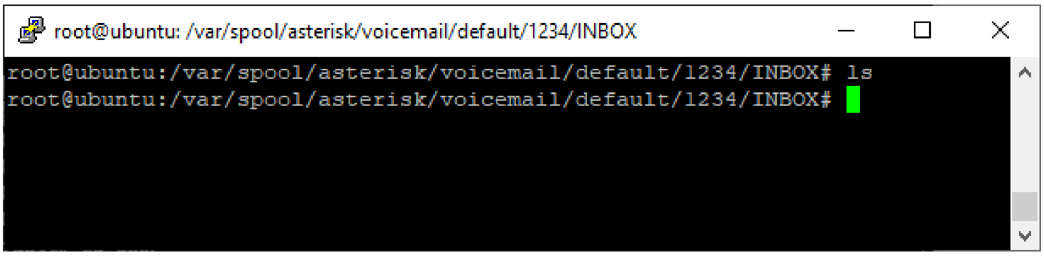
   # cd /var/spool/asterisk/voicemail/
   #cd default
   #ls -la




If you are first time user you will not see any file or inbox




To see Message INBOX



## Voicemail Dialplan Configuration

1. Open /etc/asterisk/extensions.conf

```
# vi /etc/asterisk/extensions.conf
```

2. Edit and add Voicemail Application (see below screen) in [phones] context or user context

```
root@ubuntu: ~

exten =>s,1,GotoIfTime(8:00-17:00,mon-thu,*,*?ok,1)
exten =>s,n,GotoIfTime(8:00-12:00,mon-thu,*,*?ok,1)
exten =>s,n,Playback(tt-monkeys)
exten =>s,n,Hangup

exten =>ok,1,Return

[phones]

exten=>100,1,NoOp(Call For Nirmal)
same=>n,Dial(SIP/nirmal,5)
same=>n,VoiceMail(${EXTEN})
same=>n,Hangup

exten=>200,1,NoOp(Call For James)
same=>n,Dial(SIP/james,10)
same=>n,VoiceMail(${EXTEN})
same=>n,Hangup

exten=>_0X.,1,NoOp(${EXTEN:1})
same=>n,Goto(outgoing,${EXTEN:1},1)

[outgoing]

exten=>_X.,1,Dial(SIP/outside)

                                                    37,0-1        Bot
```

3. Go to Asterisk console, reload dialplan and Dial 100

```
#asterisk -rvvv
#dialplan reload
```

```
root@ubuntu: ~
root@ubuntu:~# asterisk -rvvv
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1238)
  == Using SIP RTP CoS mark 5
    -- Executing [100@phones:1] NoOp("SIP/james-00000003", "Call For Nirmal") in new stack
    -- Executing [100@phones:2] Dial("SIP/james-00000003", "SIP/nirmal,5") in new stack
  == Using SIP RTP CoS mark 5
    -- Called SIP/nirmal
    -- SIP/nirmal-00000004 is ringing
    -- Nobody picked up in 5000 ms
    -- Executing [100@phones:3] VoiceMail("SIP/james-00000003", "100") in new stack
    -- <SIP/james-00000003> Playing 'vm-intro.gsm' (language 'en')
    -- <SIP/james-00000003> Playing 'beep.gsm' (language 'en')
    -- Recording the message
    -- x=0, open writing:  /var/spool/asterisk/voicemail/default/100/tmp/i99sVr format: wav49, 0x7f9a80137
48
    -- x=1, open writing:  /var/spool/asterisk/voicemail/default/100/tmp/i99sVr format: gsm, 0x7f9a8012f98
    -- x=2, open writing:  /var/spool/asterisk/voicemail/default/100/tmp/i99sVr format: wav, 0x7f9a8013bf8
    -- Recording automatically stopped after a silence of 10 seconds
    -- <SIP/james-00000003> Playing 'auth-thankyou.gsm' (language 'en')
    -- Executing [100@phones:4] Hangup("SIP/james-00000003", "") in new stack
  == Spawn extension (phones, 100, 4) exited non-zero on 'SIP/james-00000003'
ubuntu*CLI>
```

**To listen mailbox**

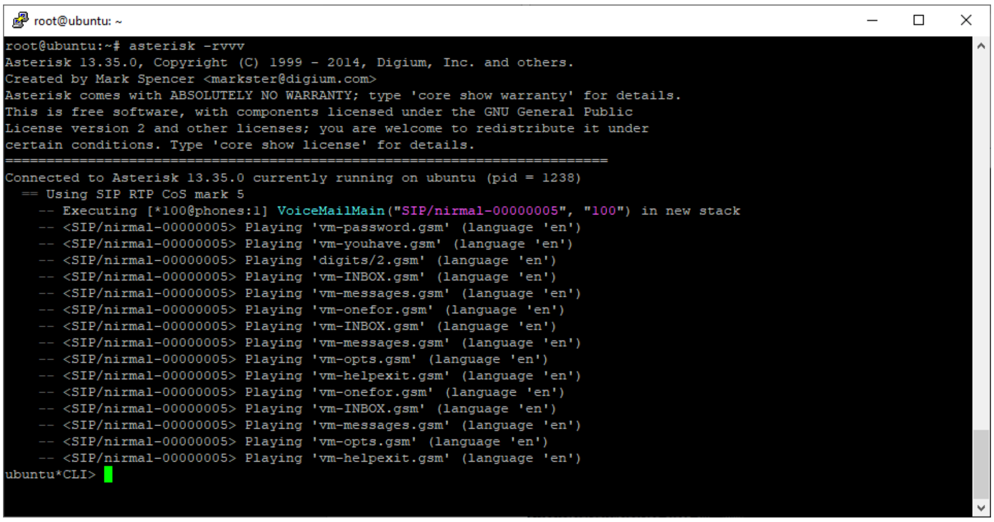
1. Open /etc/asterisk/extensions.conf

```
# vi /etc/asterisk/extensions.conf
```

2. Add context in [phone] context sections

```
exten => *100,1,VoiceMailMain(100)
same => n, Hangup
```


3. Open asterisk console, reload dialplan and dial *100 (password 1234)



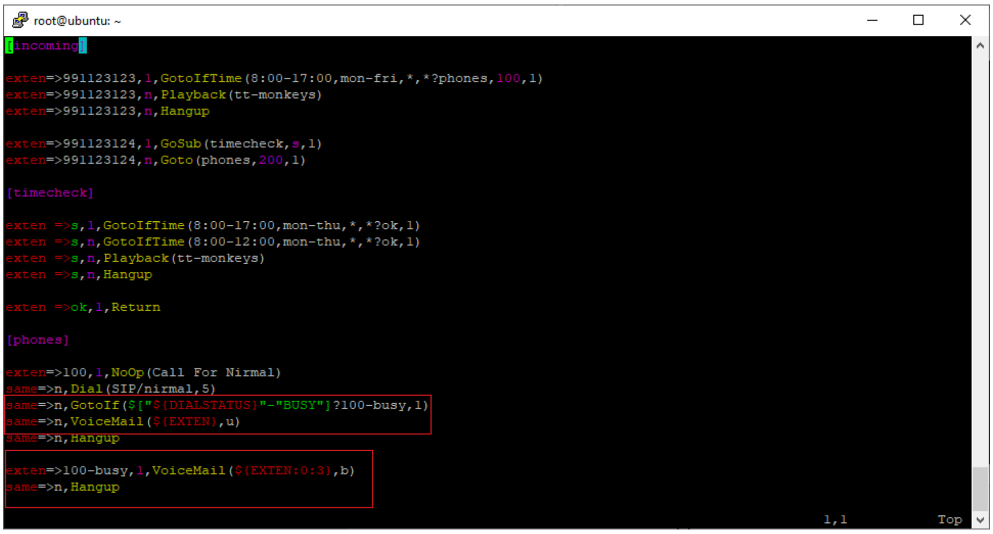
### Voicemail Greetings- Playing Busy/ Unavailable Messages

1. Configure mail box and Record Messages: Call your mail system, enter password and follow the instruction and record messages. (for e.g. dial 100)

   File location: `/var/spool/asterisk/voicemail/default/100/INBOX`

2. Edit /etc/asterisk/extensions.conf, add code (see below screen)

   Check **dial** application in asterisk console and add parameter in user mailbox




   Stimulate busy add bellow code in context


3. Go asterisk console and reload dialplan, dial 100

```
root@ubuntu: ~
    -- Time to restore hints and swap in new dialplan: 0.000002 sec
    -- Time to delete the old dialplan: 0.000024 sec
    -- Total time merge_contexts_delete: 0.000144 sec
    -- pbx_config successfully loaded 26 contexts (enable debug for details).
  == Using SIP RTP CoS mark 5
    -- Executing [100@phones:1] NoOp("SIP/james-00000006", "Call For Nirmal") in new stack
    -- Executing [100@phones:2] Dial("SIP/james-00000006", "SIP/nirmal,5") in new stack
  == Using SIP RTP CoS mark 5
    -- Called SIP/nirmal
    -- SIP/nirmal-00000007 is ringing
    -- Nobody picked up in 5000 ms
[Aug 18 14:16:00] WARNING[3295][C-00000008]: ast_expr2.y:1340 op_minus: non-numeric argument
    -- Executing [100@phones:3] GotoIf("SIP/james-00000006", "0?100-busy,1") in new stack
    -- Executing [100@phones:4] VoiceMail("SIP/james-00000006", "100,u") in new stack
    -- <SIP/james-00000006> Playing 'vm-theperson.gsm' (language 'en')
    -- <SIP/james-00000006> Playing 'digits/1.gsm' (language 'en')
    -- <SIP/james-00000006> Playing 'digits/0.gsm' (language 'en')
    -- <SIP/james-00000006> Playing 'digits/0.gsm' (language 'en')
    -- <SIP/james-00000006> Playing 'vm-isunavail.gsm' (language 'en')
    -- <SIP/james-00000006> Playing 'vm-intro.gsm' (language 'en')
    -- <SIP/james-00000006> Playing 'beep.gsm' (language 'en')
    -- Recording the message
    -- x=0, open writing:  /var/spool/asterisk/voicemail/default/100/tmp/wZLEpl format: wav49, 0x7f9a10001fb8
    -- x=1, open writing:  /var/spool/asterisk/voicemail/default/100/tmp/wZLEpl format: gsm, 0x7f9a1000b6c8
    -- x=2, open writing:  /var/spool/asterisk/voicemail/default/100/tmp/wZLEpl format: wav, 0x7f9a10005008
    -- Recording automatically stopped after a silence of 10 seconds
    -- <SIP/james-00000006> Playing 'auth-thankyou.gsm' (language 'en')
    -- Executing [100@phones:5] Hangup("SIP/james-00000006", "") in new stack
  == Spawn extension (phones, 100, 5) exited non-zero on 'SIP/james-00000006'
ubuntu*CLI>
```

And call *200 for stimulating busy

```
root@ubuntu: ~
    -- Executing [100@phones:2] Dial("SIP/james-00000006", "SIP/nirmal,5") in new stack
  == Using SIP RTP CoS mark 5
    -- Called SIP/nirmal
    -- SIP/nirmal-00000007 is ringing
    -- Nobody picked up in 5000 ms
[Aug 18 14:16:00] WARNING[3295][C-00000008]: ast_expr2.y:1340 op_minus: non-numeric argument
    -- Executing [100@phones:3] GotoIf("SIP/james-00000006", "0?100-busy,1") in new stack
    -- Executing [100@phones:4] VoiceMail("SIP/james-00000006", "100,u") in new stack
    -- <SIP/james-00000006> Playing 'vm-theperson.gsm' (language 'en')
    -- <SIP/james-00000006> Playing 'digits/1.gsm' (language 'en')
    -- <SIP/james-00000006> Playing 'digits/0.gsm' (language 'en')
    -- <SIP/james-00000006> Playing 'digits/0.gsm' (language 'en')
    -- <SIP/james-00000006> Playing 'vm-isunavail.gsm' (language 'en')
    -- <SIP/james-00000006> Playing 'vm-intro.gsm' (language 'en')
    -- <SIP/james-00000006> Playing 'beep.gsm' (language 'en')
    -- Recording the message
    -- x=0, open writing:  /var/spool/asterisk/voicemail/default/100/tmp/wZLEpl format: wav49, 0x7f9a10001fb8
    -- x=1, open writing:  /var/spool/asterisk/voicemail/default/100/tmp/wZLEpl format: gsm, 0x7f9a1000b6c8
    -- x=2, open writing:  /var/spool/asterisk/voicemail/default/100/tmp/wZLEpl format: wav, 0x7f9a10005008
    -- Recording automatically stopped after a silence of 10 seconds
    -- <SIP/james-00000006> Playing 'auth-thankyou.gsm' (language 'en')
    -- Executing [100@phones:5] Hangup("SIP/james-00000006", "") in new stack
  == Spawn extension (phones, 100, 5) exited non-zero on 'SIP/james-00000006'
  == Using SIP RTP CoS mark 5
    -- Executing [*200@phones:1] Answer("SIP/james-00000008", "") in new stack
    -- Executing [*200@phones:2] WaitExten("SIP/james-00000008", "") in new stack
    -- Timeout on SIP/james-00000008, continuing...
    -- Executing [*200@phones:3] Hangup("SIP/james-00000008", "") in new stack
  == Spawn extension (phones, *200, 3) exited non-zero on 'SIP/james-00000008'
ubuntu*CLI>
```

4. Dial 100, you will listen your busy messages

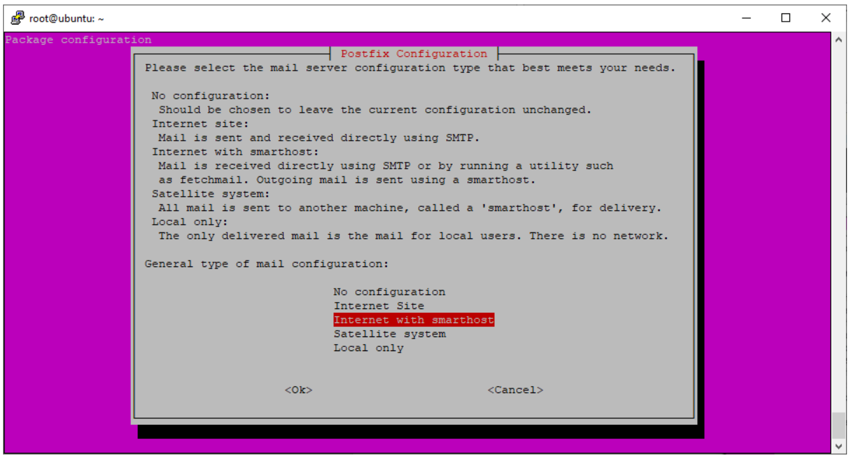
*******Please record voice message and test*******

## <u>Test Mail Server</u>

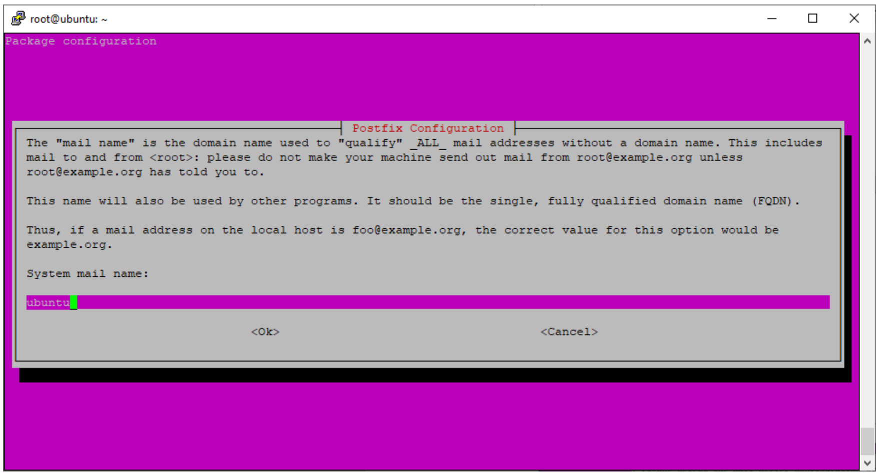
1. Install package postfix and configure

```
# apt-get install postfix
```

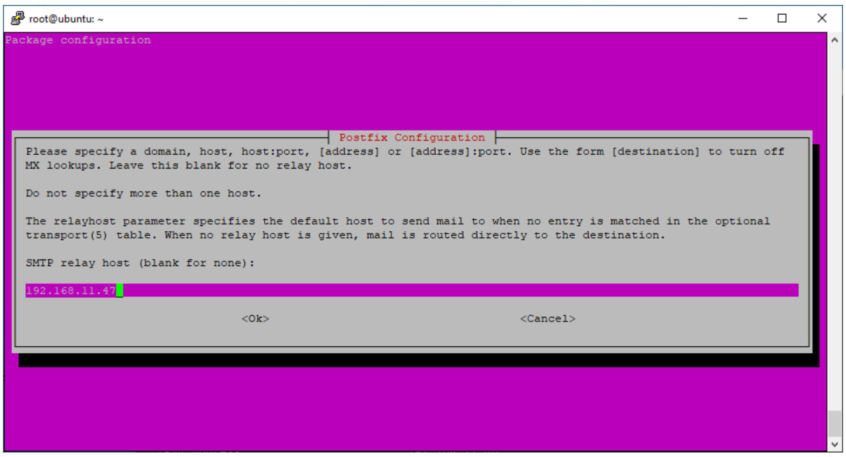
Choose Internet with smarthost

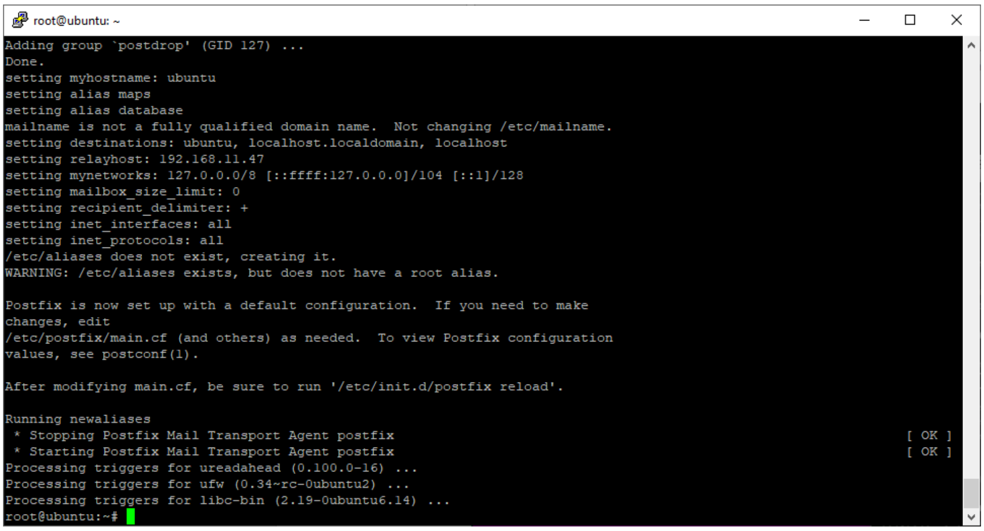



Next



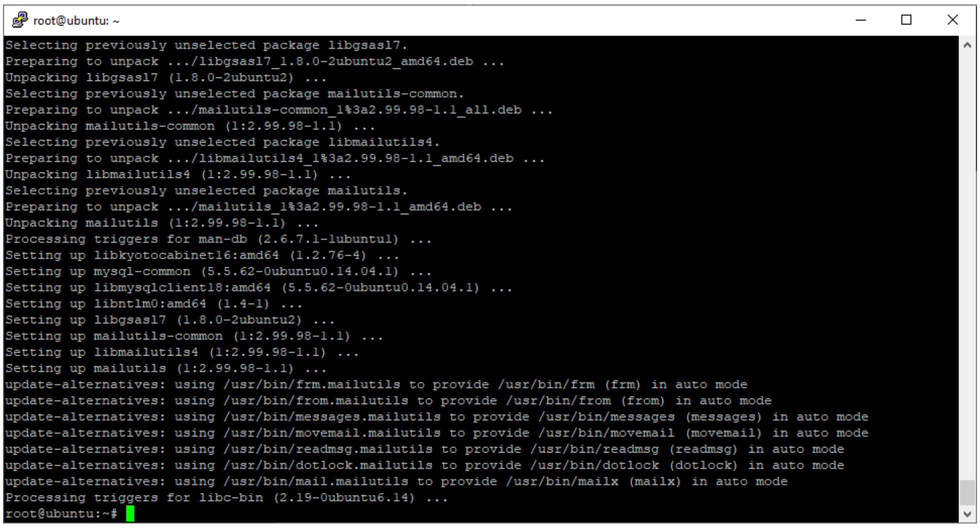

Enter Host Address






2. Install package mailutils

```
# apt-get install mailutils
```




3. Install **mailcatcher**, dial 100 and see mailcatcher (catches mail)

   Note: to send text mail

```
#echo hello | mail -s test nirmal@example.com
```

## Call Distribution

### Simple example for call distribution

1. Open /etc/asterisk/extensions.conf

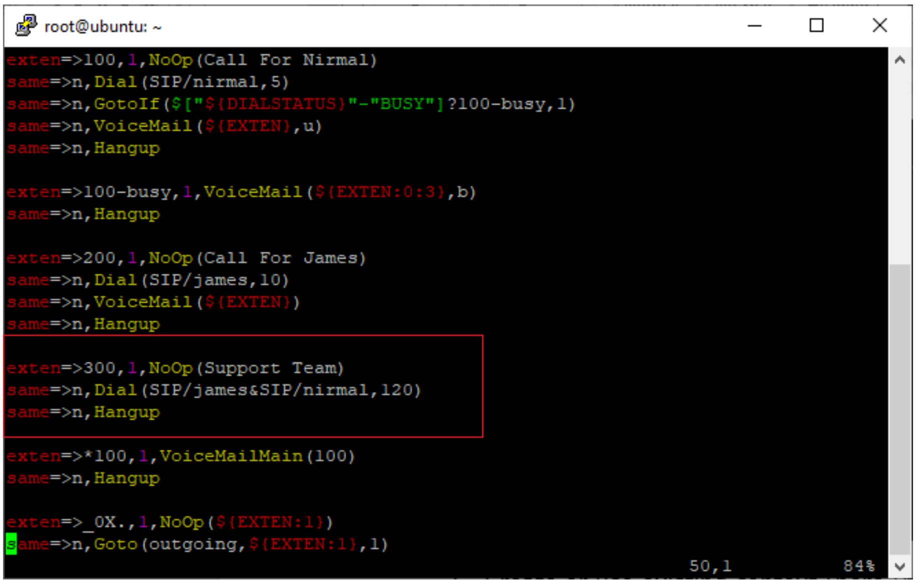
```
# vi /etc/asterisk/extensions.conf
```

2. Create new extension [incoming] add below code

```
exten =>991123300,1,GoSup(timecheck,s,1)
exten =>991123300,n,Goto(phones,300,1)
```


3. Create or use existing context (here [phones]) add below code

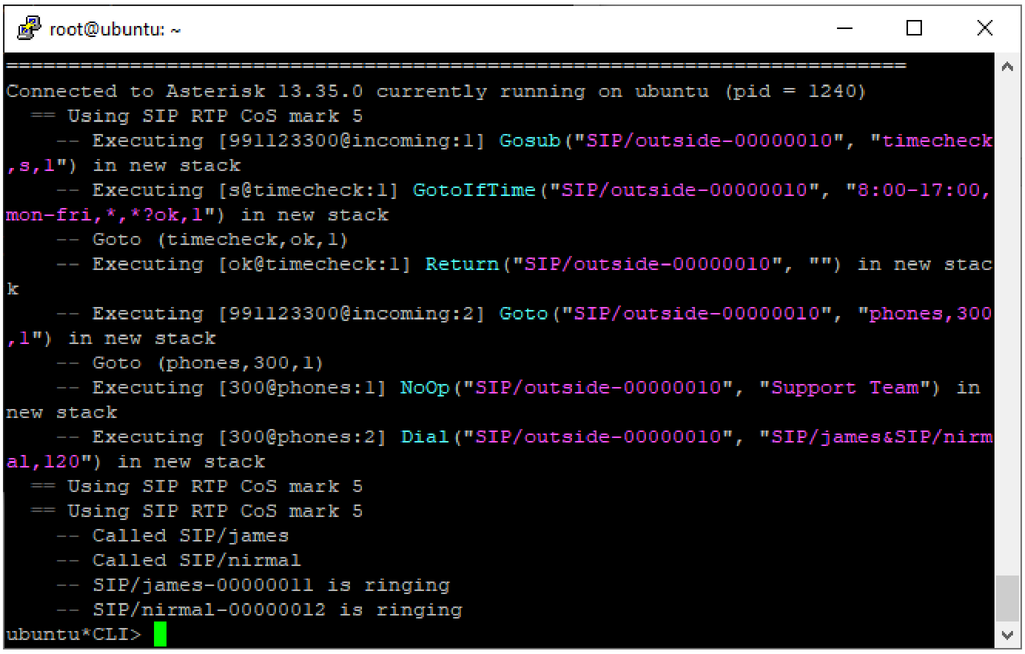
```
exten=>300,1,NoOp(Support Team)
same=>n,Dial(SIP/james&SIP/nirmal,120)
same=> n,Hangup
```



4. Goto asterisk console and reload dial plan

```
# asterisk -rvvv
# dialplan reload
```

5. Dial 991123300 – This will ring both phones mentioned in dial plans



```
root@ubuntu: ~
==========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1240)
  == Using SIP RTP CoS mark 5
    -- Executing [991123300@incoming:1] Gosub("SIP/outside-00000010", "timecheck
,s,1") in new stack
    -- Executing [s@timecheck:1] GotoIfTime("SIP/outside-00000010", "8:00-17:00,
mon-fri,*,*?ok,1") in new stack
    -- Goto (timecheck,ok,1)
    -- Executing [ok@timecheck:1] Return("SIP/outside-00000010", "") in new stac
k
    -- Executing [991123300@incoming:2] Goto("SIP/outside-00000010", "phones,300
,1") in new stack
    -- Goto (phones,300,1)
    -- Executing [300@phones:1] NoOp("SIP/outside-00000010", "Support Team") in
new stack
    -- Executing [300@phones:2] Dial("SIP/outside-00000010", "SIP/james&SIP/nirm
al,120") in new stack
  == Using SIP RTP CoS mark 5
  == Using SIP RTP CoS mark 5
    -- Called SIP/james
    -- Called SIP/nirmal
    -- SIP/james-00000011 is ringing
    -- SIP/nirmal-00000012 is ringing
ubuntu*CLI>
```

## Queue

Queues are defined and configured in **queues.conf**. This file is - as you are already familiar - divided into sections. Under **[general]**, we always set **persistentmembers = yes**, so that agents are re-added to their respective queues when Asterisk is started.

Every queue goes in its own section. In our example, we are configuring a support queue in its own section, [support].

### Implement Call Queues – Simple Example

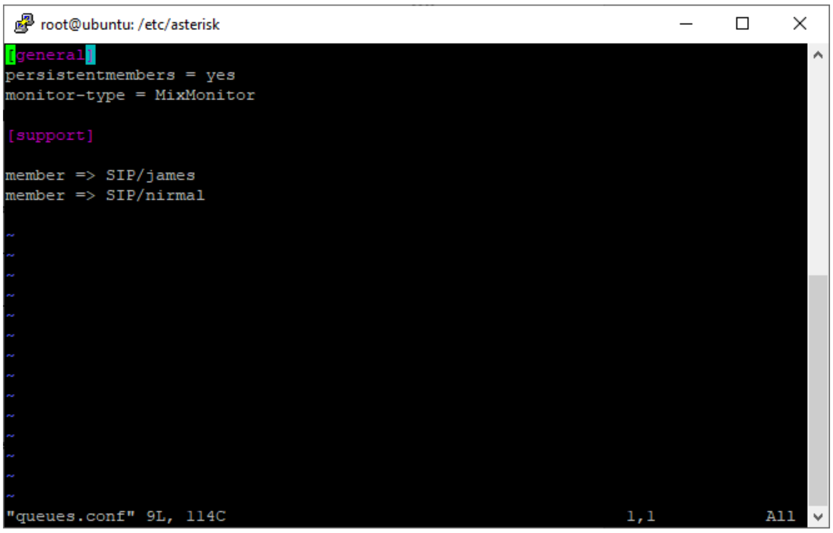
1. Copy existing queues.conf

```
#cp queues.conf queues.conf.org
```

2. Remove all unwanted lines (Optional)

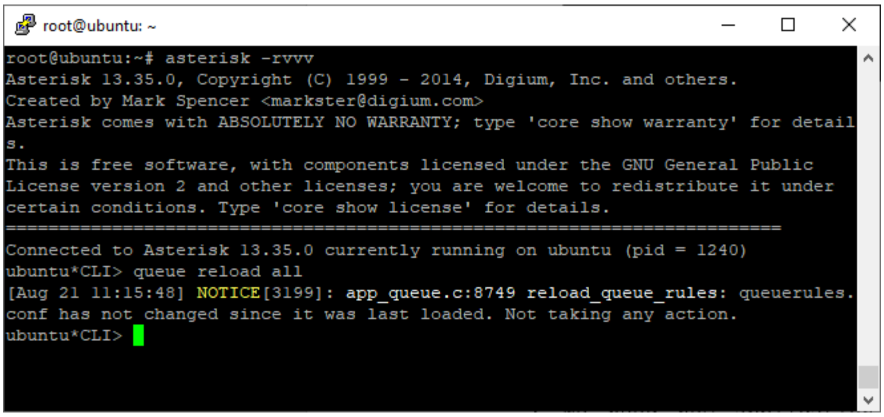
3. Add members in queue (example – support)

```
[support]
      member => SIP/james
      member => SIP/nirmal
```



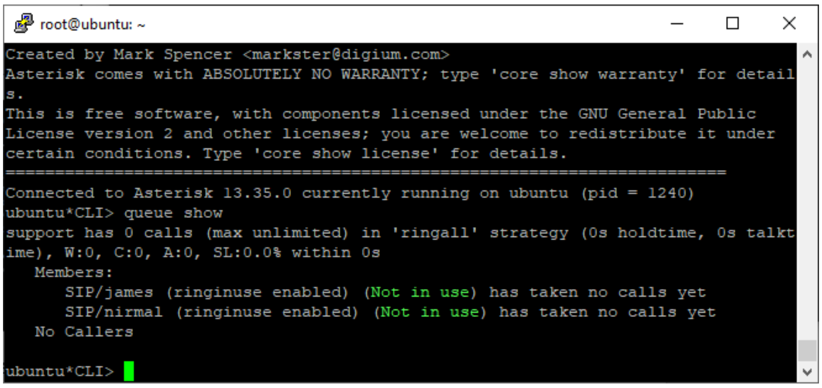

4. Go to asterisk console and reload queue

```
#queue reload all
```



To show all available queues

```
# queue show
```



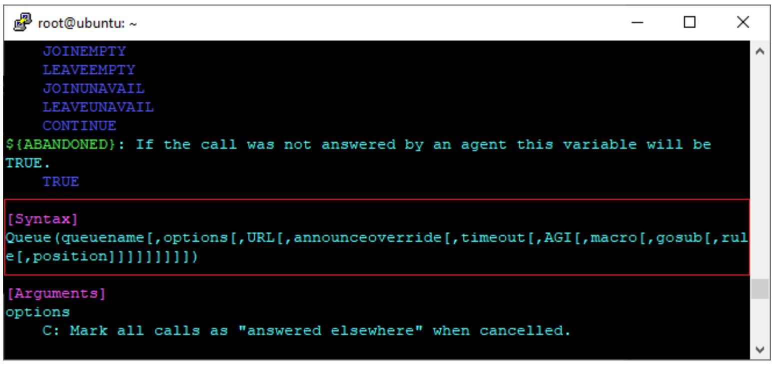
```
root@ubuntu: ~
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for detail
s.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1240)
ubuntu*CLI> queue show
support has 0 calls (max unlimited) in 'ringall' strategy (0s holdtime, 0s talkt
ime), W:0, C:0, A:0, SL:0.0% within 0s
   Members:
      SIP/james (ringinuse enabled) (Not in use) has taken no calls yet
      SIP/nirmal (ringinuse enabled) (Not in use) has taken no calls yet
   No Callers

ubuntu*CLI>
```

To see specific queue

```
# queue show support
```

5. We need new application called Queue. Check queue application and understand syntax

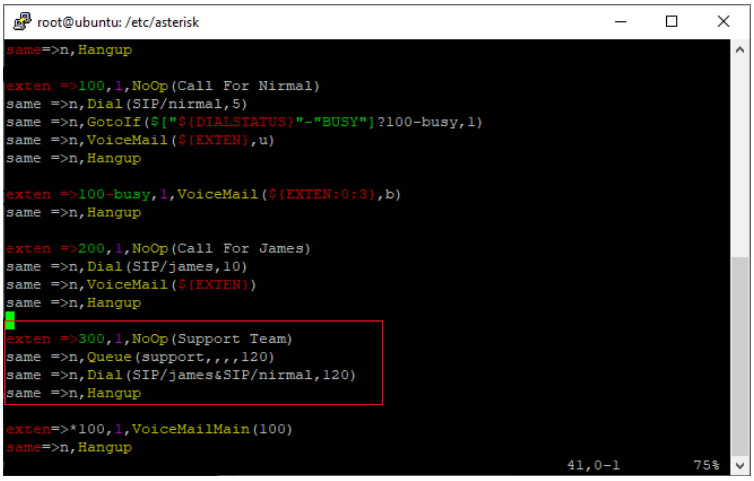
```
#core show application queue
```



```
root@ubuntu: ~
    JOINEMPTY
    LEAVEEMPTY
    JOINUNAVAIL
    LEAVEUNAVAIL
    CONTINUE
${ABANDONED}: If the call was not answered by an agent this variable will be
TRUE.
    TRUE

[Syntax]
Queue(queuename[,options[,URL[,announceoverride[,timeout[,AGI[,macro[,gosub[,rul
e[,position]]]]]]]]])

[Arguments]
options
    C: Mark all calls as "answered elsewhere" when cancelled.
```

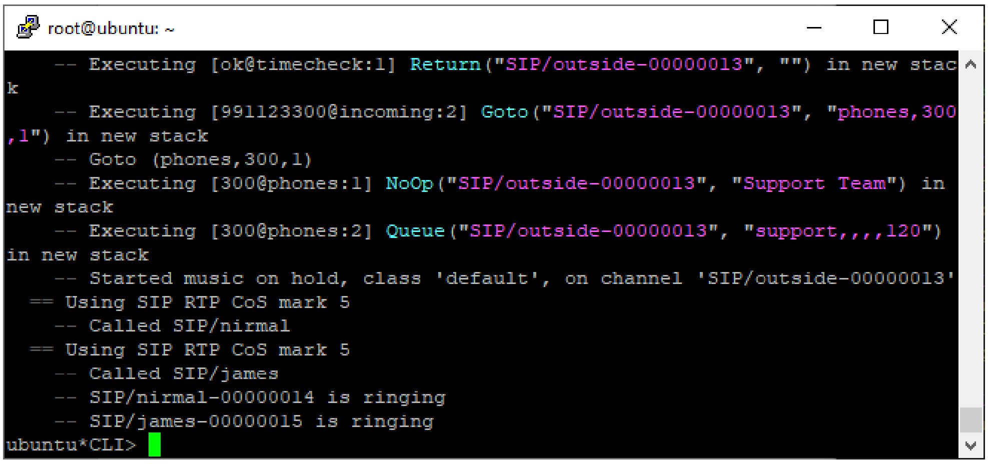
6. Go to dialplan extension and add queue application in extension supports

```
exten =>300,1,NoOp(Support Team)
same =>n,Queue(support,,,,120)
same =>n,Dial(SIP/james&SIP/nirmal,120)
same => n,Hangup
```

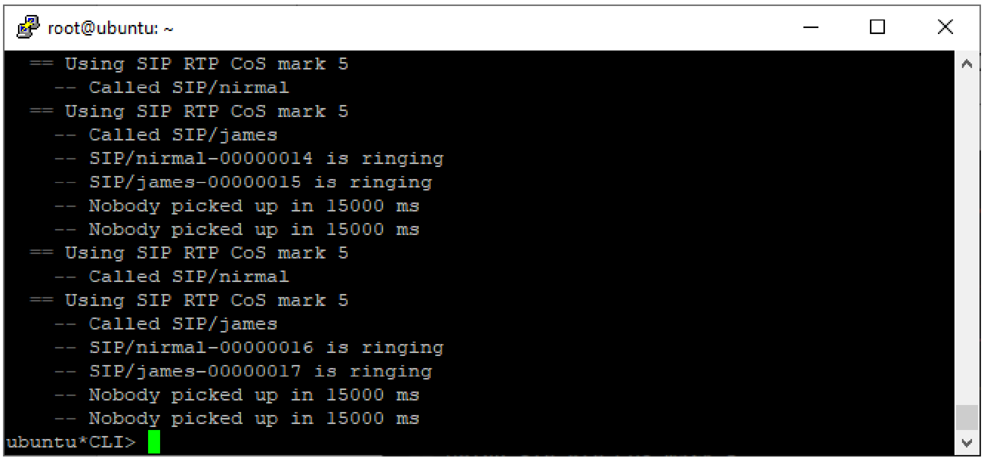


```
root@ubuntu: /etc/asterisk
same=>n,Hangup

exten =>100,1,NoOp(Call For Nirmal)
same =>n,Dial(SIP/nirmal,5)
same =>n,GotoIf($["${DIALSTATUS}"="BUSY"]?100-busy,1)
same =>n,VoiceMail(${EXTEN},u)
same =>n,Hangup

exten =>100-busy,1,VoiceMail(${EXTEN:0:3},b)
same =>n,Hangup

exten =>200,1,NoOp(Call For James)
same =>n,Dial(SIP/james,10)
same =>n,VoiceMail(${EXTEN})
same =>n,Hangup

exten =>300,1,NoOp(Support Team)
same =>n,Queue(support,,,,120)
same =>n,Dial(SIP/james&SIP/nirmal,120)
same =>n,Hangup

exten=>*100,1,VoiceMailMain(100)
same=>n,Hangup
                                                    41,0-1          75%
```

7. Go to asterisk, reload dialplan and dial 991123300 (from outside)




After timeout it try again



**Queue Call Stategies**

Valid strategies include:

- **ringall** - ring all available channels until one answers (default)
- **leastrecent** - ring interface which was least recently hung up by this queue
- **fewestcalls** - ring the one with fewest completed calls from this queue
- **random** - ring random interface
- **rrmemory** - round robin with memory, remember where we left off last ring pass
- **rrordered** - same as rrmemory, except the queue member order from config file is preserved
- **linear** - rings interfaces in the order specified in this configuration file.
  If you use dynamic members, the members will be rung in the order in which they were added
- **wrandom** - rings random interface, but uses the member's penalty as a weight when calculating their metric. So, a member with penalty 0 will have a metric somewhere between 0 and 1000, and a member with penalty 1 will have a metric between 0 and 2000, and a member with penalty 2 will have a metric between 0 and 3000.

  Please note, if using this strategy, the member penalty is not the same as when using other queue strategies. It is ONLY used as a weight for calculating metric.

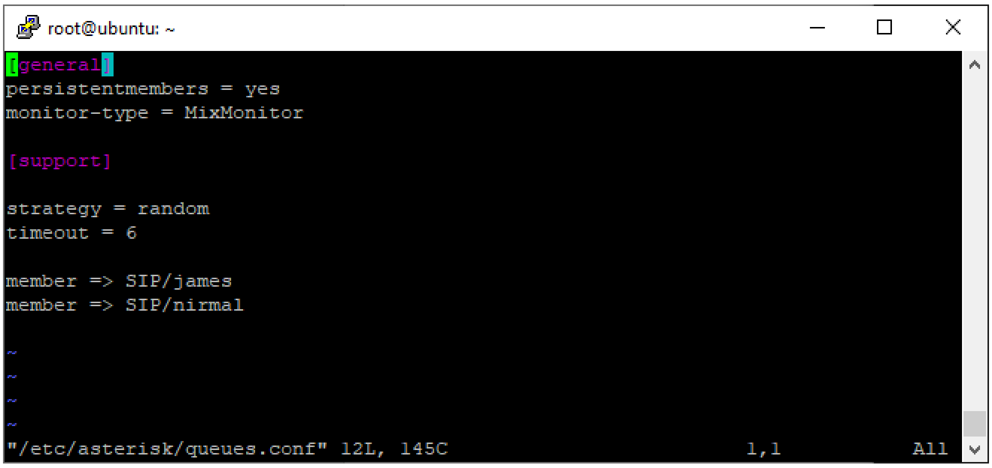
**Implement Call Strategies**

1. Go to queues.conf, add **strategy** and **timeout** parameter

   #vi /etc/asterisk/queues.conf

   Add

   ```
   strategy =random
   timeout=6
   ```

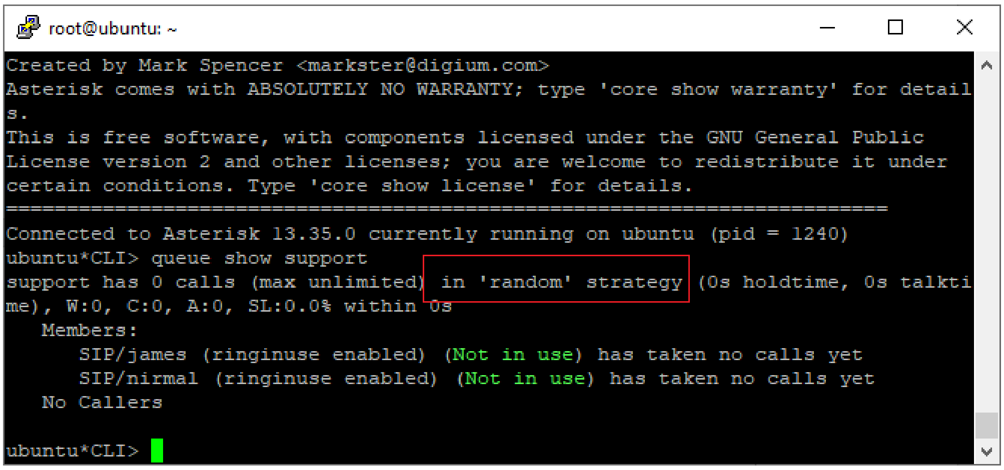
   
   

2. Go to asterisk console and reload queues.conf

   ```
   #queue reload all
   ```

3. Check queue strategy

   ```
   #queue show support
   ```



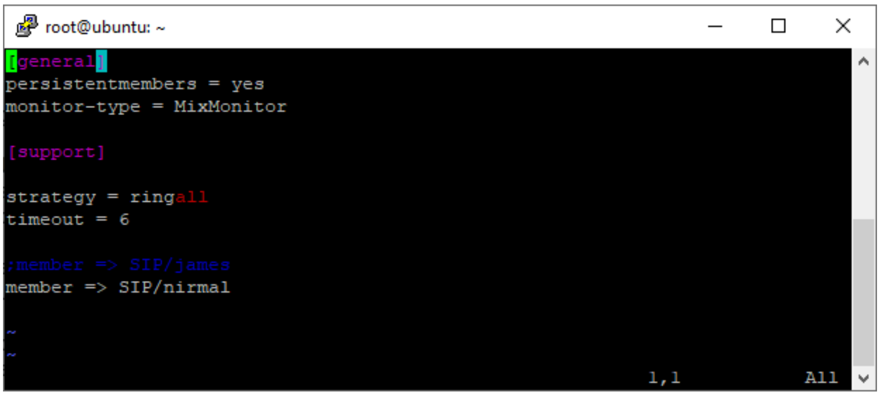
```
root@ubuntu: ~
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for detail
s.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1240)
ubuntu*CLI> queue show support
support has 0 calls (max unlimited) in 'random' strategy (0s holdtime, 0s talkti
me), W:0, C:0, A:0, SL:0.0% within 0s
   Members:
      SIP/james (ringinuse enabled) (Not in use) has taken no calls yet
      SIP/nirmal (ringinuse enabled) (Not in use) has taken no calls yet
   No Callers

ubuntu*CLI>
```

Dial

### Dynamic Queue Member

1. Go to queues.conf and comment one member

   #vi /etc/asterisk/queues.conf



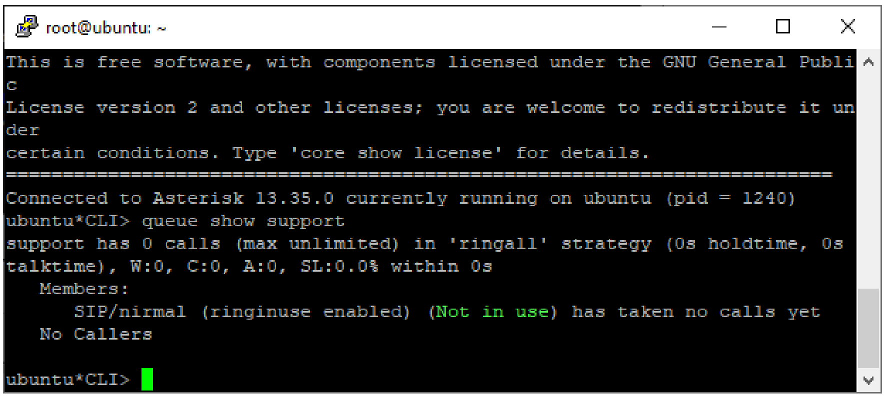
```
root@ubuntu: ~
[general]
persistentmembers = yes
monitor-type = MixMonitor

[support]

strategy = ringall
timeout = 6

;member => SIP/james
member => SIP/nirmal

~
~
                                                  1,1           All
```

2. Go to asterisk console and reload queue

   #queue reload all

3. To show member in queue

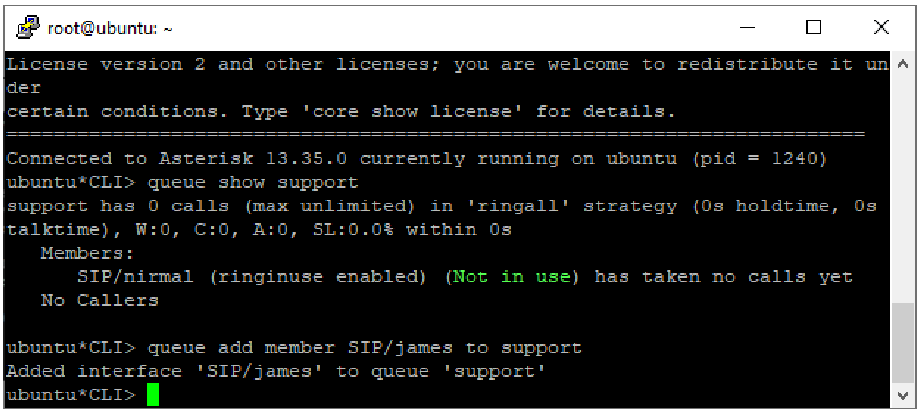
   #queue show support



```
root@ubuntu: ~
This is free software, with components licensed under the GNU General Publi
c
License version 2 and other licenses; you are welcome to redistribute it un
der
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1240)
ubuntu*CLI> queue show support
support has 0 calls (max unlimited) in 'ringall' strategy (0s holdtime, 0s
talktime), W:0, C:0, A:0, SL:0.0% within 0s
   Members:
      SIP/nirmal (ringinuse enabled) (Not in use) has taken no calls yet
   No Callers

ubuntu*CLI>
```

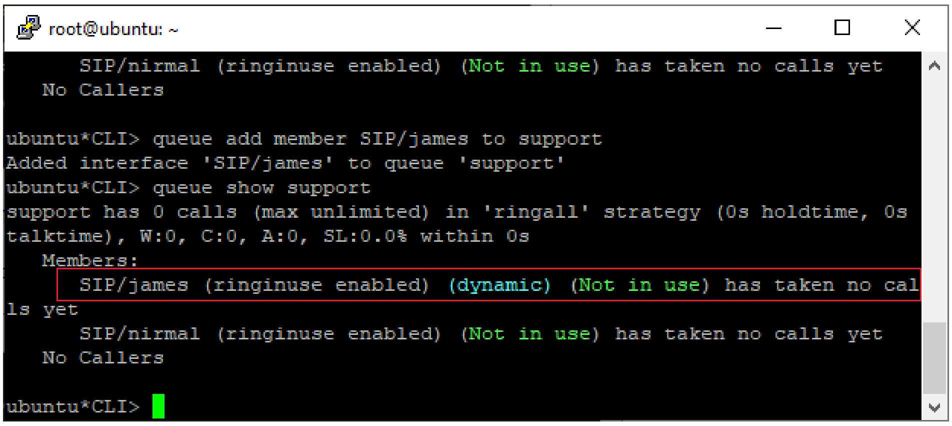
4. To add dynamic member

   #queue add member SIP/james to support




5. To show member

   #queue show support




6. To remove member

   #queue remove member SIP/james to support

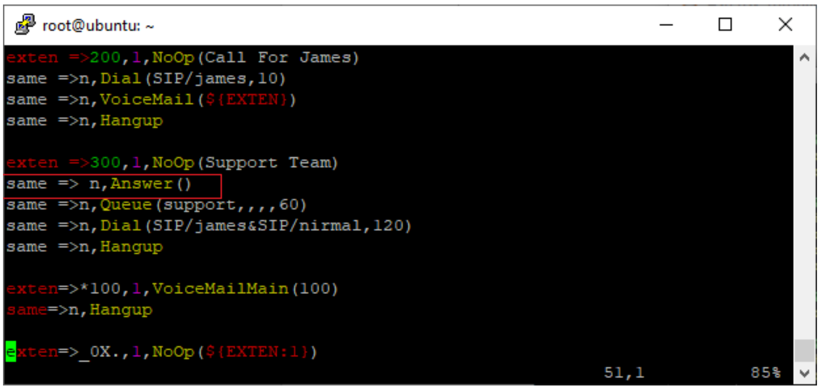
7. Dial and check

## Music On Hold

1. Add below code in extension

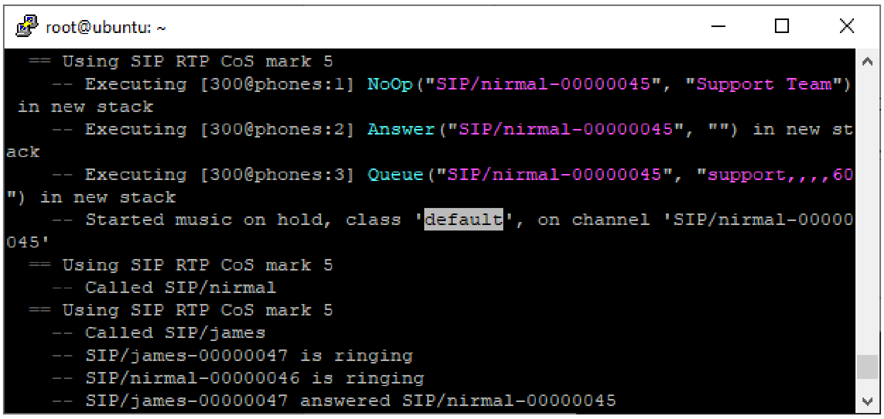
   ```
   #vi /etc/asterisk/extensions.conf
   ```

   ```
   Same=>n,Answer()
   ```



```
root@ubuntu: ~
exten =>200,1,NoOp(Call For James)
same =>n,Dial(SIP/james,10)
same =>n,VoiceMail(${EXTEN})
same =>n,Hangup

exten =>300,1,NoOp(Support Team)
same => n,Answer()
same =>n,Queue(support,,,,60)
same =>n,Dial(SIP/james&SIP/nirmal,120)
same =>n,Hangup

exten=>*100,1,VoiceMailMain(100)
same=>n,Hangup

exten=>_0X.,1,NoOp(${EXTEN:1})
                                                51,1          85%
```

2. Dial 300 (Optional)



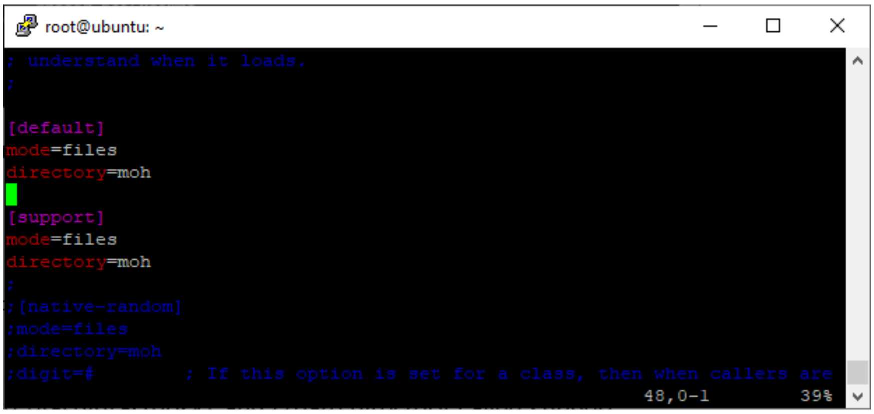
```
root@ubuntu: ~
  == Using SIP RTP CoS mark 5
    -- Executing [300@phones:1] NoOp("SIP/nirmal-00000045", "Support Team")
 in new stack
    -- Executing [300@phones:2] Answer("SIP/nirmal-00000045", "") in new st
ack
    -- Executing [300@phones:3] Queue("SIP/nirmal-00000045", "support,,,,60
") in new stack
    -- Started music on hold, class 'default', on channel 'SIP/nirmal-00000
045'
  == Using SIP RTP CoS mark 5
    -- Called SIP/nirmal
  == Using SIP RTP CoS mark 5
    -- Called SIP/james
    -- SIP/james-00000047 is ringing
    -- SIP/nirmal-00000046 is ringing
    -- SIP/james-00000047 answered SIP/nirmal-00000045
```

3. Open **musiconhold.conf** and add context

   ```
   #vi /etc/asterisk/musiconhold.conf
   ```

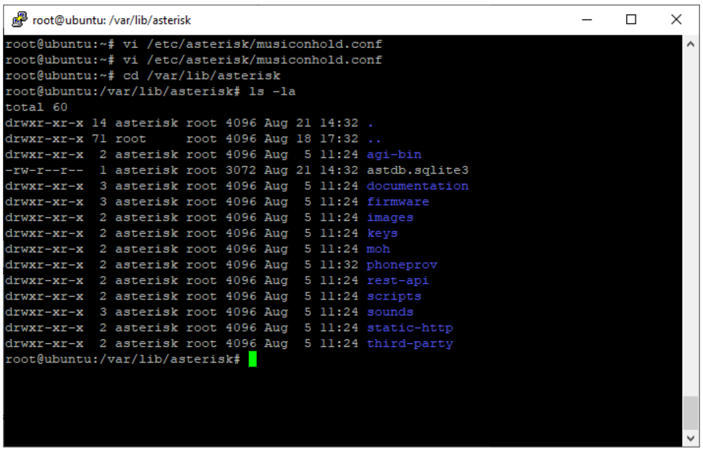
   Add

   ```
   [support]
    mode=files
    directory=support
   ```




4. Go to **/var/lib/asterisk/** and create directory called support

```
#cd /var/lib/asterisk
```



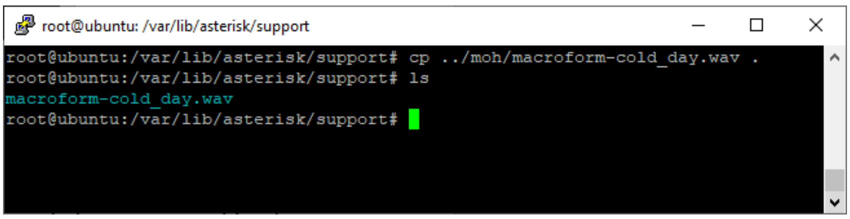

#mkdir **support**

5. Change ownership of newly created directory

```
#chown asterisk support/
```

6. Go newly created directory and copy required files and change ownership of newly copied files

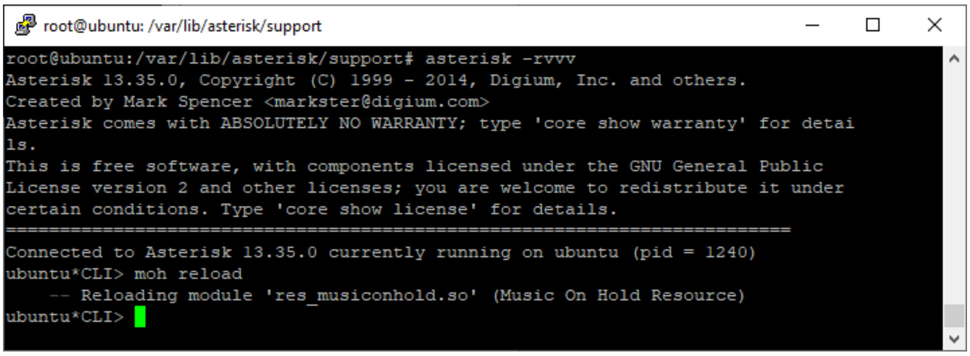
```
#cd support
#cp ../moh/macroform-cold_day.wav .
```




```
#Chown asterisk macroform-cold_day.wav
```

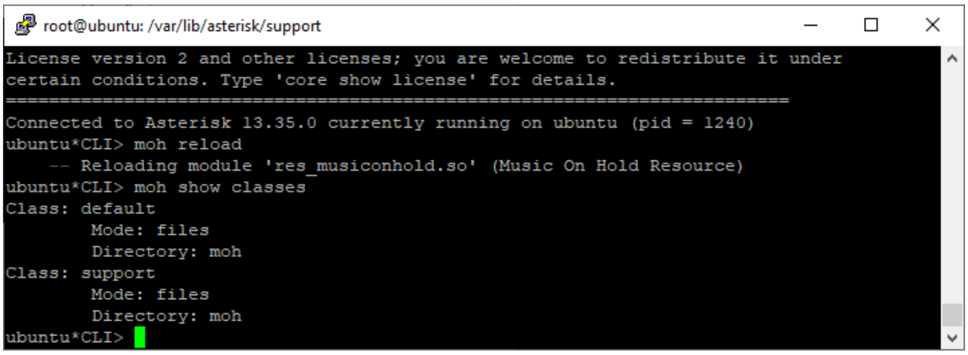
7. Go to asterisk console and reload

```
#asterisk -rvvv
#moh reload
```




To show all classes

```
#moh show classes
```

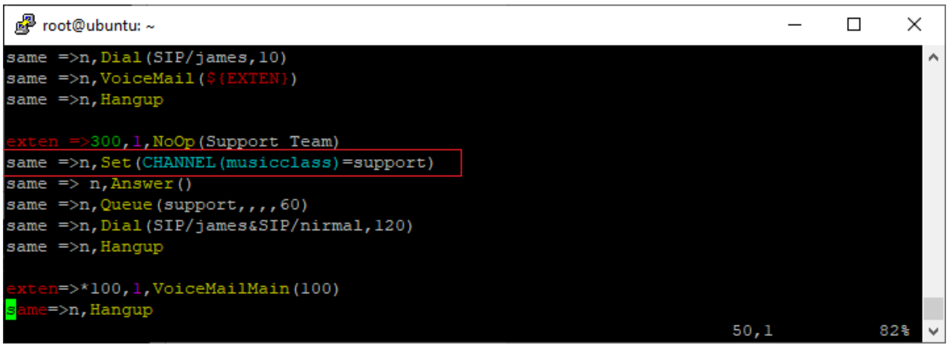



8. Go to extensions.conf

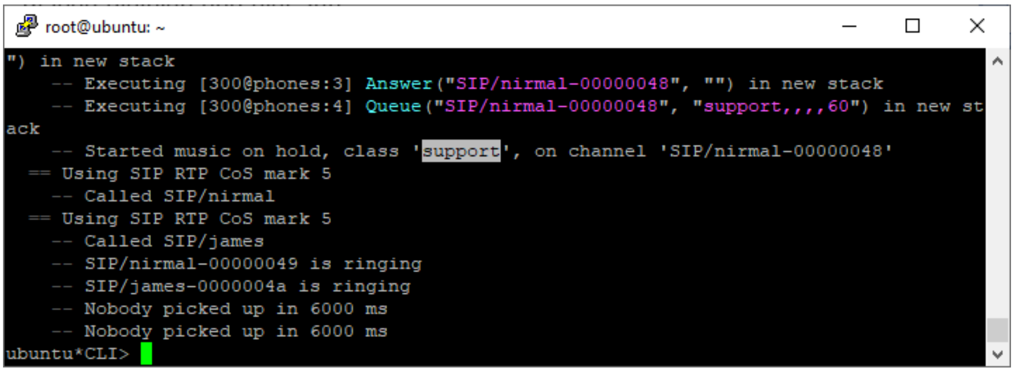
```
#vi /etc/asterisk/extensions.conf
```

Add

```
Same=>n,Set(CHANNEL(musicclass)=support)
```



9. Reload dialplan and dial 300



### Agents - Dynamic Queue Agents

Agents are human beings – well, phone extensions that are used by human beings. Setting up agents in the Asterisk config **agents.conf** file allows you to then assign agents in your call queues as a member. One agent can be assigned to many queues, and you can permit an agent to login from any extension.

ACD distributes incoming calls to the agents of a Queue. Agents are configured in the file **queues.conf**:

New in Asterisk v1.2.0: The default for **ackcall** has been changed to "no" instead of "yes" because of a bug which caused the "**yes**" behavior to generally act like "**no**". You may need to adjust the value if your agents behave differently than you expect with respect to acknowledgement.

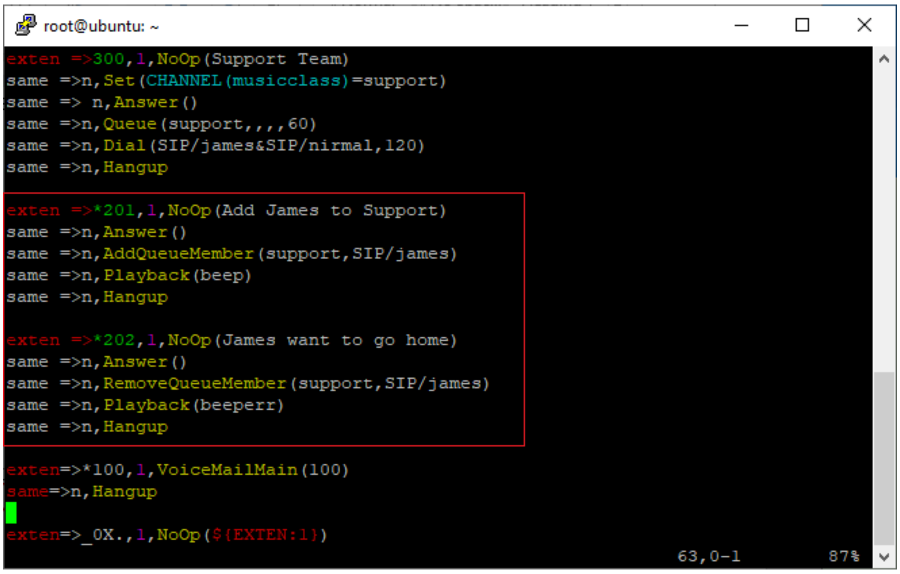
### Implement Dynamic Queue Agents

1. Open application called AddQueueMember and RemoveQueueMember in asterisk console and check syntax

```
# core show application AddQueueMember
# core show application RemoveQueueMember
```

2. Open /etc/asterisk/extensions.conf and add extension

```
exten =>*201,1,NoOp(Add James to Support)
same =>n,Answer()
same =>n,AddQueueMember(support,SIP/james)
same =>n,Playback(beep)
same =>n,Hangup

exten =>*202,1,NoOp(James want to go home)
same =>n,Answer()
same =>n,RemoveQueueMember(support,SIP/james)
same =>n,Playback(beeperr)
same =>n,Hangup
```



3. Go to asterisk console and reload queue and dial plan

```
#asterisk -rvvv
#dialplan reload
#queue reload all
```

4. To check Member

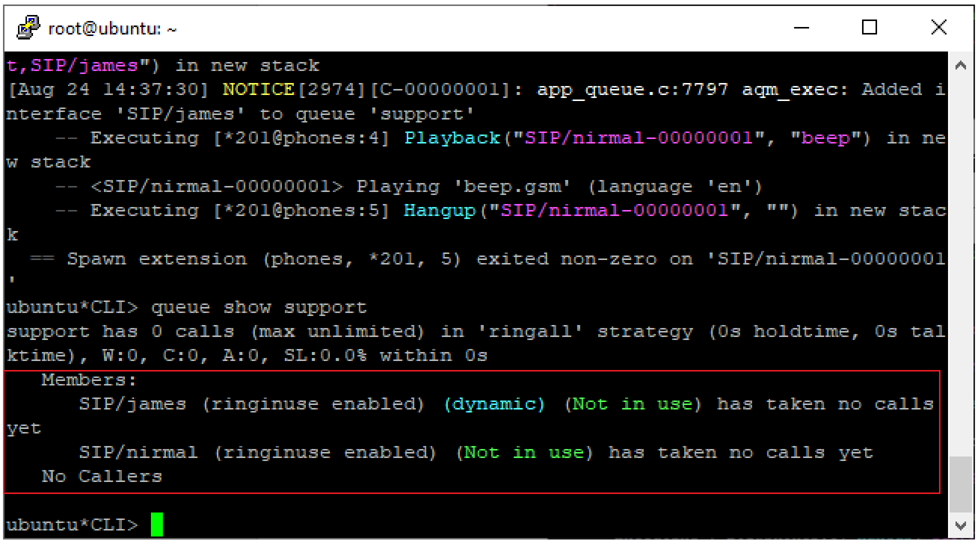
#queue show support

```
root@ubuntu: ~
root@ubuntu:~# asterisk -rvvv
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for deta
s.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1253)
ubuntu*CLI> queue show support
support has 0 calls (max unlimited) in 'ringall' strategy (0s holdtime, 0s tal
ime), W:0, C:0, A:0, SL:0.0% within 0s
   Members:
      SIP/nirmal (ringinuse enabled) (Not in use) has taken no calls yet
   No Callers

ubuntu*CLI>
```

5. To add dynamic member / Agent dial ***201**

```
root@ubuntu: ~
      SIP/nirmal (ringinuse enabled) (Not in use) has taken no calls yet
   No Callers

  == Using SIP RTP CoS mark 5
    -- Executing [*201@phones:1] NoOp("SIP/nirmal-00000001", "Add James to Sup
port") in new stack
    -- Executing [*201@phones:2] Answer("SIP/nirmal-00000001", "") in new stac
k
    -- Executing [*201@phones:3] AddQueueMember("SIP/nirmal-00000001", "suppor
t,SIP/james") in new stack
[Aug 24 14:37:30] NOTICE[2974][C-00000001]: app_queue.c:7797 aqm_exec: Added i
nterface 'SIP/james' to queue 'support'
    -- Executing [*201@phones:4] Playback("SIP/nirmal-00000001", "beep") in ne
w stack
    -- <SIP/nirmal-00000001> Playing 'beep.gsm' (language 'en')
    -- Executing [*201@phones:5] Hangup("SIP/nirmal-00000001", "") in new stac
k
  == Spawn extension (phones, *201, 5) exited non-zero on 'SIP/nirmal-00000001
'
ubuntu*CLI>
```

#queue show support



```
root@ubuntu: ~
t,SIP/james") in new stack
[Aug 24 14:37:30] NOTICE[2974][C-00000001]: app_queue.c:7797 aqm_exec: Added i
nterface 'SIP/james' to queue 'support'
    -- Executing [*201@phones:4] Playback("SIP/nirmal-00000001", "beep") in ne
w stack
    -- <SIP/nirmal-00000001> Playing 'beep.gsm' (language 'en')
    -- Executing [*201@phones:5] Hangup("SIP/nirmal-00000001", "") in new stac
k
  == Spawn extension (phones, *201, 5) exited non-zero on 'SIP/nirmal-00000001
'
ubuntu*CLI> queue show support
support has 0 calls (max unlimited) in 'ringall' strategy (0s holdtime, 0s tal
ktime), W:0, C:0, A:0, SL:0.0% within 0s
   Members:
      SIP/james (ringinuse enabled) (dynamic) (Not in use) has taken no calls
yet
      SIP/nirmal (ringinuse enabled) (Not in use) has taken no calls yet
   No Callers

ubuntu*CLI>
```

6. To Remove Dial *202

## Introducing the Asterisk Database

Asterisk comes with a database that is used internally and made available for Asterisk programmers and administrators to use as they see fit.

### Purpose of the internal database

The database really has two purposes:

1. Asterisk uses it to store information that needs to persist between reloads/restarts. Various modules use it for this purpose automatically.
2. Users can use it to store arbitrary data. This is done using a variety of dialplan applications and functions such as:

   - Functions:

     - DB
     - DB_DELETE
     - DB_EXISTS
     - DB_KEYS

   - Application: DBdeltree

The functions and applications for Asterisk 11 are linked above, but you should look at the documentation for the version you have deployed.

### Database commands on the CLI

Sub-commands under the command "database" allow a variety of functions to be performed on or with the database.

```
*CLI> core show help database
database del                -- Removes database key/value
database deltree            -- Removes database keytree/values
database get                -- Gets database value
database put                -- Adds/updates database value
database query              -- Run a user-specified query on the astdb
database show               -- Shows database contents
database showkey            -- Shows database contents
```

## Syntax

**#database del**

```
Usage: database del <family> <key>

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

**#database deltree**

```
Usage: database deltree <family> [keytree]
   OR: database deltree <family>[/keytree]

      Deletes a family or specific keytree within a family      in the Asterisk
      database.  The two arguments may be separated by either a space or a slash.
```

**#database get**

```
Usage: database get <family> <key>

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

**#database put**

```
Usage: database put <family> <key> <value>

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

**#database query**

```
Usage: database query "<SQL Statement>"

       Run a user-specified SQL query on the database. Be careful.
```

**#database showkey**

```
Usage: database showkey <keytree>

       Shows Asterisk database contents, restricted to a given key.
```

## Asterisk Functions

In addition to dialplan applications, which have been part of Asterisk almost from the very beginning, Asterisk also supports functions as of Asterisk 1.2.

This is part of a long-standing effort to make Asterisk behave more like a programming environment. In contrast to applications, functions may not be called directly. Instead, they are called inside applications and return a value, or -- in a departure from the classical definition of a function -- they may even be written to using the application **Set().** Function names are always written in uppercase letters. Surprisingly, functions are written in the same way as variables, inside curly braces and preceded by a **$** character ( **${}** ). This is necessary because strings are not always bounded by quotation marks.

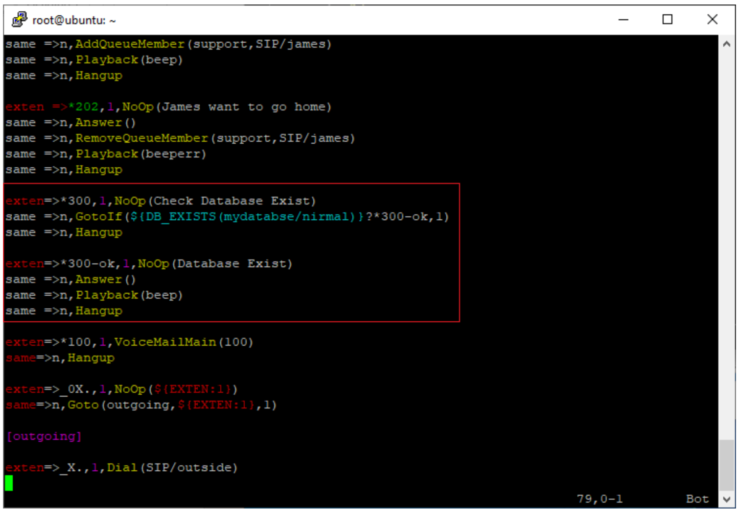
To find out which functions are currently available in your installation, enter **show functions** and **show function FUNCTIONNAME** or **core show functions** and **core show function FUNCTIONNAME** (depending on your Asterisk version) in the CLI. Note that these commands are case-sensitive. Function names must be written entirely in uppercase.

### Example For DB_EXISTS ()

```
[Syntax]
DB_EXISTS(<family>/<key>)

[Synopsis]
Check to see if a key exists in the Asterisk database

[Description]
This function will check to see if a key exists in the Asterisk
database. If it exists, the function will return "1". If not,
it will return "0".  Checking for existence of a database key will
also set the variable DB_RESULT to the key's value if it exists.
```

1. Open extensions.conf and add two extensions add exten

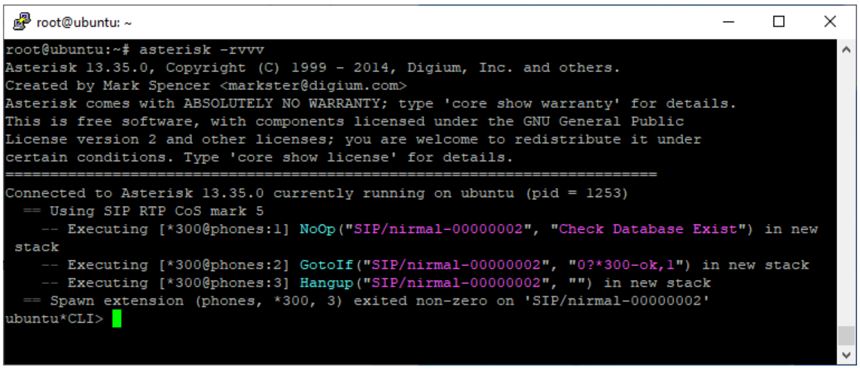
   ```
   #vi /etc/asterisk/extensions.conf

   exten=>*300,1,NoOp(Check Database Exist)
   same =>n,GotoIf(${DB_EXISTS(mydatabse/nirmal)}?*300-ok,1)
   same =>n,Hangup

   exten=>*300-ok,1,NoOp(Database Exist)
   same =>n,Answer()
   same=>n,Playback(beep)
   same =>n,Hangup
   ```



```
root@ubuntu: ~
same =>n,AddQueueMember(support,SIP/james)
same =>n,Playback(beep)
same =>n,Hangup

exten =>*202,1,NoOp(James want to go home)
same =>n,Answer()
same =>n,RemoveQueueMember(support,SIP/james)
same =>n,Playback(beeperr)
same =>n,Hangup

exten=>*300,1,NoOp(Check Database Exist)
same =>n,GotoIf(${DB_EXISTS(mydatabse/nirmal)}?*300-ok,1)
same =>n,Hangup

exten=>*300-ok,1,NoOp(Database Exist)
same =>n,Answer()
same =>n,Playback(beep)
same =>n,Hangup

exten=>*100,1,VoiceMailMain(100)
same=>n,Hangup

exten=>_0X.,1,NoOp(${EXTEN:1})
same=>n,Goto(outgoing,${EXTEN:1},1)

[outgoing]

exten=>_X.,1,Dial(SIP/outside)

79,0-1        Bot
```

2. In asterisk console, reload dialplan and Dial *300

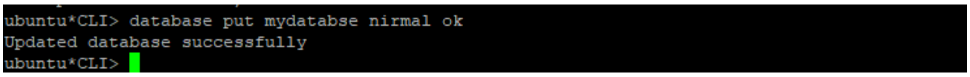
```
#asterisk -rvvv
```



```
root@ubuntu: ~
root@ubuntu:~# asterisk -rvvv
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1253)
  == Using SIP RTP CoS mark 5
    -- Executing [*300@phones:1] NoOp("SIP/nirmal-00000002", "Check Database Exist") in new
 stack
    -- Executing [*300@phones:2] GotoIf("SIP/nirmal-00000002", "0?*300-ok,1") in new stack
    -- Executing [*300@phones:3] Hangup("SIP/nirmal-00000002", "") in new stack
  == Spawn extension (phones, *300, 3) exited non-zero on 'SIP/nirmal-00000002'
ubuntu*CLI>
```

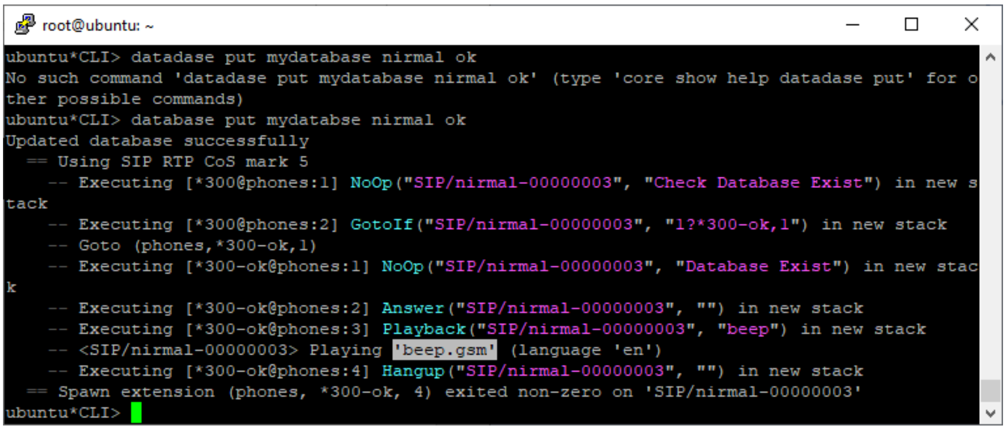
3. Database value not exist add values into database – (based on above screen)

```
#database put mydatabse nirmal ok
```



```
ubuntu*CLI> database put mydatabse nirmal ok
Updated database successfully
ubuntu*CLI>
```

4. Dial *300



```
root@ubuntu: ~
ubuntu*CLI> datadase put mydatabase nirmal ok
No such command 'datadase put mydatabase nirmal ok' (type 'core show help datadase put' for o
ther possible commands)
ubuntu*CLI> database put mydatabse nirmal ok
Updated database successfully
  == Using SIP RTP CoS mark 5
    -- Executing [*300@phones:1] NoOp("SIP/nirmal-00000003", "Check Database Exist") in new s
tack
    -- Executing [*300@phones:2] GotoIf("SIP/nirmal-00000003", "1?*300-ok,1") in new stack
    -- Goto (phones,*300-ok,1)
    -- Executing [*300-ok@phones:1] NoOp("SIP/nirmal-00000003", "Database Exist") in new stac
k
    -- Executing [*300-ok@phones:2] Answer("SIP/nirmal-00000003", "") in new stack
    -- Executing [*300-ok@phones:3] Playback("SIP/nirmal-00000003", "beep") in new stack
    -- <SIP/nirmal-00000003> Playing 'beep.gsm' (language 'en')
    -- Executing [*300-ok@phones:4] Hangup("SIP/nirmal-00000003", "") in new stack
  == Spawn extension (phones, *300-ok, 4) exited non-zero on 'SIP/nirmal-00000003'
ubuntu*CLI>
```

### Dynamic Queues - The Solution

Function Required is **DB**

```
[Syntax]
DB(<family>/<key>)

[Synopsis]
Read from or write to the Asterisk database

[Description]
This function will read from or write a value to the Asterisk database.  On a
read, this function returns the corresponding value from the database, or blank
if it does not exist.  Reading a database value will also set the variable
DB_RESULT.  If you wish to find out if an entry exists, use the DB_EXISTS
function.
```

Example: 1

1. In Asterisk Console – Check existing queue

```
#asterisk -rvvv
#queue show
```

```
root@ubuntu: ~
root@ubuntu:~# asterisk -rvvv
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1253)
ubuntu*CLI> queue show
support has 0 calls (max unlimited) in 'ringall' strategy (0s holdtime, 0s talktime), W:0, C:
0, A:0, SL:0.0% within 0s
   Members:
      SIP/james (ringinuse enabled) (dynamic) (Not in use) has taken no calls yet
      SIP/nirmal (ringinuse enabled) (Not in use) has taken no calls yet
   No Callers

ubuntu*CLI>
```

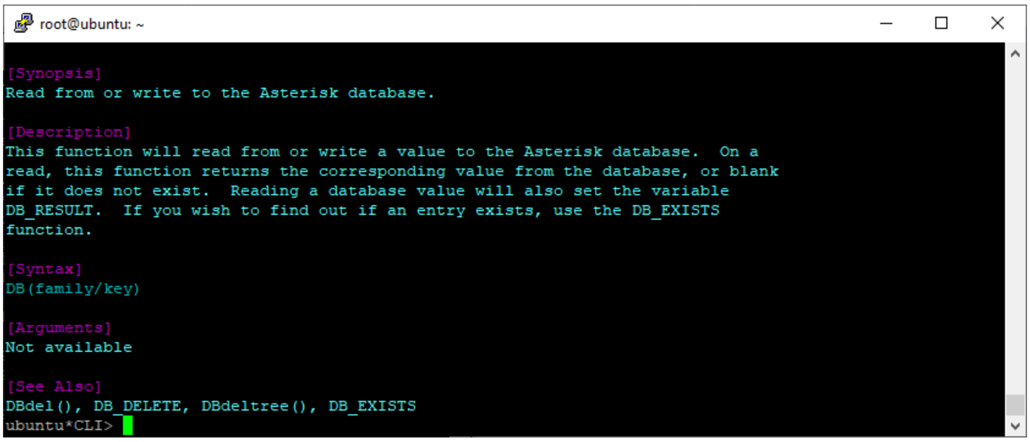
2. Check database for queue entry

```
#database show
```

```
root@ubuntu: ~
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1253)
ubuntu*CLI> database show
/Queue/PersistentMembers/support                  : SIP/james;0;0;SIP/james;SIP/james
/SIP/Registry/james                               : 192.168.11.46:5060:3600:james:sip:james@123.50.200
ort=UDP
/SIP/Registry/nirmal                              : 192.168.11.46:5060:3600:nirmal:sip:nirmal@123.50.2
sport=UDP
/dundi/secret                                     : +R1BPOYG6ytXiv0mw0KaHg==;S+fOJd4EyCXELzYLmEeXWw==
/dundi/secretexpiry                               : 1598259299
/mydatabse/nirmal                                 : ok
/pbx/UUID                                         : 9ef392ed-6b58-41a1-857b-d0a3f9e8b553
7 results found.
ubuntu*CLI>
```

3. We require one new function called DB, check syntax and usage

```
#core show function DB
```

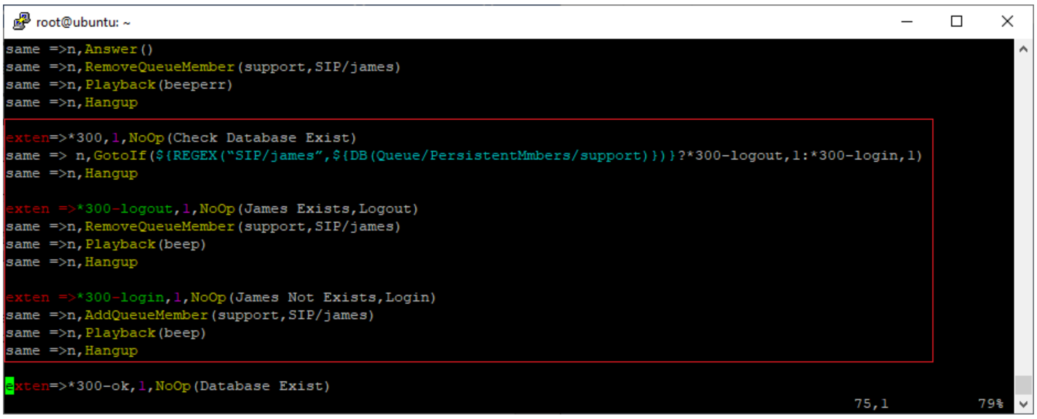



4. Go extensions.conf and edit existing add new exten

```
#vi /etc/asterisk/extensions.conf

exten =>*300,1,NoOp(James Login/Logout)
same =>
n,GotoIf(${REGEX("SIP/james",${DB(Queue/PersistentMembers/support)})}?*300
-logout,1:*300-login,1)
same =>n,Hangup

; ${DB(Queue/PersistentMembers/support)}:- to get database queue entry
; ${REGEX("SIP/james" ),;${DB(Queue/PersistentMembers/support)}}
: - for checking SIP/james is existing in database entry
```

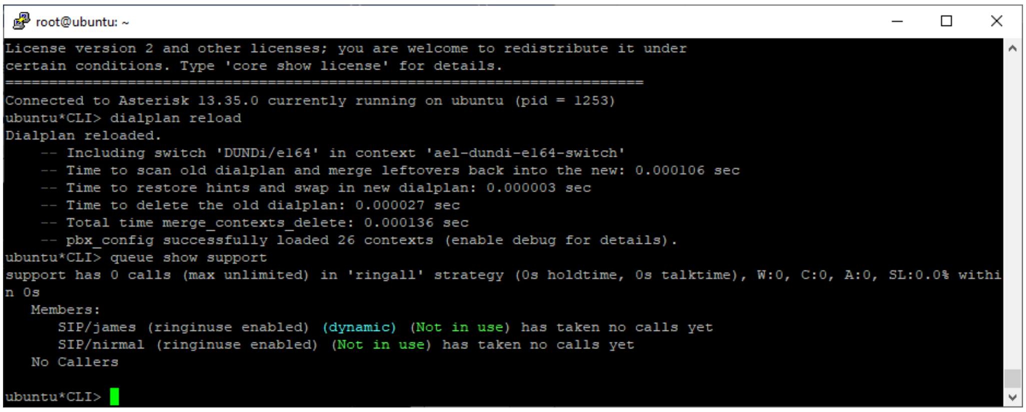
Add new extension

```
exten =>*300-logout,1,NoOp(James Exists,Logout)
same =>n,RemoveQueueMember(support,SIP/james)
same =>n,Playback(beep)
same =>n,Hangup

exten =>*300-login,1,NoOp(James Not Exists,Login)
same =>n,AddQueueMember(support,SIP/james)
same =>n,Playback(beep)
same =>n,Hangup
```



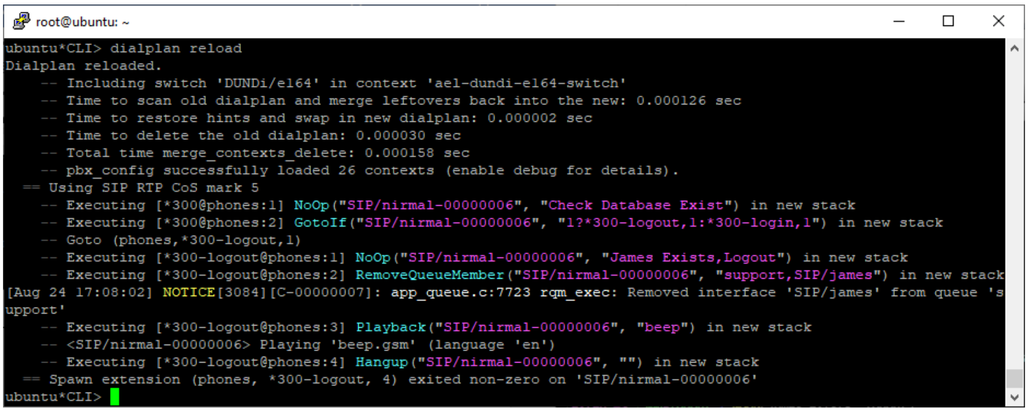
5. In asterisk console, reload dial plan and check support queue

```
#queue show support
```



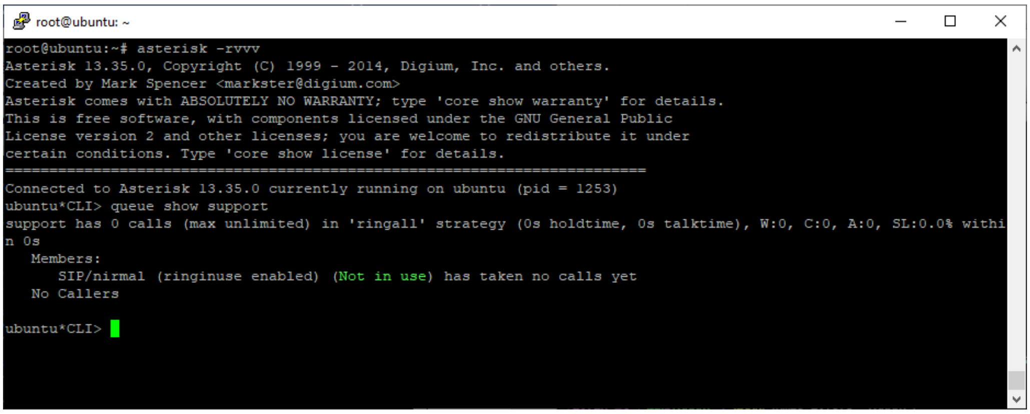
```
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1253)
ubuntu*CLI> dialplan reload
Dialplan reloaded.
    -- Including switch 'DUNDi/e164' in context 'ael-dundi-e164-switch'
    -- Time to scan old dialplan and merge leftovers back into the new: 0.000106 sec
    -- Time to restore hints and swap in new dialplan: 0.000003 sec
    -- Time to delete the old dialplan: 0.000027 sec
    -- Total time merge_contexts_delete: 0.000136 sec
    -- pbx_config successfully loaded 26 contexts (enable debug for details).
ubuntu*CLI> queue show support
support has 0 calls (max unlimited) in 'ringall' strategy (0s holdtime, 0s talktime), W:0, C:0, A:0, SL:0.0% withi
n 0s
   Members:
      SIP/james (ringinuse enabled) (dynamic) (Not in use) has taken no calls yet
      SIP/nirmal (ringinuse enabled) (Not in use) has taken no calls yet
   No Callers

ubuntu*CLI>
```

6. Dial *300 and check support queue in asterisk console – it will remove member from support queue



```
ubuntu*CLI> dialplan reload
Dialplan reloaded.
    -- Including switch 'DUNDi/e164' in context 'ael-dundi-e164-switch'
    -- Time to scan old dialplan and merge leftovers back into the new: 0.000126 sec
    -- Time to restore hints and swap in new dialplan: 0.000002 sec
    -- Time to delete the old dialplan: 0.000030 sec
    -- Total time merge_contexts_delete: 0.000158 sec
    -- pbx_config successfully loaded 26 contexts (enable debug for details).
  == Using SIP RTP CoS mark 5
    -- Executing [*300@phones:1] NoOp("SIP/nirmal-00000006", "Check Database Exist") in new stack
    -- Executing [*300@phones:2] GotoIf("SIP/nirmal-00000006", "1?*300-logout,1:*300-login,1") in new stack
    -- Goto (phones,*300-logout,1)
    -- Executing [*300-logout@phones:1] NoOp("SIP/nirmal-00000006", "James Exists,Logout") in new stack
    -- Executing [*300-logout@phones:2] RemoveQueueMember("SIP/nirmal-00000006", "support,SIP/james") in new stack
[Aug 24 17:08:02] NOTICE[3084][C-00000007]: app_queue.c:7723 rqm_exec: Removed interface 'SIP/james' from queue 's
upport'
    -- Executing [*300-logout@phones:3] Playback("SIP/nirmal-00000006", "beep") in new stack
    -- <SIP/nirmal-00000006> Playing 'beep.gsm' (language 'en')
    -- Executing [*300-logout@phones:4] Hangup("SIP/nirmal-00000006", "") in new stack
  == Spawn extension (phones, *300-logout, 4) exited non-zero on 'SIP/nirmal-00000006'
ubuntu*CLI>
```

```
# queue show support
```



```
root@ubuntu:~# asterisk -rvvv
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1253)
ubuntu*CLI> queue show support
support has 0 calls (max unlimited) in 'ringall' strategy (0s holdtime, 0s talktime), W:0, C:0, A:0, SL:0.0% withi
n 0s
   Members:
      SIP/nirmal (ringinuse enabled) (Not in use) has taken no calls yet
   No Callers

ubuntu*CLI>
```

7. Dial *300 again - it will add Member

```
root@ubuntu: ~
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1253)
ubuntu*CLI> queue show support
support has 0 calls (max unlimited) in 'ringall' strategy (0s holdtime, 0s talktime), W:0, C:0, A:0, SL:0.0% withi
n 0s
   Members:
      SIP/nirmal (ringinuse enabled) (Not in use) has taken no calls yet
   No Callers

  == Using SIP RTP CoS mark 5
    -- Executing [*300@phones:1] NoOp("SIP/nirmal-00000007", "Check Database Exist") in new stack
    -- Executing [*300@phones:2] GotoIf("SIP/nirmal-00000007", "0?*300-logout,1:*300-login,1") in new stack
    -- Goto (phones,*300-login,1)
    -- Executing [*300-login@phones:1] NoOp("SIP/nirmal-00000007", "James Not Exists,Login") in new stack
    -- Executing [*300-login@phones:2] AddQueueMember("SIP/nirmal-00000007", "support,SIP/james") in new stack
[Aug 24 17:09:30] NOTICE[3089][C-00000008]: app_queue.c:7797 aqm_exec: Added interface 'SIP/james' to queue 'suppo
rt'
    -- Executing [*300-login@phones:3] Playback("SIP/nirmal-00000007", "beep") in new stack
    -- <SIP/nirmal-00000007> Playing 'beep.gsm' (language 'en')
    -- Executing [*300-login@phones:4] Hangup("SIP/nirmal-00000007", "") in new stack
  == Spawn extension (phones, *300-login, 4) exited non-zero on 'SIP/nirmal-00000007'
ubuntu*CLI>
```

# queue show support

```
root@ubuntu: ~
root@ubuntu:~# asterisk -rvvv
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1253)
ubuntu*CLI> queue show support
support has 0 calls (max unlimited) in 'ringall' strategy (0s holdtime, 0s talktime), W:0, C:0, A:0, SL:0.0% withi
n 0s
   Members:
      SIP/james (ringinuse enabled) (dynamic) (Not in use) has taken no calls yet
      SIP/nirmal (ringinuse enabled) (Not in use) has taken no calls yet
   No Callers

ubuntu*CLI>
```

Enhancement:- Above example , anyone can add/ remove any member. To avoid this problem, add condition in dialplans (condition or subroutine)

**Introducing Custom Prompts**

We have two technique for recording prompt

- Using your phone and call asterisk system and record
- Use third party software and copy recorded file to corresponding folders (we recommend software called Audacity)

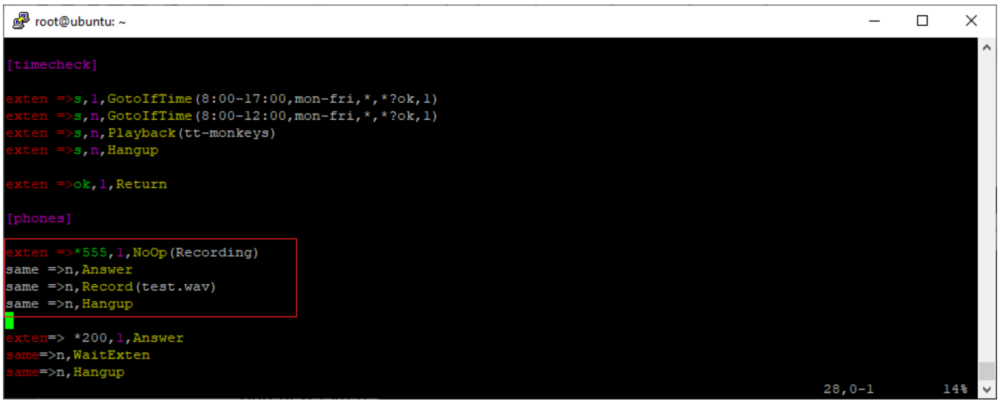
Using Your Phone

1. Open dial plan and add extension

```
#vi /etc/asterisk/extensions.conf
```

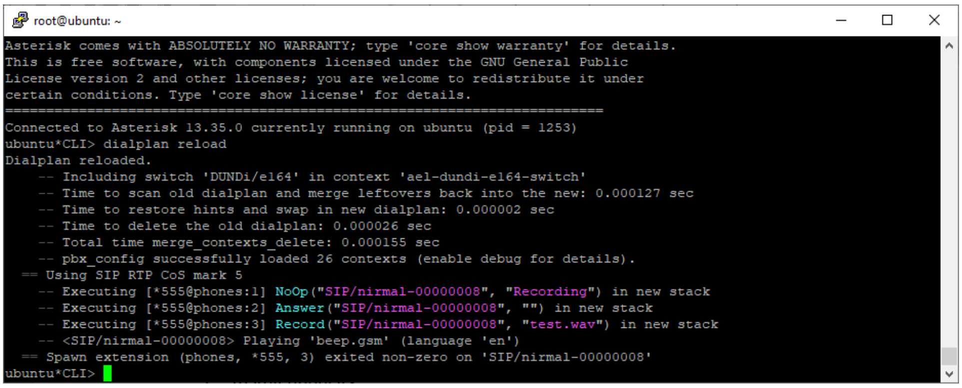
   Add new exten

```
exten =>*555,1,NoOp(Recording)
same =>n,Answer
same =>n,Record(test.wav)
same =>n,Hangup
```



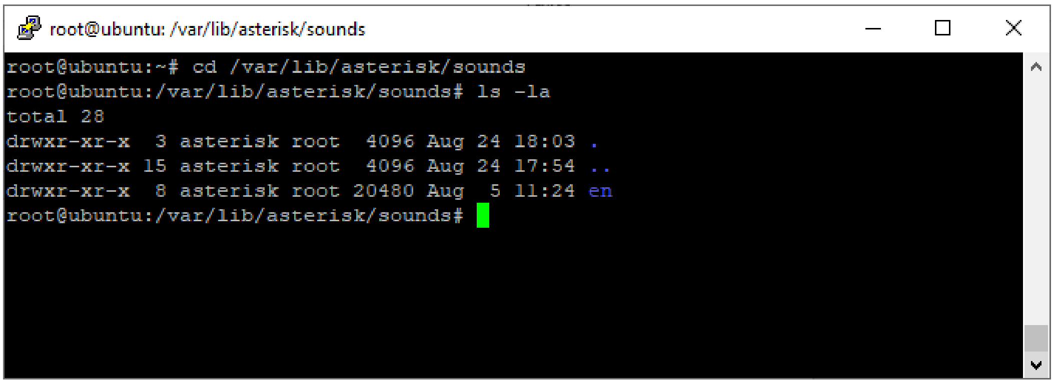

2. In asterisk console, reload dial plan and Dial *555

```
#asterisk -rvvv
#dialplan reload
```




3. Next check recorded files in asterisk

```
#cd /var/lib/asterisk/sounds
#ls -la
```




No Microphone * check this and update screen later

Using third party software - Audacity

1. Install audacity (https://www.audacityteam.org/download/)
2. Record and copy files to asterisk

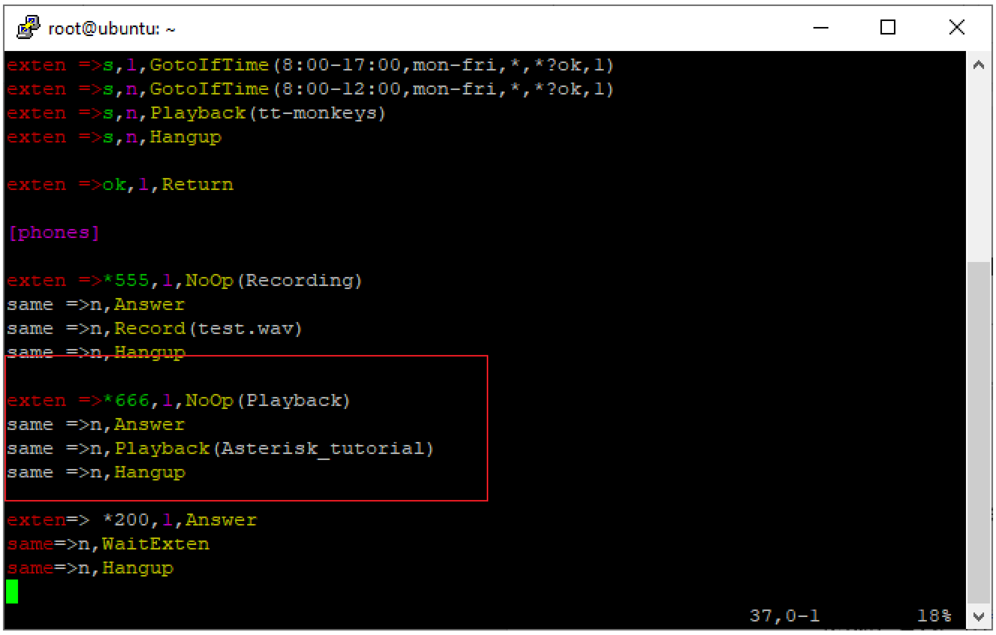
**Installing Your Custom Prompts**

1. Record your prompt (recommend Audacity, and Project rate is 8000, save it
2. Copy to `/var/lib/asterisk/sounds`
3. Add context in dialplan

   `#vi /etc/asterisk/extensions.conf`

   Add New exten

```
exten =>*666,1,NoOp(Playback)
same =>n,Answer
same =>n,Playback(Asterisk_tutorial)
same =>n,Hangup
```




4. In asterisk console, Reload dialplan

   #asterisk -rvvv
   #dial plan reload

5. Dial *666 and listen the recording

## Interactive Voice Response (IVR)

An interactive voice response system lets computer systems interact with telephone callers, who provide input to the system either by pressing the keypad on their telephone set or by saying something (natural language speech recognition). Most IVR systems provide selection menus for routing calls without requiring operator intervention, but modern IVR systems can also be very complex applications that handle information or control equipment.

The basic principle common to all IVR systems, however, is that the caller is read a menu and chooses options from that menu to perform actions, or, alternatively, enters information (in numerical format, through pressing the keypad).

Systems vary in their complexity. The most advanced generate spoken text "on-the-fly" using text-to-speech (TTS) systems and accept spoken user input with speech recognition. When properly implemented they can provide a high level of user-friendliness, but implementation is so complex that they are rarely used, except in larger organizations.
The simplest form of IVR is also the most common. Pre-recorded messages are played to the caller; the caller responds with DTMF keypad input, which Asterisk can recognize easily in the default install.

**Implement Basic IVR Menu: - Simple IVR**

1. Record your IVR prompt and copy to asterisk (`/var/lib/asterisk/sounds)`
2. Go to dialplan config file and add below code

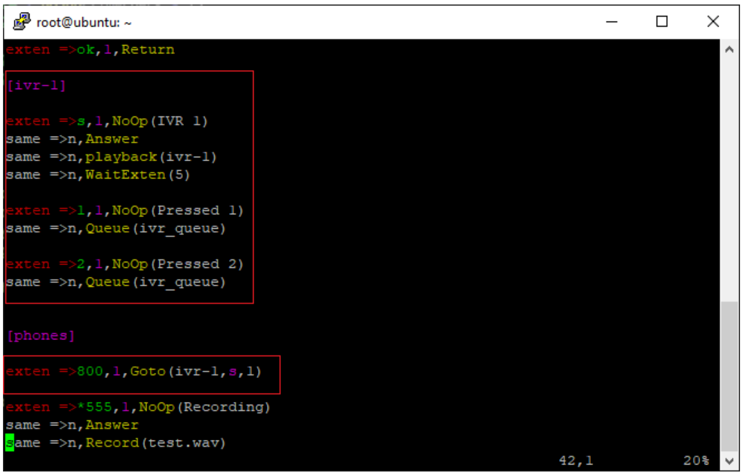
```
#vi /etc/asterisk/extensions.conf
```

   Add In [phone] context – asper above settings

```
exten =>800,1,Goto(ivr-1,s,1)
```

   Create new context: - it is very import

```
[ivr-1]

exten =>s,1,NoOp(IVR 1)
same =>n,Answer
same =>n,playback(ivr-1)
same =>n,WaitExten(5)

exten =>1,1,NoOp(Pressed 1)
same =>n,Queue(ivr_queue)

exten =>2,1,NoOp(Pressed 2)
same =>n,Queue(ivr_queue)
```



```
root@ubuntu: ~
exten =>ok,1,Return

[ivr-1]

exten =>s,1,NoOp(IVR 1)
same =>n,Answer
same =>n,playback(ivr-1)
same =>n,WaitExten(5)

exten =>1,1,NoOp(Pressed 1)
same =>n,Queue(ivr_queue)

exten =>2,1,NoOp(Pressed 2)
same =>n,Queue(ivr_queue)


[phones]

exten =>800,1,Goto(ivr-1,s,1)

exten =>*555,1,NoOp(Recording)
same =>n,Answer
same =>n,Record(test.wav)
                                          42,1          20%
```

3. In asterisk console, Reload dialplan

```
#asterisk -rvvv
#dialplan reload
```

4. Call 800

```
  == Using SIP RTP CoS mark 5
    -- Executing [800@phones:1] Goto("SIP/mathias-0000003f", "ivr-1,s,1") in new stack
    -- Goto (ivr-1,s,1)
    -- Executing [s@ivr-1:1] NoOp("SIP/mathias-0000003f", "IVR 1") in new stack
    -- Executing [s@ivr-1:2] Answer("SIP/mathias-0000003f", "") in new stack
    -- Executing [s@ivr-1:3] Playback("SIP/mathias-0000003f", "ivr_q1") in new stack
    -- <SIP/mathias-0000003f> Playing 'ivr_q1.slin' (language 'en')
    -- Executing [s@ivr-1:4] WaitExten("SIP/mathias-0000003f", "5") in new stack
  == CDR updated on SIP/mathias-0000003f
    -- Executing [1@ivr-1:1] NoOp("SIP/mathias-0000003f", "Pressed 1") in new stack
    -- Executing [1@ivr-1:2] Queue("SIP/mathias-0000003f", "test") in new stack
[Jun 24 14:36:30] WARNING[6548][C-00000036]: app_queue.c:7141 queue_exec: Unable to join queue 'test
'

    -- Auto fallthrough, channel 'SIP/mathias-0000003f' status is 'UNKNOWN'
```

Replace this screens after record your ivr files

**Advance Asterisk IVR Configurations**

After Pressing Invalid extension or Handling Timeout Error

1. Go to dialplan config file and add below code

   ```
   #vi /etc/asterisk/extensions.conf
   ```

   Add extension in `[ivr-1]`

   ```
   exten =>i,1,NoOp(Invalid)
   same =>n,playback(ivr_wrong_key)
   same =>n,Goto(s,1)

   exten =>t,1,NoOp(Timeout)
   same =>n,Queue(test)
   ```

   ```
   root@ubuntu: ~
   exten =>ok,1,Return

   [ivr-1]

   exten =>s,1,NoOp(IVR 1)
   same =>n,Answer
   same =>n,playback(ivr-1)
   same =>n,WaitExten(5)

   exten =>1,1,NoOp(Pressed 1)
   same =>n,Queue(ivr_queue)

   exten =>2,1,NoOp(Pressed 2)
   same =>n,Queue(ivr_queue)

   exten =>i,1,NoOp(Invalid)
   same =>n,playback(ivr_wrong_key)
   same =>n,Goto(s,1)

   exten =>t,1,NoOp(Timeout)
   same =>n,Queue(ivr_queue)

   [phones]

   exten =>800,1,Goto(ivr-1,s,1)

   exten =>*555,1,NoOp(Recording)
                                                  47,1          19%
   ```

2. In asterisk console, Reload dialplan

   ```
   #asterisk -rvvv
   #dialplan reload
   ```

3. Call 800

   ```
   == Using SIP RTP CoS mark 5
     -- Executing [800@phones:1] Goto("SIP/mathias-00000042", "ivr-1,s,1") in new stack
     -- Goto (ivr-1,s,1)
     -- Executing [s@ivr-1:1] NoOp("SIP/mathias-00000042", "IVR 1") in new stack
     -- Executing [s@ivr-1:2] Answer("SIP/mathias-00000042", "") in new stack
     -- Executing [s@ivr-1:3] Playback("SIP/mathias-00000042", "ivr_q1") in new stack
     -- <SIP/mathias-00000042> Playing 'ivr_q1.slin' (language 'en')
   ```

   After pressing Invalid option 3

```
    -- <SIP/mathias-00000042> Playing 'ivr_q1.slin' (language 'en')
    -- Executing [s@ivr-1:4] WaitExten("SIP/mathias-00000042", "5") in new stack
    -- Invalid extension '3' in context 'ivr-1' on SIP/mathias-00000042
  == CDR updated on SIP/mathias-00000042
    -- Executing [i@ivr-1:1] NoOp("SIP/mathias-00000042", "Invalid") in new stack
    -- Executing [i@ivr-1:2] Playback("SIP/mathias-00000042", "ivr_wrong_key") in new sta
ck
[Jun 26 10:57:01] WARNING[7247][C-00000039]: file.c:701 ast_openstream_full: File ivr_wro
ng_key does not exist in any format
[Jun 26 10:57:01] WARNING[7247][C-00000039]: file.c:1017 ast_streamfile: Unable to open i
vr_wrong_key (format (ulaw)): No such file or directory
[Jun 26 10:57:01] WARNING[7247][C-00000039]: app_playback.c:484 playback_exec: ast_stream
file failed on SIP/mathias-00000042 for ivr_wrong_key
    -- Executing [i@ivr-1:3] Goto("SIP/mathias-00000042", "s,1") in new stack
    -- Goto (ivr-1,s,1)
    -- Executing [s@ivr-1:1] NoOp("SIP/mathias-00000042", "IVR 1") in new stack
    -- Executing [s@ivr-1:2] Answer("SIP/mathias-00000042", "") in new stack
    -- Executing [s@ivr-1:3] Playback("SIP/mathias-00000042", "ivr_q1") in new stack
    -- <SIP/mathias-00000042> Playing 'ivr_q1.slin' (language 'en')
```

Timeout

```
file failed on SIP/mathias-00000043 for ivr_wrong_key
    -- Executing [i@ivr-1:3] Goto("SIP/mathias-00000043", "s,1") in new stack
    -- Goto (ivr-1,s,1)
    -- Executing [s@ivr-1:1] NoOp("SIP/mathias-00000043", "IVR 1") in new stack
    -- Executing [s@ivr-1:2] Answer("SIP/mathias-00000043", "") in new stack
    -- Executing [s@ivr-1:3] Playback("SIP/mathias-00000043", "ivr_q1") in new stack
    -- <SIP/mathias-00000043> Playing 'ivr_q1.slin' (language 'en')
    -- Executing [s@ivr-1:4] WaitExten("SIP/mathias-00000043", "5") in new stack
    -- Timeout on SIP/mathias-00000043, going to 't'
    -- Executing [t@ivr-1:1] NoOp("SIP/mathias-00000043", "timeout") in new stack
    -- Executing [t@ivr-1:2] Queue("SIP/mathias-00000043", "test") in new stack
[Jun 26 11:00:32] WARNING[7251][C-0000003a]: app_queue.c:7141 queue_exec: Unable to join
queue 'test'
    -- Auto fallthrough, channel 'SIP/mathias-00000043' status is 'UNKNOWN'
```

Replace this screen after record your IVR files

**IVR Menu Looping**

Solution: - If user not pressed anything, repeat IVR menu again

1. Go to dialplan config file and add below code

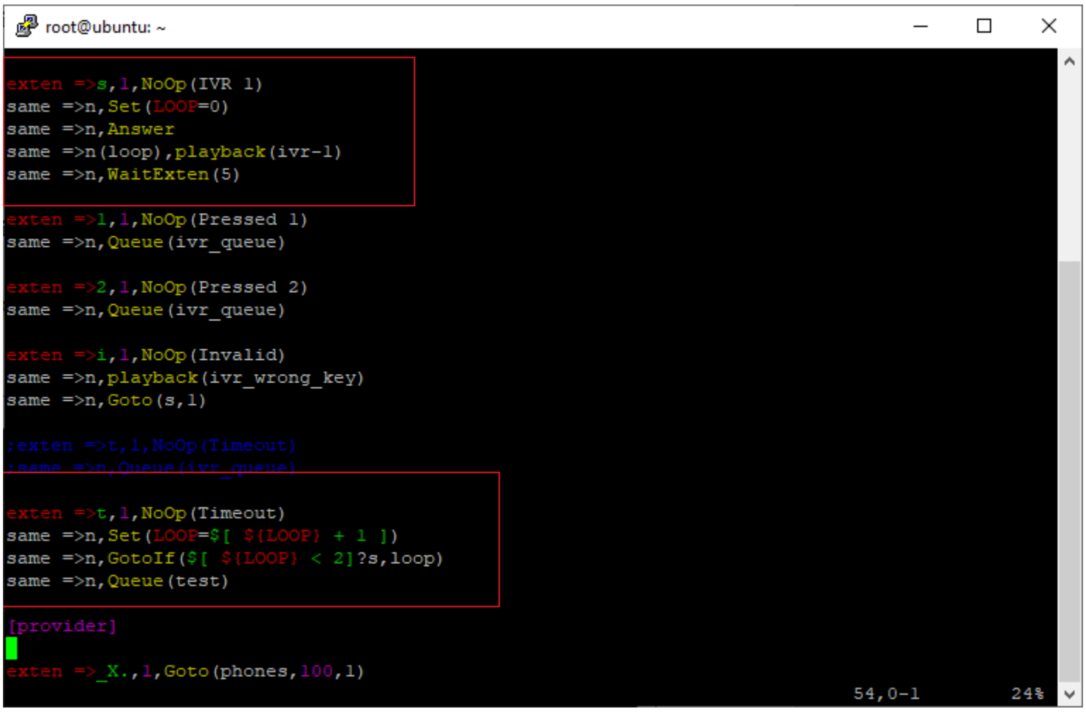
```
#vi /etc/asterisk/extensions.conf
```

In `[ivr-1] context extension "s"`

```
exten =>s,1,NoOp(IVR 1)
same =>n,Set(LOOP=0)
same =>n,Answer
same =>n(loop),playback(ivr-1) ;Label
same =>n,WaitExten(5)
```

In `[ivr-1] context extension "t"`

```
exten =>t,1,NoOp(Timeout)
same => Set(LOOP=$[ ${LOOP} + 1 ]
same -> GotoIf($[ ${LOOP} < 2]?s,loop)
same =>n,Queue(test)
```




2. In asterisk console, Reload dialplan

```
#asterisk -rvvv
#dialplan reload
```

3. Call 800

```
== Using SIP RTP CoS mark 5
  -- Executing [800@phones:1] Goto("SIP/mathias-00000048", "ivr-1,s,1") in new stack
  -- Goto (ivr-1,s,1)
  -- Executing [s@ivr-1:1] NoOp("SIP/mathias-00000048", "IVR 1") in new stack
  -- Executing [s@ivr-1:2] Set("SIP/mathias-00000048", "LOOP=0") in new stack
  -- Executing [s@ivr-1:3] Answer("SIP/mathias-00000048", "") in new stack
  -- Executing [s@ivr-1:4] Playback("SIP/mathias-00000048", "ivr_q1") in new stack
  -- <SIP/mathias-00000048> Playing 'ivr_q1.slin' (language 'en')
  -- Executing [s@ivr-1:5] WaitExten("SIP/mathias-00000048", "5") in new stack
  -- Timeout on SIP/mathias-00000048, going to 't'
  -- Executing [t@ivr-1:1] NoOp("SIP/mathias-00000048", "timeout") in new stack
  -- Executing [t@ivr-1:2] Set("SIP/mathias-00000048", "LOOP=1") in new stack
  -- Executing [t@ivr-1:3] GotoIf("SIP/mathias-00000048", "1?s,loop") in new stack
  -- Goto (ivr-1,s,4)
  -- Executing [s@ivr-1:4] Playback("SIP/mathias-00000048", "ivr_q1") in new stack
  -- <SIP/mathias-00000048> Playing 'ivr_q1.slin' (language 'en')
  -- Executing [s@ivr-1:5] WaitExten("SIP/mathias-00000048", "5") in new stack
```

```
    -- Executing [s@ivr-1:5] WaitExten("SIP/mathias-00000048", "5") in new stack
    -- Timeout on SIP/mathias-00000048, going to 't'
    -- Executing [t@ivr-1:1] NoOp("SIP/mathias-00000048", "timeout") in new stack
    -- Executing [t@ivr-1:2] Set("SIP/mathias-00000048", "LOOP=1") in new stack
    -- Executing [t@ivr-1:3] GotoIf("SIP/mathias-00000048", "1?s,loop") in new stack
    -- Goto (ivr-1,s,4)
    -- Executing [s@ivr-1:4] Playback("SIP/mathias-00000048", "ivr_q1") in new stack
    -- <SIP/mathias-00000048> Playing 'ivr_q1.slin' (language 'en')
    -- Executing [s@ivr-1:5] WaitExten("SIP/mathias-00000048", "5") in new stack
    -- Timeout on SIP/mathias-00000048, going to 't'
    -- Executing [t@ivr-1:1] NoOp("SIP/mathias-00000048", "timeout") in new stack
    -- Executing [t@ivr-1:2] Set("SIP/mathias-00000048", "LOOP=2") in new stack
    -- Executing [t@ivr-1:3] GotoIf("SIP/mathias-00000048", "0?s,loop") in new stack
    -- Executing [t@ivr-1:4] Queue("SIP/mathias-00000048", "test") in new stack
[Jun 26 11:24:44] WARNING[7277][C-0000003f]: app_queue.c:7141 queue_exec: Unable to join
queue 'test'
    -- Auto fallthrough, channel 'SIP/mathias-00000048' status is 'UNKNOWN'
```

Replace this screen after record your IVR files

## Session Initiation Protocol - SIP

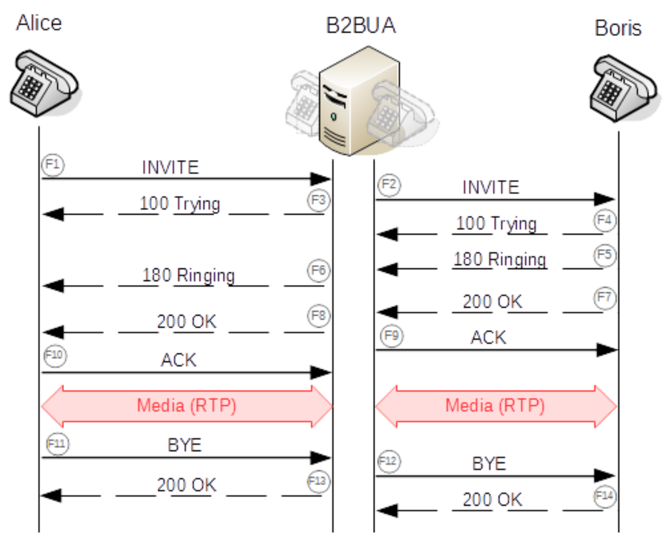
The **Session Initiation Protocol (SIP)** is a signaling protocol used for initiating, maintaining, and terminating real-time sessions that include voice, video and messaging applications. SIP is used for signaling and controlling multimedia communication sessions in applications of Internet telephony for voice and video calls, in private IP telephone systems, in instant messaging over **Internet Protocol** (IP) networks as well as mobile phone calling over LTE (VoLTE).

The protocol defines the specific format of messages exchanged and the sequence of communications for cooperation of the participants. SIP is a text-based protocol, incorporating many elements of the **Hypertext Transfer Protocol (HTTP)** and the **Simple Mail Transfer Protocol (SMTP)**. A call established with SIP may consist of multiple media streams, but no separate streams are required for applications, such as text messaging, that exchange data as payload in the SIP message.

SIP works in conjunction with several other protocols that specify and carry the session media. Most commonly, media type and parameter negotiation and media setup are performed with the Session Description Protocol (SDP), which is carried as payload in SIP messages. SIP is designed to be independent of the underlying transport layer protocol, and can be used with the User Datagram Protocol (UDP), the Transmission Control Protocol (TCP), and the Stream Control Transmission Protocol (SCTP). For secure transmissions of SIP messages over insecure network links, the protocol may be encrypted with Transport Layer Security (TLS). For the transmission of media streams (voice, video) the SDP payload carried in SIP messages typically employs the Real-time Transport Protocol (RTP) or the Secure Real-time Transport Protocol (SRTP).

Session Initiation Protocol (SIP) is a signaling protocol used for initiating, maintaining, modifying and terminating real-time sessions that involve video, voice, messaging and other communications applications and services between two or more endpoints on IP networks.

Native support for mobility, interoperability and multimedia were among the drivers behind SIP's development. SIP complements other communications protocols, such as **Real-Time Transport Protocol (RTP)** and **Real-Time Streaming Protocols (RTSP)**, used in IP-based session



**SIP Debugging**

Go to asterisk console and enable debug

```
#asterisk -rvvv
#sip set debug on
```

root@ubuntu: ~

```
root@ubuntu:~# asterisk -rvvv
Asterisk 13.35.0, Copyright (C) 1999 - 2014, Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for detail
s.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
Connected to Asterisk 13.35.0 currently running on ubuntu (pid = 1227)
ubuntu*CLI> sip set debug on
SIP Debugging enabled
ubuntu*CLI>
```

Dial 100 → Answer Call → Disconnect Call and See asterisk Console (it will record all information)

root@ubuntu: ~

```
<------------->
--- (16 headers 0 lines) ---
Sending to 192.168.11.46:5060 (NAT)
Creating new subscription
Sending to 192.168.11.46:5060 (NAT)
sip_route_dump: route/path hop: <sip:james@123.50.200.187:5060;transport=UDP>
Found peer 'james' for 'james' from 192.168.11.46:5060

<--- Transmitting (no NAT) to 192.168.11.46:5060 --->
SIP/2.0 401 Unauthorized
Via: SIP/2.0/UDP 123.50.200.187:5060;branch=z9hG4bK-d8754z-78e31817ab4d3051-1---
d8754z-;received=192.168.11.46
From: <sip:james@192.168.11.47;transport=UDP>;tag=9365dc67
To: <sip:james@192.168.11.47;transport=UDP>;tag=as59743906
Call-ID: Njg3OWI0ZWM2MTczNWM2ZGUwYTJhYWZlYThjZGZlYmY.
CSeq: 1 SUBSCRIBE
Server: Asterisk PBX 13.35.0
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE, NOTIFY, INFO, PUBLIS
H, MESSAGE
Supported: replaces, timer
WWW-Authenticate: Digest algorithm=MD5, realm="asterisk", nonce="73810ef4"
Content-Length: 0


<------------>
Scheduling destruction of SIP dialog 'Njg3OWI0ZWM2MTczNWM2ZGUwYTJhYWZlYThjZGZlYm
Y.' in 32000 ms (Method: SUBSCRIBE)

<--- SIP read from UDP:192.168.11.46:5060 --->
```

## Using tcpdump

1. First check network details

   #ip addr

```
root@ubuntu: ~
root@ubuntu:~# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:99:29:3d brd ff:ff:ff:ff:ff:ff
    inet 192.168.11.47/24 brd 192.168.11.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe99:293d/64 scope link
       valid_lft forever preferred_lft forever
root@ubuntu:~#
```

2. Write command for Record sip details and create file

   #tcpdump -i eth0 -w test.pcap

   To record all details

   #tcpdump -i eth0 -s 0 -w test.pcap

3. Dial 100 → Answer Call → Disconnect Call And Ctrl+c

```
root@ubuntu: ~
root@ubuntu:~# tcpdump -i eth0 -s 0 -w test.pcap
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
^C954 packets captured
963 packets received by filter
0 packets dropped by kernel
root@ubuntu:~#
```

4. Check the file

   #ls -la

```
root@ubuntu: ~
drwxr-xr-x  2 nirmal nirmal     4096 Aug  4 14:58 Downloads
-rw-r--r--  1 nirmal nirmal     8980 Aug  4 14:52 examples.desktop
drwx------  3 nirmal nirmal     4096 Aug 26 10:00 .gconf
-rw-------  1 nirmal nirmal     6042 Aug 26 10:00 .ICEauthority
drwx------  3 nirmal nirmal     4096 Aug  4 14:58 .local
drwx------  5 nirmal nirmal     4096 Aug  6 11:36 .mozilla
drwxr-xr-x  2 nirmal nirmal     4096 Aug  4 14:58 Music
drwxr-xr-x  2 nirmal nirmal     4096 Aug  4 14:58 Pictures
-rw-r--r--  1 nirmal nirmal      675 Aug  4 14:52 .profile
drwxr-xr-x  2 nirmal nirmal     4096 Aug  4 14:58 Public
-rw-r--r--  1 root   root       1510 Aug  6 15:07 sip.conf
drwxr-xr-x  2 nirmal nirmal     4096 Aug  4 14:58 Templates
-rw-r--r--  1 root   root     227320 Aug 26 11:32 test.pcap
drwxr-xr-x  2 nirmal nirmal     4096 Aug  4 14:58 Videos
-rw-------  1 root   root       7801 Aug 25 17:26 .viminfo
-rw-------  1 nirmal nirmal       51 Aug 26 10:00 .Xauthority
-rw-------  1 nirmal nirmal      711 Aug 26 10:00 .xsession-errors
-rw-------  1 nirmal nirmal     1490 Aug 25 18:21 .xsession-errors.old
root@ubuntu:~#
```

## Wireshark

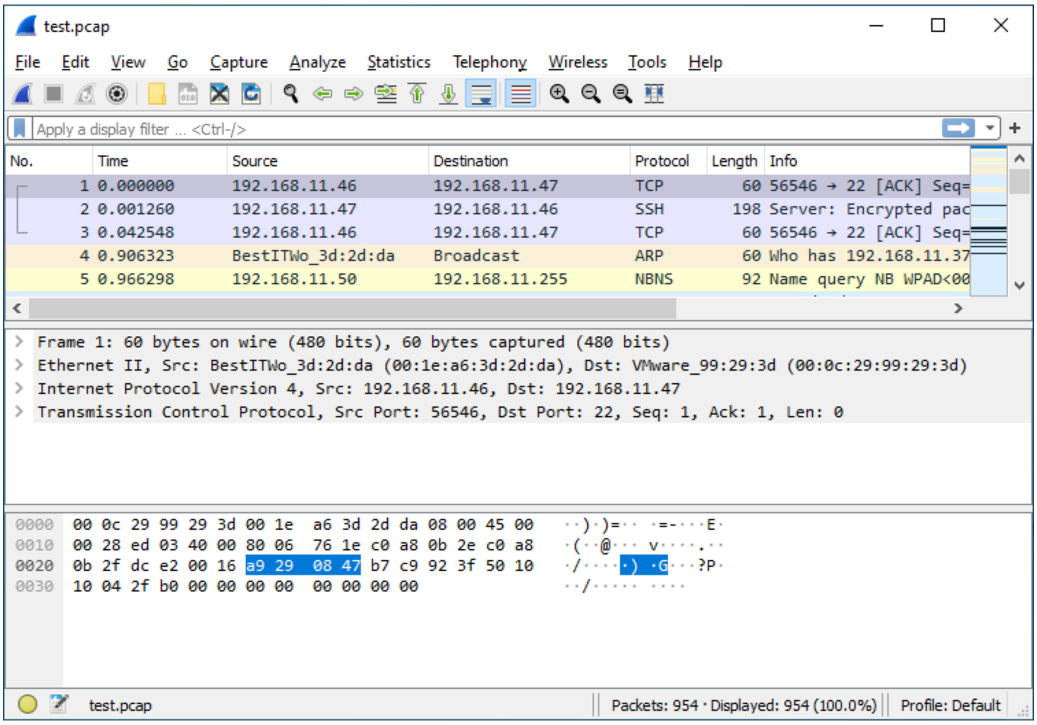
**Wireshark** is a free and open-source packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education. Originally named **Ethereal**, the project was renamed Wireshark in May 2006 due to trademark issues.

Wireshark is cross-platform, using the Qt widget toolkit in current releases to implement its user interface, and using pcap to capture packets; it runs on Linux, macOS, BSD, Solaris, some other Unix-like operating systems, and Microsoft Windows. There is also a terminal-based (non-GUI) version called TShark. Wireshark, and the other programs distributed with it such as TShark, are free software, released under the terms of the GNU General Public License.

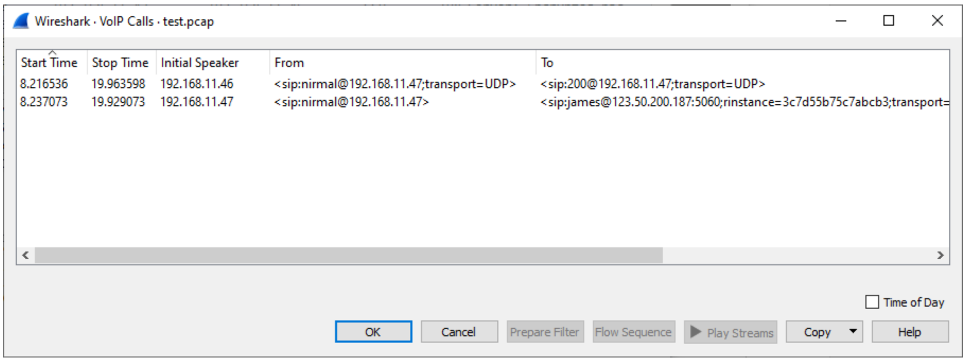
In the field of computer network administration, pcap is an application programming interface (API) for capturing network traffic. While the name is an abbreviation of packet capture, that is not the API's proper name. Unix-like systems implement pcap in the libpcap library; for Windows, there is a port of libpcap named WinPcap that is no longer supported or developed, and a port named Npcap for Windows 7 and later that is still supported.

To download: https://www.wireshark.org/#download

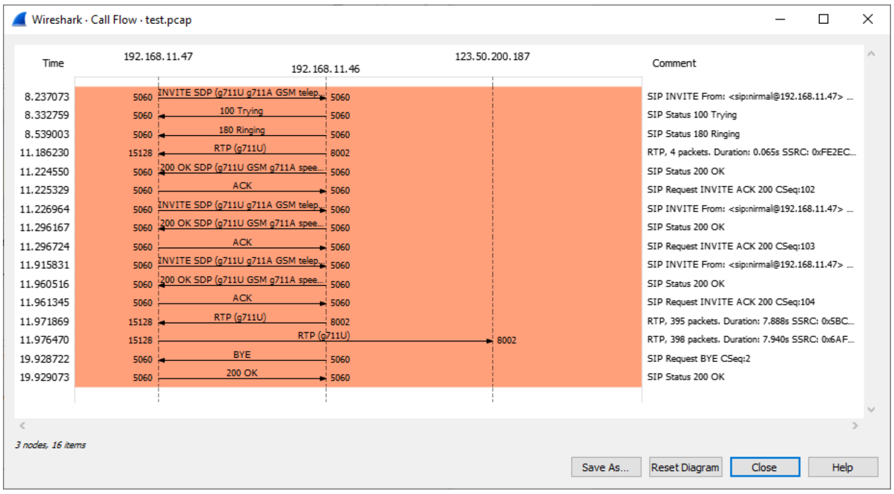
1. Copy .pcap file to local machine
2. Download and install Wireshark
3. Open .pcap file in Wireshark



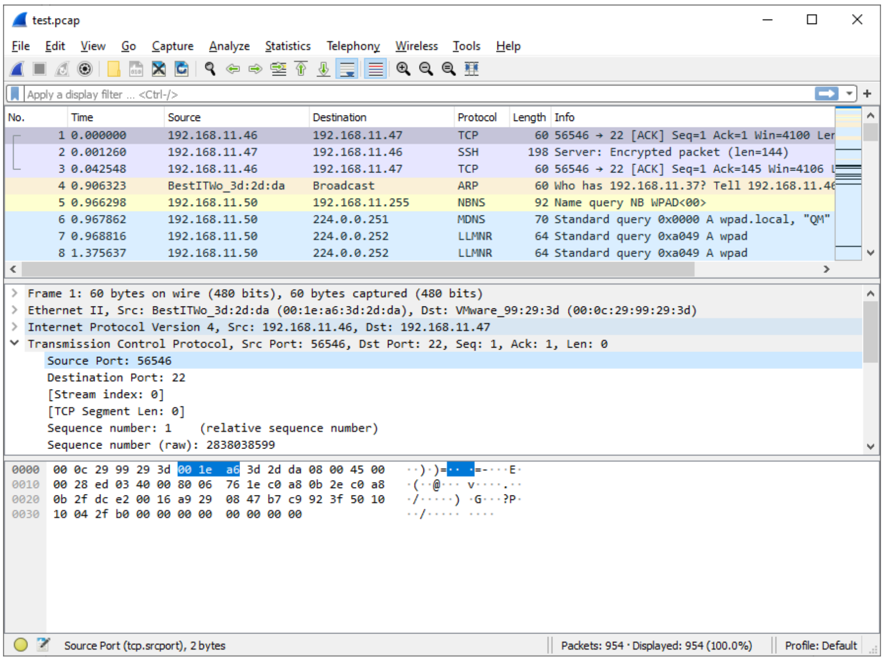

4. You can search using Telephony tool bar, choose VoIP Calls (anything you can search)



5. Choose one and click Flow Sequence, it will show all details about selected records



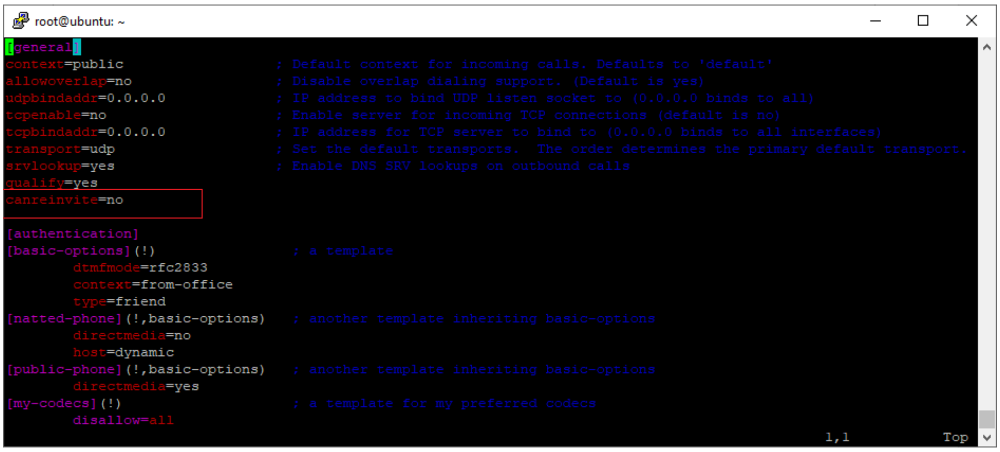
See mail windows for the details



## SIP & RTP Configuration

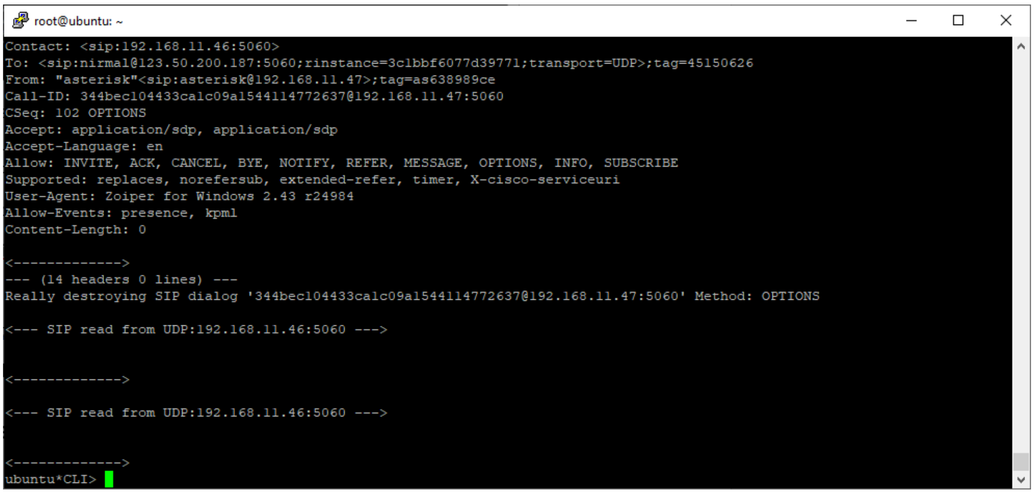
1. Open /etc/asterisk/sip.conf and add parameter called canreinvite=no in general context (you can add particular context also)

```
#vi /etc/asterisk/sip.conf
```



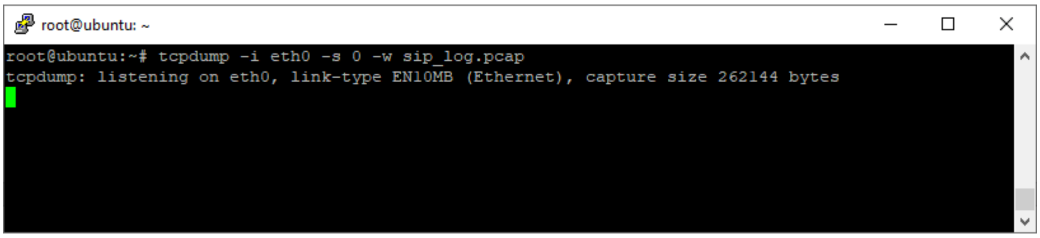

2. Go to asterisk console, Reload sip

```
#asterisk -rvvv
#sip reload
```

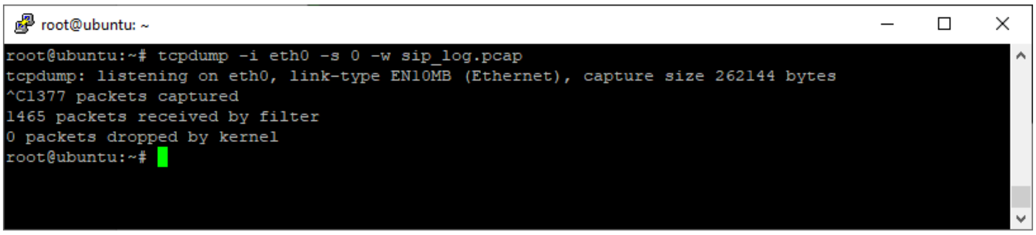



3. Exit asterisk console and Use tcpdump

```
#tcpdump -i eth0 -s 0 -w sip_log.pcap
```

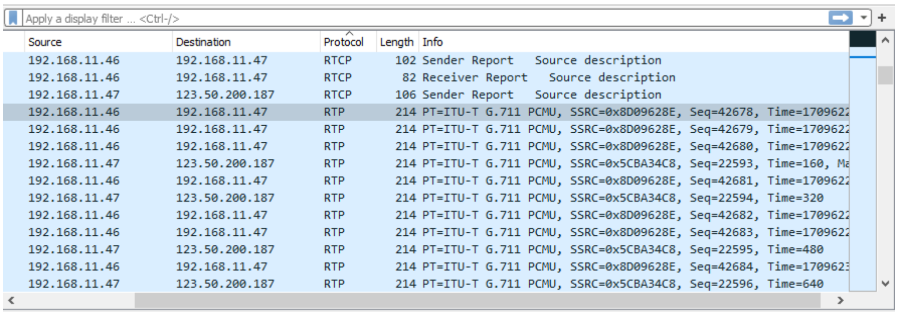


4. Make call → disconnect and ctrl+c

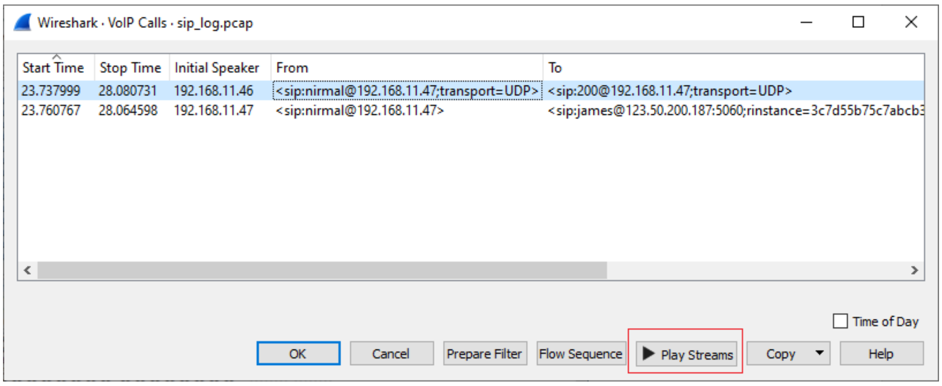


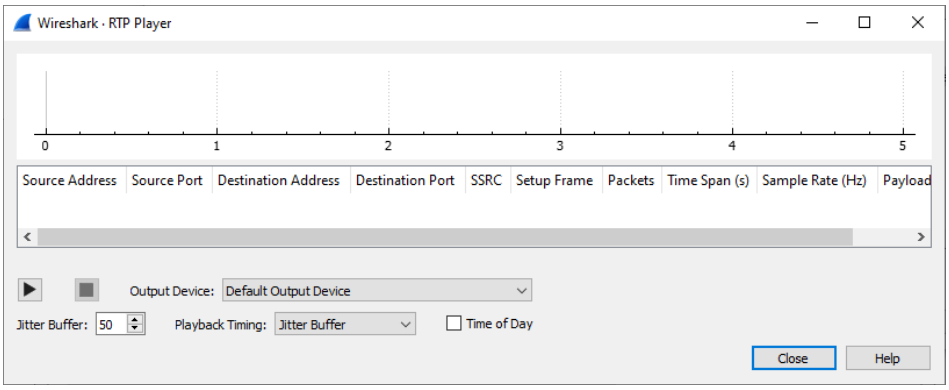

5. Copy sip_log.pcap to local machine for debugging

6. Open copied file in Wireshark and see RTP stream (follow Wireshark steps – mentioned above)




7. Use Player / Pay stream → decode and Play audio

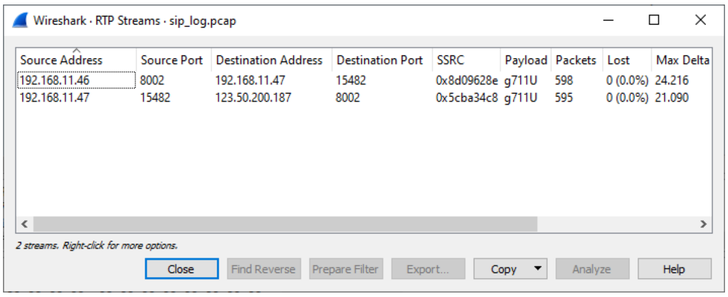


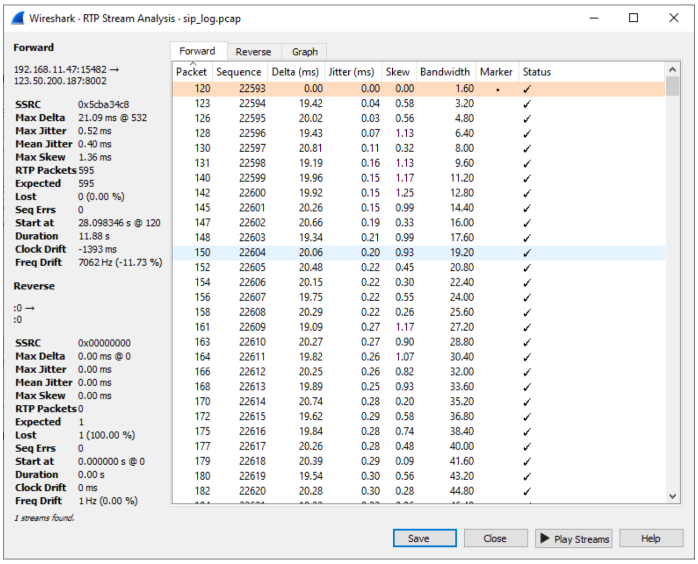

Audio not recorded

## Wireshark RTP Audio Debug

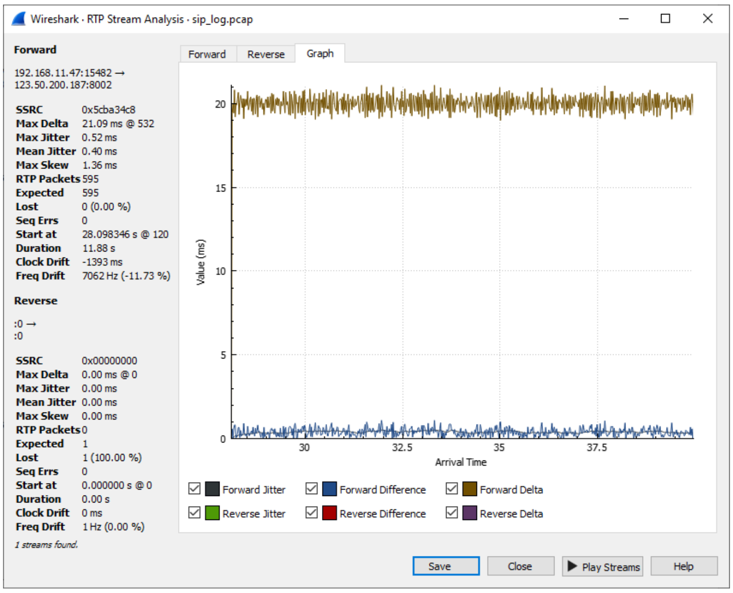
1. Go to Wireshark → Telephony → RTP → RTP Streams




2. Choose one and click **Analysis**




You can listen audio(Play Streams) or See the graph (Graph Tab)



### SIP Providers

A **SIP provider** (Session Initiation Protocol) is any telecommunications company which provides SIP trunking to customers, usually businesses.

Many companies provide SIP "termination" (outbound calling) and "origination" (inbound calling, usually with a plain old telephone service (POTS) phone number, called a direct inward dialing (DID). Most companies that provide one also provide the other.

Outbound (termination) rates vary from provider to provider and can often depend on the type of number being called as well as the geographical destination.

Inbound calling prices are more varied. Some providers offer flat rate pricing "per channel" (simultaneous call leg) while some offer an unlimited number of channels but a low, fixed per-minute rate. Some providers offer a choice of plan.

### SIP Provider Registration

https://www.flowroute.com/

### SIP Provider Registration The First Attempt – Not Tested

1. Register SIP Provider and get Credential (server, username and password)

2. Asterisk SIP register. The register directive (in sip.conf) tells Asterisk to register itself to a SIP provider.

```
[Syntax]
register => user[:secret[:authuser]]@host[:port][/extension]
```

   If no extension is given, the 's' extension is used. The extension needs to be defined in extensions.conf to be able to accept calls from this SIP provider.

   The register statement must appear in the [general] section of the sip.conf file. If it is not in the general section then it will be ignored, and no error messages will appear.

```
#vi /etc/asterisk/sip.conf
```

   Add

```
register => user:password@host
```


3. In asterisk Reload sip and see registration

```
#sip reload
#sip show registry
```

```
asterisk*CLI> sip show peers
Name/username             Host                                    Dyn Forceport Comedia    ACL Port     Status      Description
james/james               (Unspecified)                            D  Auto (No)  No             0       UNKNOWN
mathias/mathias           (Unspecified)                            D  Auto (No)  No             0       UNKNOWN
outside/outside           (Unspecified)                            D  Auto (No)  No             0       UNKNOWN
3 sip peers [Monitored: 0 online, 3 offline Unmonitored: 0 online, 0 offline]
asterisk*CLI>
asterisk*CLI>
asterisk*CLI> sip show registry
Host                                    dnsmgr Username       Refresh State                Reg.Time
sip.flowroute.com:5060                  N      36839184           105 Registered           Fri, 26 Jun 2015 15:50:33
1 SIP registrations.
```

## Adding a SIP Provider Peer – Not Tested

1. Create new context in sip.conf

```
[provider]
        type=friend
        context=provider
        allow=ulaw,alaw
        secret=XYX@123
        host=sip.floweroute.com
        nat=force_report,comedia
```

```
root@ubuntu: ~
        allow=g729
        allow=gsm
        allow=g723
        allow=ulaw
[ulaw-phone](!)                    ; and another one for ulaw-only
        disallow=all
        allow=ulaw

[provider]
        type=friend
        context=provider
        allow=ulaw,alaw
        secret=XYX@123
        host=sip.floweroute.com
        nat=force_report,comedia

[james]
        type=friend
        context=phones
        allow=ulaw,alaw
        secret=12345678
        host=dynamic
[nirmal]
        type=friend
        context=phones
                                                              43,0-1        72%
```

2. Reload sip and show peers

```
#sip reload
#sip show peers
```

```
asterisk*CLI> sip show peers
Name/username             Host                                    Dyn Forceport Comedia    ACL Port     Status      Descript
ion
james/james               192.168.100.253                          D  Auto (No)  No             55556   OK (1 ms)
mathias/mathias           192.168.100.253                          D  Auto (No)  No             55556   OK (1 ms)
outside/outside           192.168.100.253                          D  Auto (No)  No             55556   OK (1 ms)
provider                  216.115.69.144                              Yes        Yes            5060    OK (171 ms)
4 sip peers [Monitored: 4 online, 0 offline Unmonitored: 0 online, 0 offline]
asterisk*CLI>
```

## SIP Provider Inbound Calls - – Not Tested

1. Go to extensions.conf

   `#vi /etc/asterisk/extensions.conf`

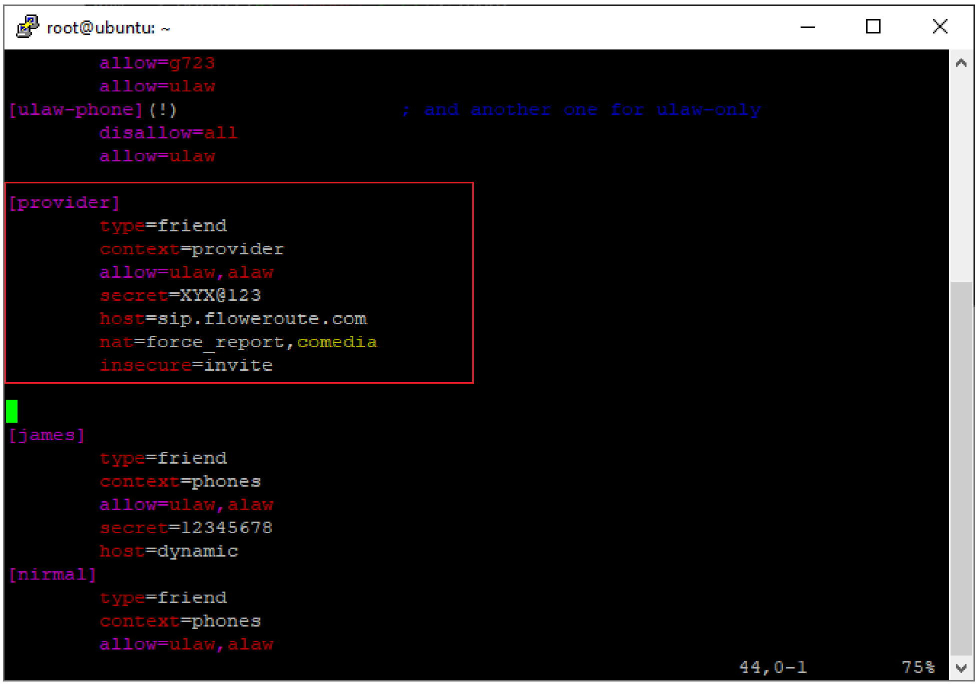
2. Add new extension

   ```
   [provider]

    exten=>_X.,1,Goto(phones,100,1)
   ```

```
root@ubuntu: ~
exten =>i,1,NoOp(Invalid)
same =>n,playback(ivr_wrong_key)
same =>n,Goto(s,1)

;exten =>t,1,NoOp(Timeout)
;same =>n,Queue(ivr_queue)

exten =>t,1,NoOp(Timeout)
same => Set(LOOP=$[ ${LOOP} + 1 ]
same -> GotoIf($[ ${LOOP} < 2]?s,loop)
same =>n,Queue(test)

[provider]

exten =>_X.,1,Goto(phones,100,1)

[phones]

exten =>800,1,Goto(ivr-1,s,1)

exten =>*555,1,NoOp(Recording)
same =>n,Answer
                                                62,1          34%
```

3. Add new line in context [provider] in sip.conf

   `insecure=invite`



```
root@ubuntu: ~
        allow=g723
        allow=ulaw
[ulaw-phone](!)                   ; and another one for ulaw-only
        disallow=all
        allow=ulaw

[provider]
        type=friend
        context=provider
        allow=ulaw,alaw
        secret=XYX@123
        host=sip.floweroute.com
        nat=force_report,comedia
        insecure=invite


[james]
        type=friend
        context=phones
        allow=ulaw,alaw
        secret=12345678
        host=dynamic
[nirmal]
        type=friend
        context=phones
        allow=ulaw,alaw
                                                44,0-1        75%
```

4. In asterisk console, Reload dialplan and sip

```
#asterisk -rvvv
#dialplan reload
#sip reload

#sip set debug peer provider ; for debugging
```

5. Dial Number [incoming] – asper my setting 991123123 - it will ring

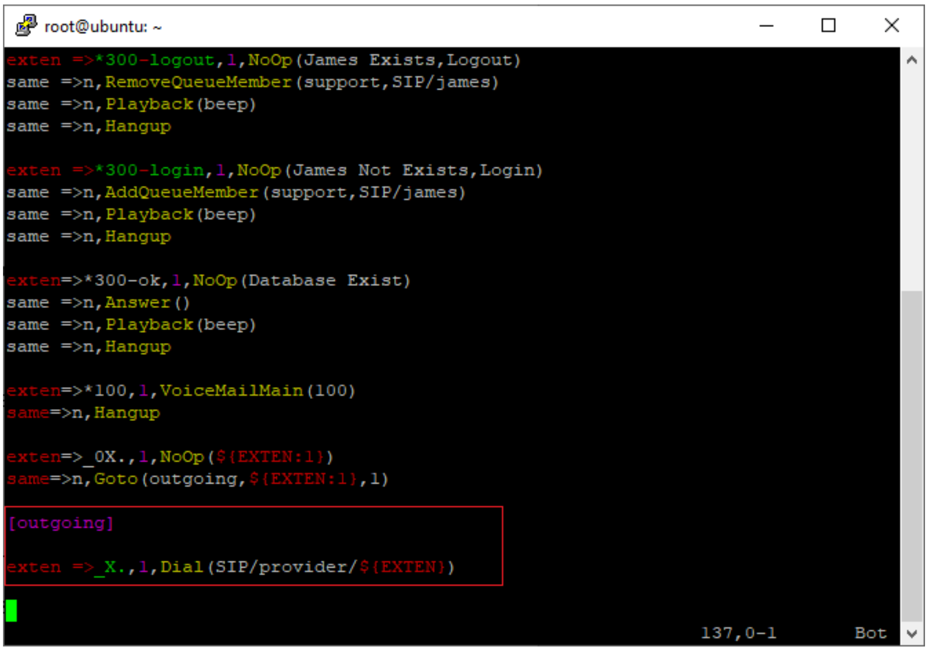
## SIP Provider External Outbound Calls – Not Tested

1. Edit dialplan – [outgoing] context

```
#vi /etc/asterisk/extensions.conf
```

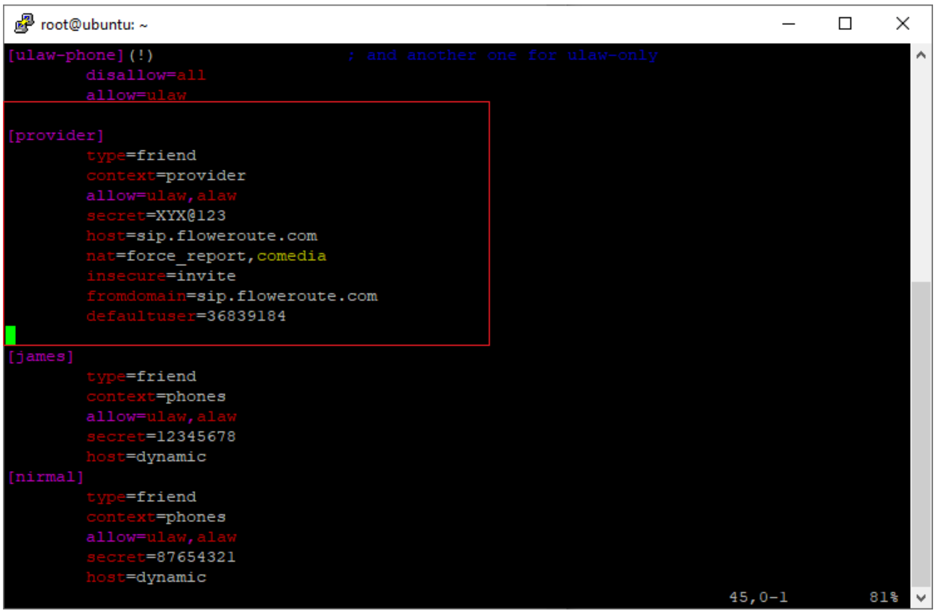
Edit context like this (asper the current settings)

```
exten =>_X.Dial(SIP/provider/${EXTEN})
```




2. Go to sip.conf add new parameters in [provider] context

```
fromdomain=sip.floweroute.com
defaultuser=username
```




3. In Asterisk console, reload dialplan and sip (enable debug if required)

```
#asterisk -rvvv
#dialplan reload
#sip reload

#sip set debug peer provider ; for debugging
```

4. Dial Number

## SIP Provider Caller ID – Not Tested

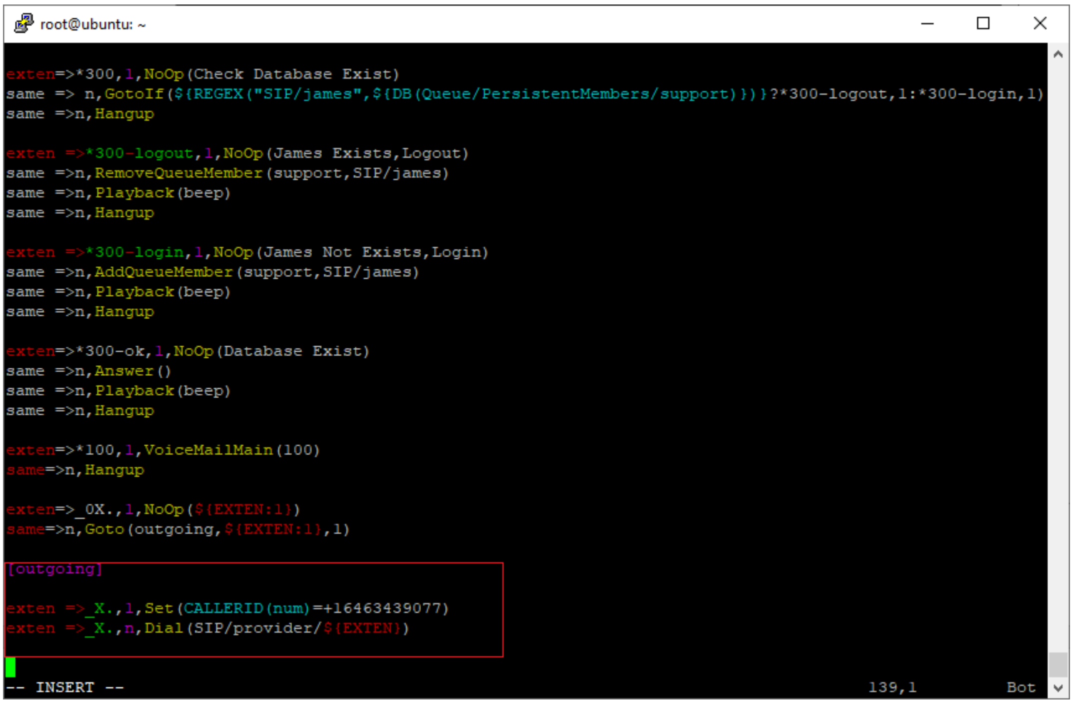
1. Enable debug in asterisk console

```
#asterisk -rvvv
#sip set debug provider
```

   Dial number to see details

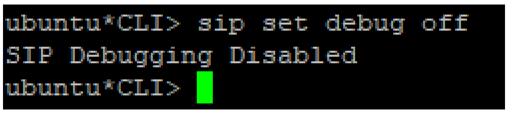

2. Go to extensions.conf, add first line and edit next line ([outgoing])

```
exten=>_X.,1,Set(CALLERID(num)=16463439077)
exten=>_X.,n,Dial(SIP/provider/${EXTEN})
```




3. In asterisk console, reload dialplan and set debug off

```
#asterisk -rvvv
#dialplan reload
#sip set debug off
```




4. Dial and check your phone – it will shows calling number

## Asterisk Logger

In this file, you configure logging to files or to the syslog system. "logger reload" at the CLI will reload the configuration of the logging system.

1. Open logger.conf

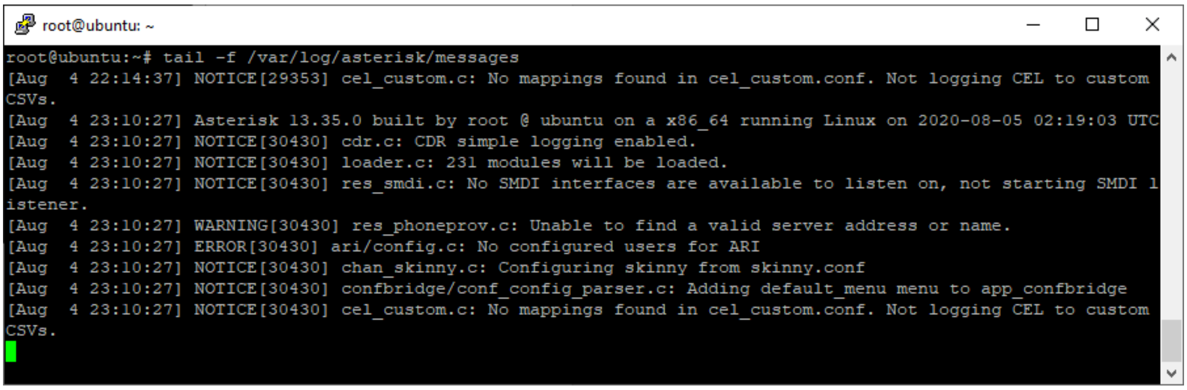
   `#vi /etc/asterisk/logger.conf`

2. Add verbose level in **message** parameter

   `Message=>notice,warning,verbose(3)`


3. Type commend below, dial number and see console. But it will not show any log files

   `#tail -f /var/log/asterisk/messages`

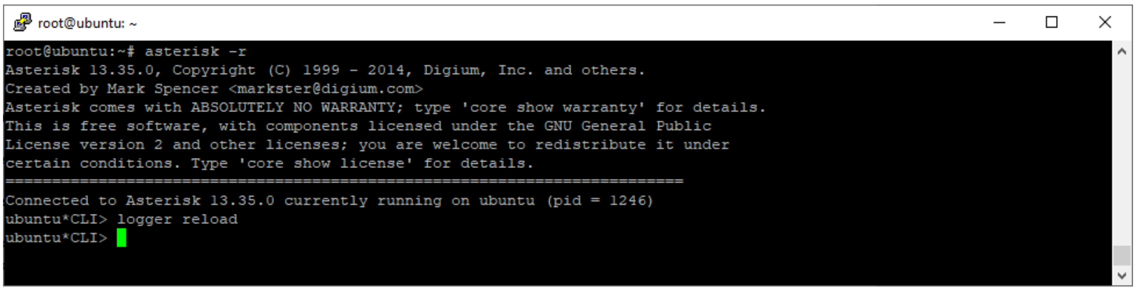


4. Enable asterisk logger

```
#asterisk -r
#logger reload
```

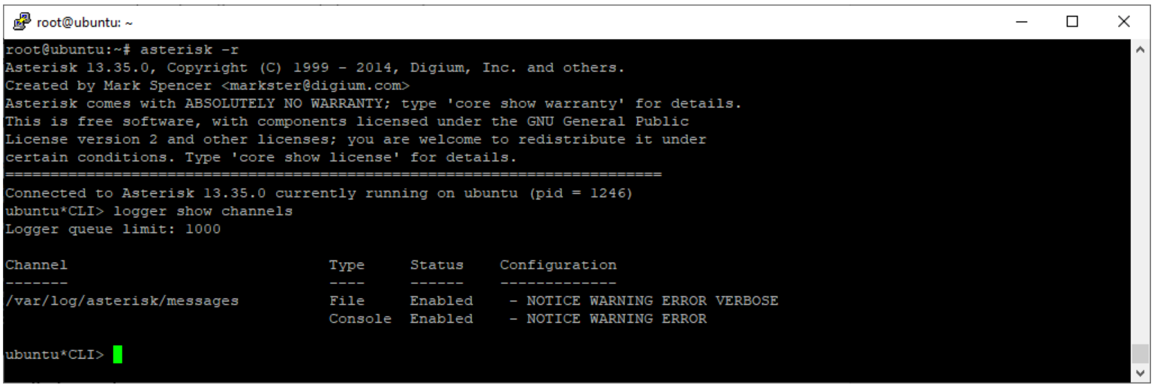
To set Permission

```
#sudo chown root /var/log/asterisk/messages
#sudo chown root /var/log/asterisk/queue_log

#sudo chmod +rwx /var/log/asterisk/messages
#sudo chmod +rwx /var/log/asterisk/queue_log
```



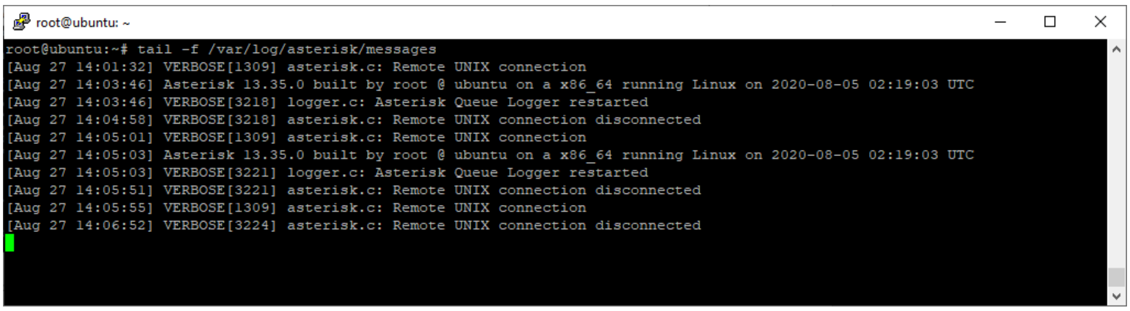

To See all channels

```
#logger show channels
```




5. Exit asterisk console and type below command

```
#tail -f /var/log/asterisk/messages
```



6. Dial and see the console

```
root@ubuntu: ~
stack
[Aug 27 14:07:34] VERBOSE[3228][C-00000000] file.c: <SIP/nirmal-00000000> Playing 'vm-theperson.gsm' (language 'en')
[Aug 27 14:07:36] VERBOSE[3228][C-00000000] file.c: <SIP/nirmal-00000000> Playing 'digits/1.gsm' (language 'en')
[Aug 27 14:07:37] VERBOSE[3228][C-00000000] file.c: <SIP/nirmal-00000000> Playing 'digits/0.gsm' (language 'en')
[Aug 27 14:07:37] VERBOSE[3228][C-00000000] file.c: <SIP/nirmal-00000000> Playing 'digits/0.gsm' (language 'en')
[Aug 27 14:07:38] VERBOSE[3228][C-00000000] file.c: <SIP/nirmal-00000000> Playing 'vm-isunavail.gsm' (language 'en')
[Aug 27 14:07:40] VERBOSE[3228][C-00000000] file.c: <SIP/nirmal-00000000> Playing 'vm-intro.gsm' (language 'en')
[Aug 27 14:07:41] VERBOSE[3228][C-00000000] pbx.c: Spawn extension (phones, 100, 4) exited non-zero on 'SIP/nirmal-00000000'
[Aug 27 14:07:48] NOTICE[1415] chan_sip.c: Received SIP subscribe for peer without mailbox: nirmal
[Aug 27 14:07:48] NOTICE[1415] chan_sip.c: Peer 'nirmal' is now Reachable. (5ms / 2000ms)
[Aug 27 14:07:48] NOTICE[1415] chan_sip.c: Received SIP subscribe for peer without mailbox: james
[Aug 27 14:07:48] NOTICE[1415] chan_sip.c: Received SIP subscribe for peer without mailbox: nirmal
[Aug 27 14:07:48] NOTICE[1415] chan_sip.c: Peer 'james' is now Reachable. (8ms / 2000ms)
[Aug 27 14:07:48] NOTICE[1415] chan_sip.c: Received SIP subscribe for peer without mailbox: james
```

7. For debugging

```
#less /var/log/asterisk/messages
```

```
root@ubuntu: ~
[Aug  4 19:42:35] Asterisk 13.35.0 built by root @ ubuntu on a x86_64 running Linux on 2020-08-05 02:19:03 UTC
[Aug  4 19:42:35] NOTICE[28924] cdr.c: CDR simple logging enabled.
[Aug  4 19:42:35] NOTICE[28924] loader.c: 231 modules will be loaded.
[Aug  4 19:42:35] NOTICE[28924] res_smdi.c: No SMDI interfaces are available to listen on, not starting SMDI listener.
[Aug  4 19:42:35] WARNING[28924] res_phoneprov.c: Unable to find a valid server address or name.
[Aug  4 19:42:35] ERROR[28924] ari/config.c: No configured users for ARI
[Aug  4 19:42:35] NOTICE[28924] chan_skinny.c: Configuring skinny from skinny.conf
[Aug  4 19:42:35] NOTICE[28924] confbridge/conf_config_parser.c: Adding default_menu menu to app_confbridge
[Aug  4 19:42:35] NOTICE[28924] cel_custom.c: No mappings found in cel_custom.conf. Not logging CEL to custom CSVs.
[Aug  4 22:14:37] Asterisk 13.35.0 built by root @ ubuntu on a x86_64 running Linux on 2020-08-05 02:19:03 UTC
[Aug  4 22:14:37] NOTICE[29353] cdr.c: CDR simple logging enabled.
[Aug  4 22:14:37] NOTICE[29353] loader.c: 231 modules will be loaded.
[Aug  4 22:14:37] NOTICE[29353] res_smdi.c: No SMDI interfaces are available to listen on, not starting SMDI listener.
[Aug  4 22:14:37] WARNING[29353] res_phoneprov.c: Unable to find a valid server address or name.
[Aug  4 22:14:37] ERROR[29353] ari/config.c: No configured users for ARI
[Aug  4 22:14:37] NOTICE[29353] chan_skinny.c: Configuring skinny from skinny.conf
[Aug  4 22:14:37] NOTICE[29353] confbridge/conf_config_parser.c: Adding default_menu menu to app_confbridge
[Aug  4 22:14:37] NOTICE[29353] cel_custom.c: No mappings found in cel_custom.conf. Not logging CEL to custom CSVs.
[Aug  4 23:10:27] Asterisk 13.35.0 built by root @ ubuntu on a x86_64 running Linux on 2020-08-05 02:19:03 UTC
[Aug  4 23:10:27] NOTICE[30430] cdr.c: CDR simple logging enabled.
[Aug  4 23:10:27] NOTICE[30430] loader.c: 231 modules will be loaded.
[Aug  4 23:10:27] NOTICE[30430] res_smdi.c: No SMDI interfaces are available to listen on, not starting SMDI listener.
[Aug  4 23:10:27] WARNING[30430] res_phoneprov.c: Unable to find a valid server address or name.
[Aug  4 23:10:27] ERROR[30430] ari/config.c: No configured users for ARI
[Aug  4 23:10:27] NOTICE[30430] chan_skinny.c: Configuring skinny from skinny.conf
[Aug  4 23:10:27] NOTICE[30430] confbridge/conf_config_parser.c: Adding default_menu menu to app_confbridge
[Aug  4 23:10:27] NOTICE[30430] cel_custom.c: No mappings found in cel_custom.conf. Not logging CEL to custom CSVs.
[Aug 27 13:45:23] Asterisk 13.35.0 built by root @ ubuntu on a x86_64 running Linux on 2020-08-05 02:19:03 UTC
[Aug 27 13:45:23] ERROR[3115] logger.c: Unable to create queue log: Permission denied
[Aug 27 13:45:23] VERBOSE[3115] logger.c: Asterisk Queue Logger restarted
[Aug 27 13:46:24] VERBOSE[3115] asterisk.c: Remote UNIX connection disconnected
[Aug 27 13:46:29] VERBOSE[1309] asterisk.c: Remote UNIX connection
[Aug 27 13:46:31] Asterisk 13.35.0 built by root @ ubuntu on a x86_64 running Linux on 2020-08-05 02:19:03 UTC
[Aug 27 13:46:31] ERROR[3118] logger.c: Unable to create queue log: Permission denied
[Aug 27 13:46:31] VERBOSE[3118] logger.c: Asterisk Queue Logger restarted
:
```

## Call Detail Records – [CDR]

The CDR system in Asterisk is used to log the history of calls in the system. In some deployments, these records are used for billing purposes. In others, call records are used for analyzing call volumes over time. They can also be used as a debugging tool by Asterisk administrators.

CDR modules are used to store Call Detail Records (CDR) in a variety of formats. Popular storage mechanisms include comma-separated value (CSV) files, as well as relational databases such as MySQL or PostgreSQL. Call detail records typically contain one record per call, and give details such as who made the call, who answered the call, the amount of time spent on the call, and so forth.

If the channel has a CDR, that CDR has its own set of variables which can be accessed just like channel variables. The following built-in variables are available.

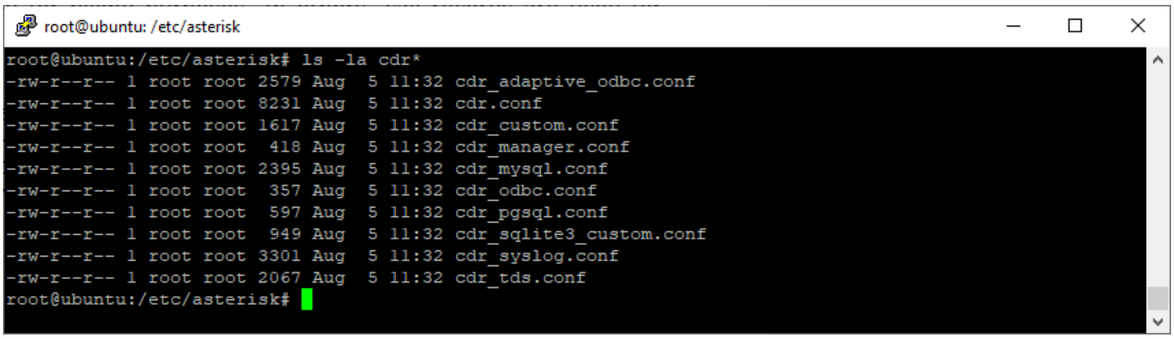
- ${CDR(clid)} Caller ID
- ${CDR(src)} Source
- ${CDR(dst)} Destination
- ${CDR(dcontext)} Destination context
- ${CDR(channel)} Channel name
- ${CDR(dstchannel)} Destination channel
- ${CDR(lastapp)} Last app executed
- ${CDR(lastdata)} Last app's arguments
- ${CDR(start)} Time the call started.
- ${CDR(answer)} Time the call was answered.
- ${CDR(end)} Time the call ended.
- ${CDR(duration)} Duration of the call.
- ${CDR(billsec)} Duration of the call once it was answered.
- ${CDR(disposition)} ANSWERED, NO ANSWER, BUSY
- ${CDR(amaflags)} DOCUMENTATION, BILL, IGNORE etc
- ${CDR(accountcode)} The channel's account code.
- ${CDR(uniqueid)} The channel's unique id.
- ${CDR(userfield)} The channels uses specified field.

In addition, you can set your own extra variables by using Set(CDR(name)=value). These variables can be output into a text-format CDR by using the cdr_custom CDR driver; see the cdr_custom.conf.sample file in the configs directory for an example of how to do this.

### Concepts Example

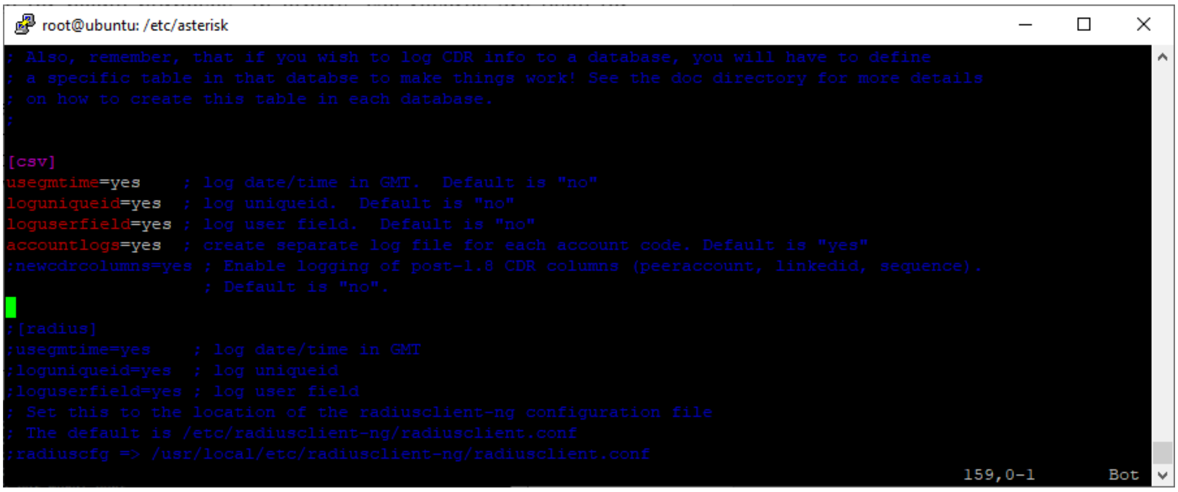
1. To see cdr related files

```
#cd /etc/asterisk
#ls -la cdr*
```



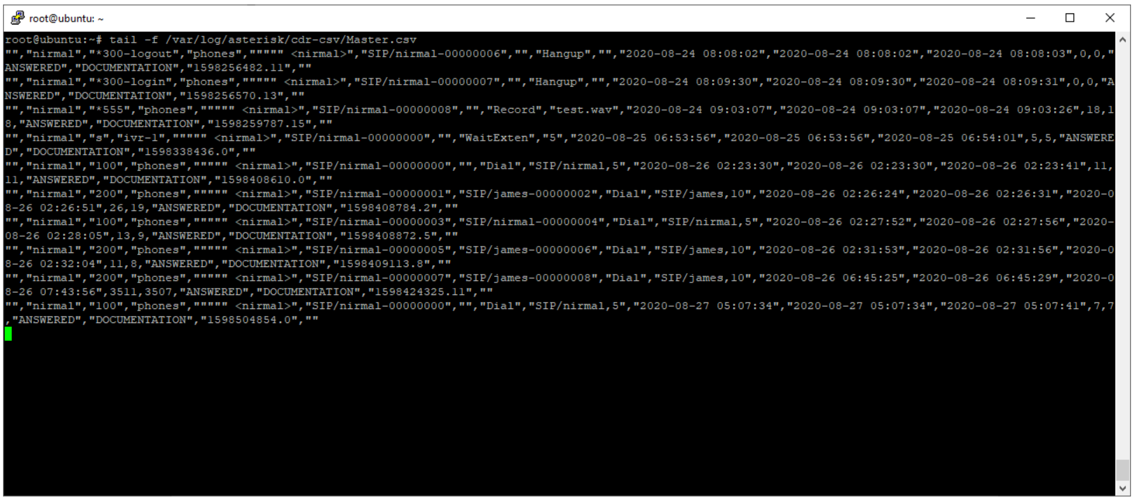
2. Open cdr.conf

    #vi /etc/asterisk/cdr.conf

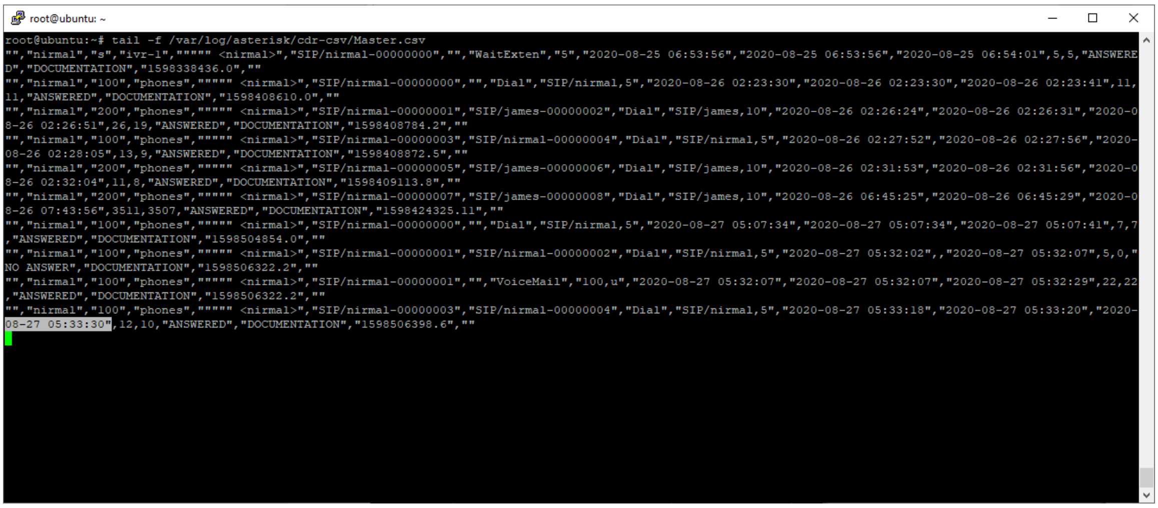



3. See the files and its record

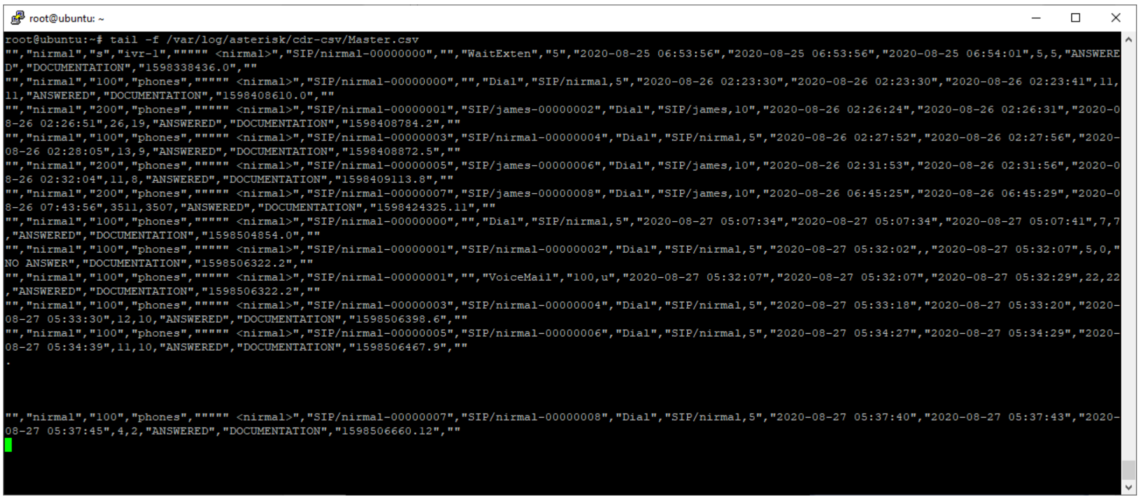
    #tail -f /var/log/asterisk/cdr-csv/Master.csv




4. Dial and disconnect the call – record will write only at end of the calls



At disconnection of calls



```
root@ubuntu:~# tail -f /var/log/asterisk/cdr-csv/Master.csv
"","nirmal","s","ivr-1","""""  <nirmal>","SIP/nirmal-00000000","","WaitExten","5","2020-08-25 06:53:56","2020-08-25 06:53:56","2020-08-25 06:54:01",5,5,"ANSWERED","DOCUMENTATION","1598338436.0",""
"","nirmal","100","phones","""""  <nirmal>","SIP/nirmal-00000000","","Dial","SIP/nirmal,5","2020-08-26 02:23:30","2020-08-26 02:23:30","2020-08-26 02:23:41",11,11,"ANSWERED","DOCUMENTATION","1598408610.0",""
"","nirmal","200","phones","""""  <nirmal>","SIP/nirmal-00000001","SIP/james-00000002","Dial","SIP/james,10","2020-08-26 02:26:24","2020-08-26 02:26:31","2020-08-26 02:26:51",26,19,"ANSWERED","DOCUMENTATION","1598408784.2",""
"","nirmal","100","phones","""""  <nirmal>","SIP/nirmal-00000003","SIP/nirmal-00000004","Dial","SIP/nirmal,5","2020-08-26 02:27:52","2020-08-26 02:27:56","2020-08-26 02:28:05",13,9,"ANSWERED","DOCUMENTATION","1598408872.5",""
"","nirmal","200","phones","""""  <nirmal>","SIP/nirmal-00000005","SIP/james-00000006","Dial","SIP/james,10","2020-08-26 02:31:53","2020-08-26 02:31:56","2020-08-26 02:32:04",11,8,"ANSWERED","DOCUMENTATION","1598409113.8",""
"","nirmal","200","phones","""""  <nirmal>","SIP/nirmal-00000007","SIP/james-00000008","Dial","SIP/james,10","2020-08-26 06:45:25","2020-08-26 06:45:29","2020-08-26 07:43:56",3511,3507,"ANSWERED","DOCUMENTATION","1598424325.11",""
"","nirmal","100","phones","""""  <nirmal>","SIP/nirmal-00000000","","Dial","SIP/nirmal,5","2020-08-27 05:07:34","2020-08-27 05:07:34","2020-08-27 05:07:41",7,7,"ANSWERED","DOCUMENTATION","1598504854.0",""
"","nirmal","100","phones","""""  <nirmal>","SIP/nirmal-00000001","SIP/nirmal-00000002","Dial","SIP/nirmal,5","2020-08-27 05:32:02",,"2020-08-27 05:32:07",5,0,"NO ANSWER","DOCUMENTATION","1598506322.2",""
"","nirmal","100","phones","""""  <nirmal>","SIP/nirmal-00000001","","VoiceMail","100,u","2020-08-27 05:32:07","2020-08-27 05:32:07","2020-08-27 05:32:29",22,22,"ANSWERED","DOCUMENTATION","1598506322.2",""
"","nirmal","100","phones","""""  <nirmal>","SIP/nirmal-00000003","SIP/nirmal-00000004","Dial","SIP/nirmal,5","2020-08-27 05:33:18","2020-08-27 05:33:20","2020-08-27 05:33:30",12,10,"ANSWERED","DOCUMENTATION","1598506398.6",""
"","nirmal","100","phones","""""  <nirmal>","SIP/nirmal-00000005","SIP/nirmal-00000006","Dial","SIP/nirmal,5","2020-08-27 05:34:27","2020-08-27 05:34:29","2020-08-27 05:34:39",11,10,"ANSWERED","DOCUMENTATION","1598506467.9",""
.


"","nirmal","100","phones","""""  <nirmal>","SIP/nirmal-00000007","SIP/nirmal-00000008","Dial","SIP/nirmal,5","2020-08-27 05:37:40","2020-08-27 05:37:43","2020-08-27 05:37:45",4,2,"ANSWERED","DOCUMENTATION","1598506660.12",""
```

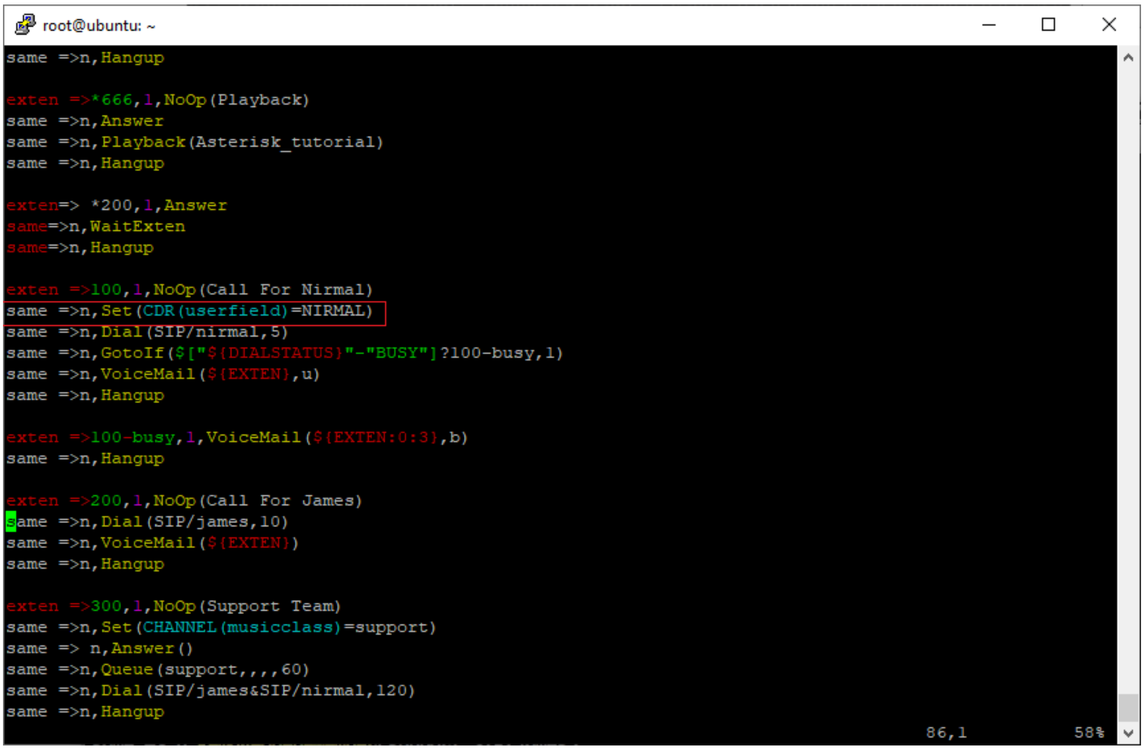
## CDR Details – Inside Record – Using CDR Variables

1. In dialplan add parameter in extensions (Here 100)

   ```
   #vi /etc/asterisk/extensions.conf
   ```

   Add

   ```
   same=>n,Set(CDR(userfield)=NIRMAL)
   ```



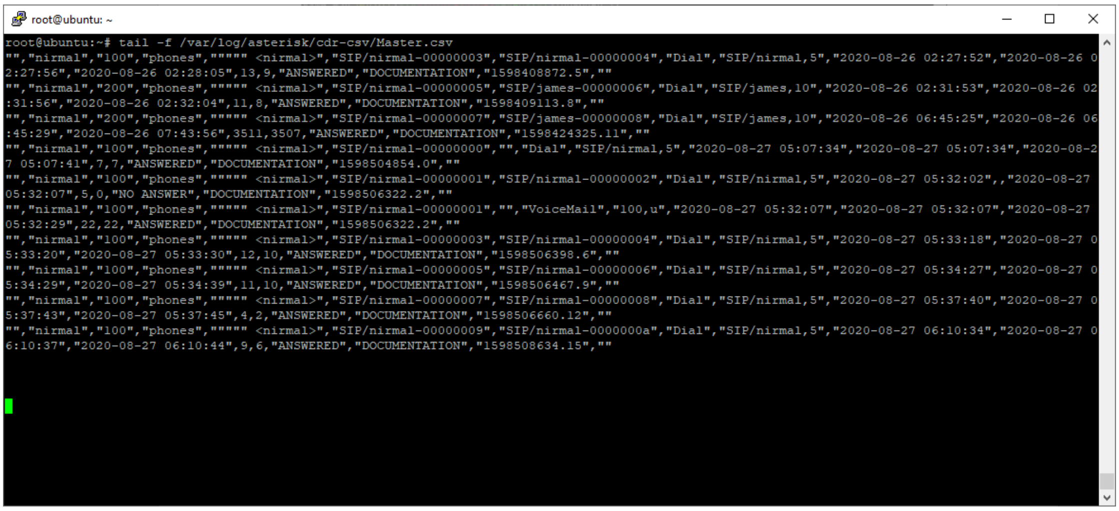
```
same =>n,Hangup

exten =>*666,1,NoOp(Playback)
same =>n,Answer
same =>n,Playback(Asterisk_tutorial)
same =>n,Hangup

exten=> *200,1,Answer
same=>n,WaitExten
same=>n,Hangup

exten =>100,1,NoOp(Call For Nirmal)
same =>n,Set(CDR(userfield)=NIRMAL)
same =>n,Dial(SIP/nirmal,5)
same =>n,GotoIf($["${DIALSTATUS}"="BUSY"]?100-busy,1)
same =>n,VoiceMail(${EXTEN},u)
same =>n,Hangup

exten =>100-busy,1,VoiceMail(${EXTEN:0:3},b)
same =>n,Hangup

exten =>200,1,NoOp(Call For James)
same =>n,Dial(SIP/james,10)
same =>n,VoiceMail(${EXTEN})
same =>n,Hangup

exten =>300,1,NoOp(Support Team)
same =>n,Set(CHANNEL(musicclass)=support)
same => n,Answer()
same =>n,Queue(support,,,,60)
same =>n,Dial(SIP/james&SIP/nirmal,120)
same =>n,Hangup
                                                            86,1          58%
```

2. Reload dialplan in asterisk console

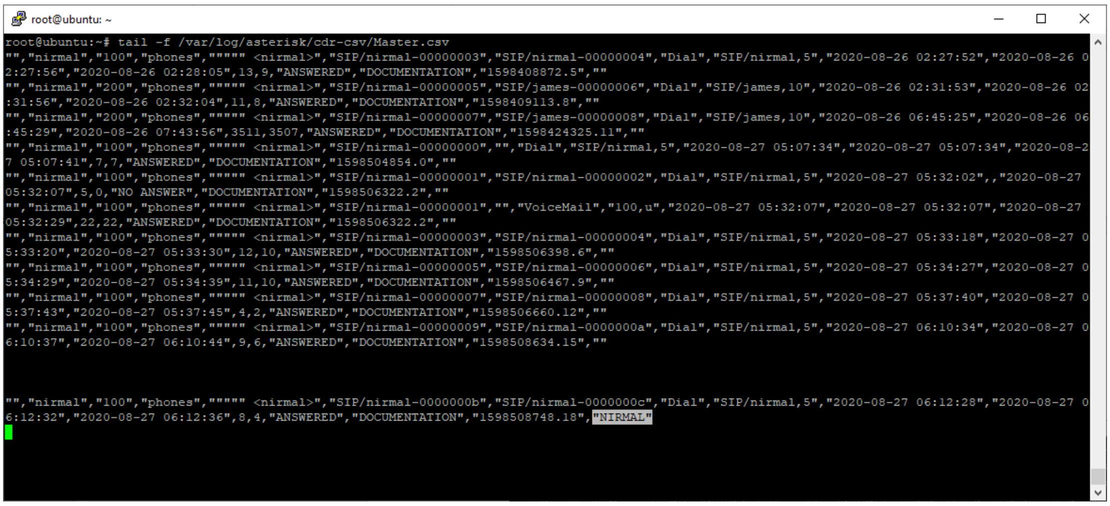
   ```
   #dialplan reload
   ```

3. Exit from asterisk console, Tail /var/log/asterisk/cdr-csv/Master.csv file

```
#tail -f /var/log/asterisk/cdr-csv/Master.csv
```




4. Dial and see the console



## Call Files

Asterisk has the ability to initiate a call from outside of the normal methods such as the dialplan, manager interface, or spooling interface.

Using the call file method, you must give Asterisk the following information:

- How to perform the call, similar to the Dial() application
- What to do when the call is answered

With call files you submit this information simply by creating a file with the required syntax and placing it in the `outgoing` spooling directory, located by default in `/var/spool/asterisk/outgoing/` (this is configurable in `asterisk.conf`).

The `pbx_spool.so` module watches the spooling directly, either using an event notification system supplied by the operating system such as **`inotify`** or **`kqueue`**, or by polling the directory each second when one of those notification systems is unavailable. When a new file appears, Asterisk initiates a new call based on the file's contents.

### Call File Syntax

The call file consists of <Key>: <value> pairs; one per line.

Comments are indicated by a '#' character that begins a line, or follows a space or tab character. To be consistent with the configuration files in Asterisk, comments can also be indicated by a semicolon. However, the multiline comments (;----;) used in Asterisk configuration files are not supported. Semicolons can be escaped by a backslash.

The following keys-value pairs are used to specify how setup a call:

- Channel: <channel> - The channel to use for the new call, in the form technology/resource as in the Dial application. This value is required.
- Callerid: <callerid> - The caller id to use.
- WaitTime: <number> - How many seconds to wait for an answer before the call fails (ring cycle). Defaults to 45 seconds.
- MaxRetries: <number> - Number of retries before failing, not including the initial attempt. Default = 0 e.g. don't retry if fails.
- RetryTime: <number> - How many seconds to wait before retry. The default is 300 (5 minutes).
- Account: <account> - The account code for the call. This value will be assigned to CDR(accountcode)

When the call answers there are two choices:

- Execute a single application, or
- Execute the dialplan at the specified context/extension/priority.

To execute an application:

- `Application: <appname>` - The application to execute
- `Data: <args>` - The application arguments

To start executing applications in the dialplan:

- `Context: <context>` - The context in the dialplan
- `Extension: <exten>` - The extension in the specified context
- `Priority: <priority>` - The priority of the specified extension; (numeric or label)
- `Setvar: <var=value>` - You may also assign values to variables that will be available to the channel, as if you had performed a Set(var=value) in the dialplan. More than one Setvar: may be specified.

  The processing of the call file ends when the call is answered and terminated; when the call was not answered in the initial attempt and subsequent retries; or if the call file can't be successfully read and parsed.

  To specify what to do with the call file at the end of processing:

- `Archive: <yes|no>` - If "no" the call file is deleted. If set to "yes" the call file is moved to the "outgoing_done" subdirectory of the Asterisk spool directory. The default is to delete the call file.

  If the call file is archived, Asterisk will append to the call file:

- `Status: <exitstatus>` - Can be "Expired", "Completed" or "Failed"

Other lines generated by Asterisk:

Asterisk keep track of how many retries the call has already attempted, appending to the call file the following key-pairs in the form:

```
StartRetry: <pid> <retrycount> (<time>)
EndRetry: <pid> <retrycount> (<time>)
```

With the main process ID (pid) of the Asterisk process, the retry number, and the attempts start and end times in time_t format.

**Directory locations**

- `<astspooldir>/outgoing` - The outgoing dir, where call files are put for processing
- `<astspooldir>/outgoing_done` - The archive dir
- `<astspooldir>` - Is specified in asterisk.conf, usually /var/spool/asterisk

**How to schedule a call**

Call files that have the time of the last modification in the future are ignored by Asterisk. This makes it possible to modify the time of a call file to the wanted time, move to the outgoing directory, and Asterisk will attempt to create the call at that time.

## Implement Call Files

### Example 1

1. Create new file

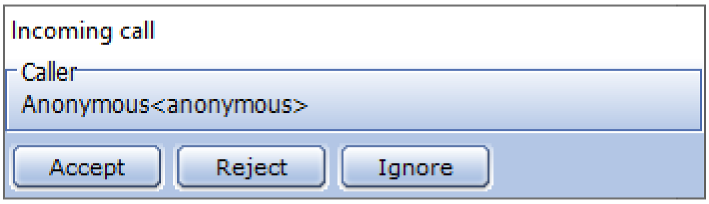
```
#vi test.call
```

Add

```
Channel:SIP/james
Application:Playback
Data:tt-monkeys
```


2. Copy to /var/spool/asterisk/outgoing/ , it will call immediately

```
#cp test.call /var/spool/asterisk/outgoing/.
```



**Example 2**

To check module installed or not in asterisk console

```
#asterisk -rvvv
#module show
```

Check module pbx_spool

1. Create new file

   ```
   #vi test2.call
   ```

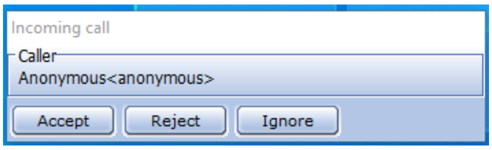
   Add

   ```
   Channel:SIP/james
   MaxRetries:2
   RetryTime:60
   WaitTime:30
   Context:phones
   Extension:100
   ```

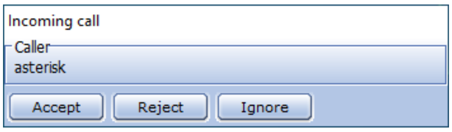
   
   

3. Copy to /var/spool/asterisk/outgoing/ , it will call immediately

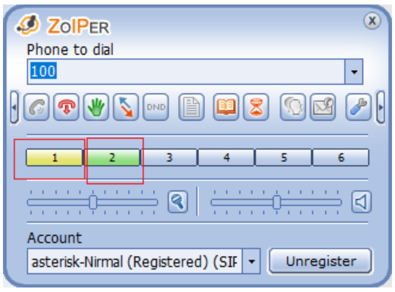
   ```
   #cp test2.call /var/spool/asterisk/outgoing/.
   ```

   ```
   Accept the call
   ```

   Two calls are active

## Asterisk Manager Interface - AMI

The **Asterisk Manager Interface** (**AMI**) protocol is a very simple protocol that allows you to communicate and manage your asterisk server, almost completely. It has support to edit/create asterisk configuration files and also manage the calls, clients, agents, dialplan, etc.

AMI is a plain text protocol, and it works by sending and receiving packets. Each packet consists in a series of text lines delimited by \r\n. And each packet is delimited by \r\n\r\n. AMI uses a **tcp** port configured in **manager.conf**. As soon as the connection is established, asterisk salutes you with something like:

```
Asterisk Call Manager/1.1
```

This message indicates that the communication can begin. Packets may be transmitted in either direction at any time after authentication.

### AMI Packets (Actions, Responses, Events)

AMI defines 3 kind of possible packets:

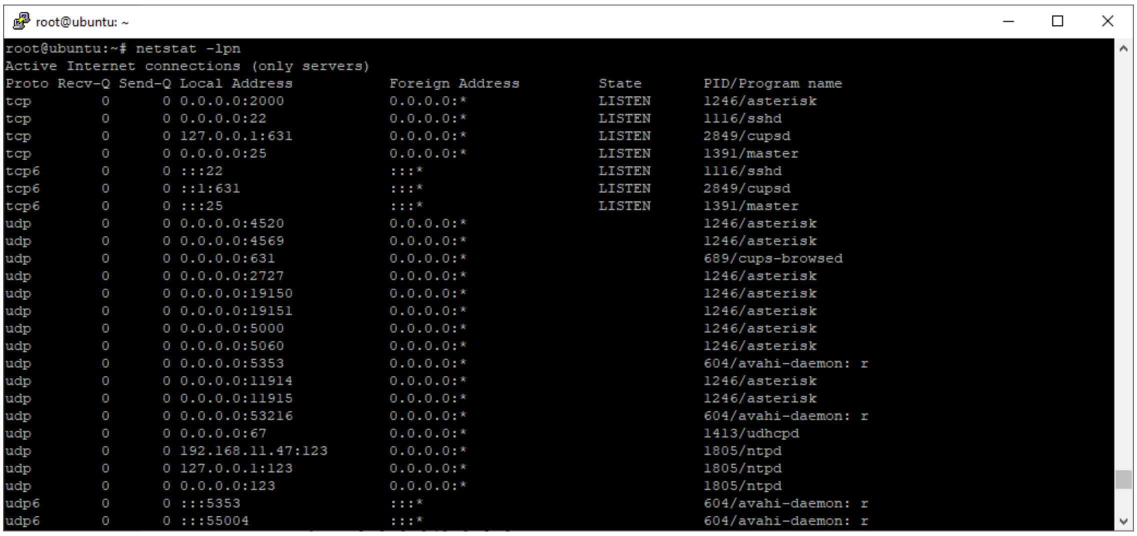
- **Actions**: This kind of packet is what the client sends. Only the client can generate Actions.
- **Responses**: Actions have at least one Response, indicating the result of the executed (or requested) action.
- **Events**: There are two kind of events. The ones attached to a particular response for a particular action, and the ones that asterisk generate to inform the connected client about things that are happening in the server (like call events, changes in variables values, agents and other clients that connect/disconnect to/from the server, etc).

The first line of a packet will have a key of "Action" when sent from the client to Asterisk, but "Event" or "Response" when sent from Asterisk to the client.

### AMI Configuration

1. To check opened port

```
#netstat -lpn
```

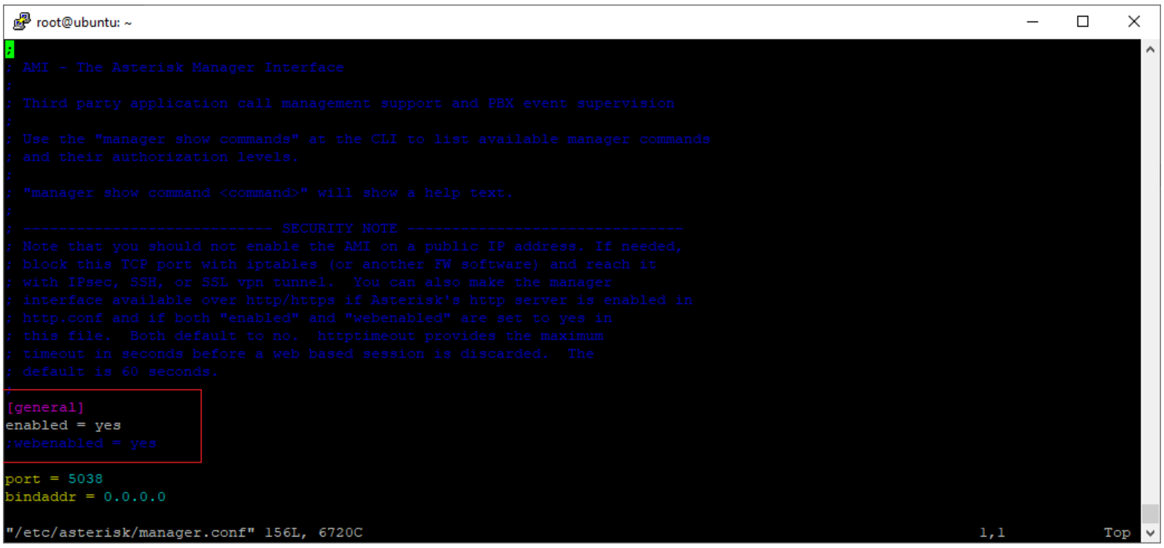


2. Enable AMI port

```
#vi /etc/asterisk/manager.conf
```

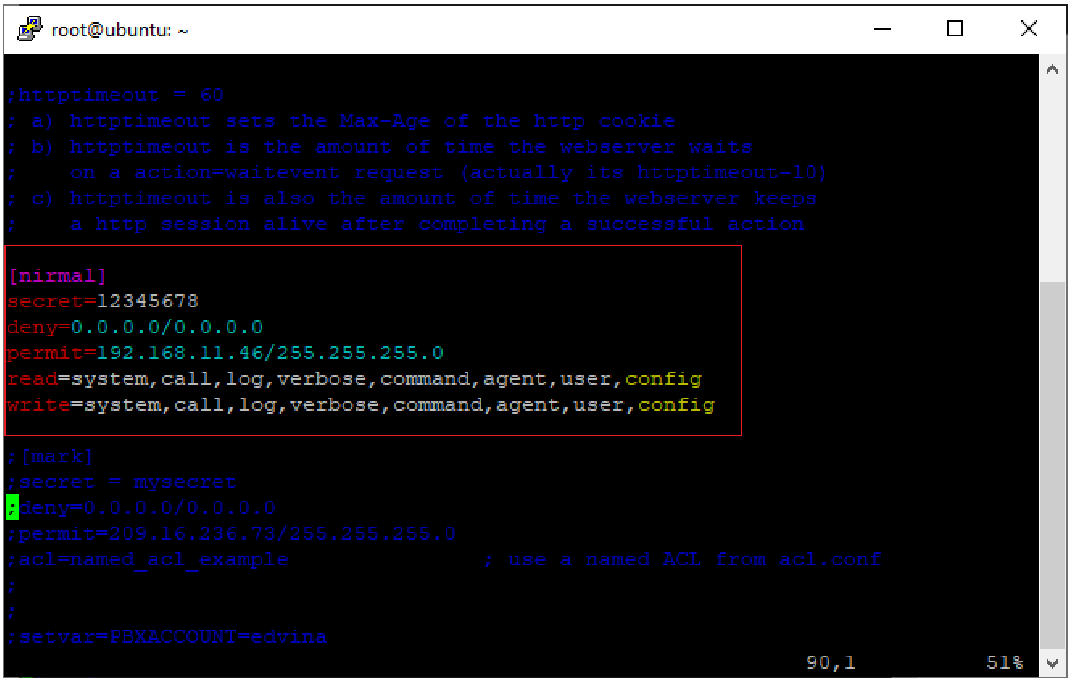
In [general] context change enabled =no to yes

```
[general]
enabled=yes
```




3. Add new context for user

```
[nirmal]
secret=12345678
deny=0.0.0.0/0.0.0.0
permit=192.168.11.46/255.255.255.0
read=all,system,call,log,verbose,command,agent,user,config
write=all,system,call,log,verbose,command,agent,user,config
```

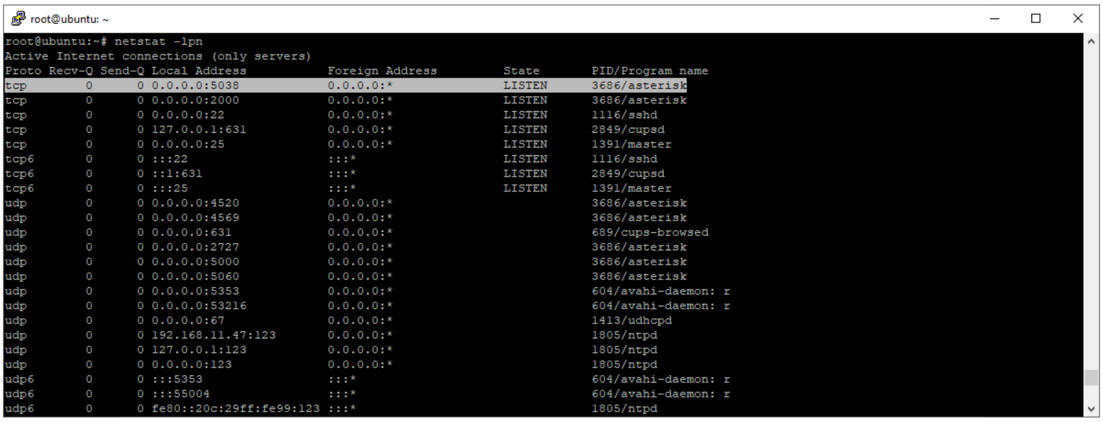


4. Restart asterisk Server

```
#service asterisk restart
```


5. Check new AMI Port

```
#netstat -lpn
```




6. To connect – double enter

#telnet 192.168.11.47 5038



7. To authenticate – after connecting (– double enter)

```
Action:login
Username:nirmal
Secret:12345678
```

root@ubuntu: ~

```
root@ubuntu:~# telnet 192.168.11.47 5038
Trying 192.168.11.47...
Connected to 192.168.11.47.
Escape character is '^]'.
Asterisk Call Manager/2.10.6
Action:login
Username:nirmal
Secret:12345678

Response: Success
Message: Authentication accepted

Event: FullyBooted
Privilege: system,all
Status: Fully Booted
```

8. Dial and see the console for the messages

root@ubuntu: ~

```
Priority: 3
Uniqueid: 1598518677.0
Linkedid: 1598518677.0
Cause: 16

Event: Hangup
Privilege: call,all
Channel: SIP/nirmal-00000000
ChannelState: 6
ChannelStateDesc: Up
CallerIDNum: nirmal
CallerIDName: <unknown>
ConnectedLineNum: <unknown>
ConnectedLineName: <unknown>
Language: en
AccountCode:
Context: phones
Exten: 100
Priority: 3
Uniqueid: 1598518677.0
Linkedid: 1598518677.0
Cause: 16
Cause-txt: Normal Clearing
```

**AMI Manager Commands**

#manager show 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
GetConfig        config,all       Retrieve configuration
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
Park             call,all         Park a channel
ParkedCalls      <none>           List parked calls
PauseMonitor     call,all         Pause monitoring of a channel
Ping             <none>           Keepalive command
PlayDTMF         call,all         Play DTMF signal on a specific channel.
QueueAdd         agent,all        Add interface to queue.
QueuePause       agent,all        Makes a queue member temporarily unavailable
QueueRemove      agent,all        Remove interface from queue.
Queues           <none>           Queues
QueueStatus      <none>           Queue Status
Redirect         call,all         Redirect (transfer) a call
SetCDRUserField  call,all         Set the CDR UserField
Setvar           call,all         Set Channel Variable
SIPpeers         system,all       List SIP peers (text format)
SIPshowpeer      system,all       Show SIP peer (text format)
Status           call,all         Lists channel status
StopMonitor      call,all         Stop monitoring a channel
UnpauseMonitor   call,all         Unpause monitoring of a channel
UpdateConfig     config,all       Update basic configuration
UserEvent        user,all         Send an arbitrary event
WaitEvent        <none>           Wait for an event to occur
```

**Originate – The Asterisk AMI Command**

Originate a call.

Generates an outgoing call to a *Extension*/*Context*/*Priority* or *Application*/*Data*

```
[Syntax]

Action: Originate
ActionID: <value>
Channel: <value>
Exten: <value>
Context: <value>
Priority: <value>
Application: <value>
Data: <value>
Timeout: <value>
CallerID: <value>
Variable: <value>
Account: <value>
EarlyMedia: <value>
Async: <value>
Codecs: <value>
```

Arguments

- `ActionID` – ActionID for this transaction. Will be returned.
- `Channel` – Channel name to call.
- `Exten` – Extension to use (requires `Context` and `Priority`)
- `Context` – Context to use (requires `Exten` and `Priority`)
- `Priority` – Priority to use (requires `Exten` and `Context`)
- `Application` – Application to execute.
- `Data` – Data to use (requires `Application`).
- `Timeout` – How long to wait for call to be answered (in ms.).
- `CallerID` – Caller ID to be set on the outgoing channel.
- `Variable` – Channel variable to set, multiple Variable: headers are allowed.
- `Account` – Account code.
- `EarlyMedia` – Set to `true` to force call bridge on early media.
- `Async` – Set to `true` for fast origination.
- `Codecs` – Comma-separated list of codecs to use for this call.

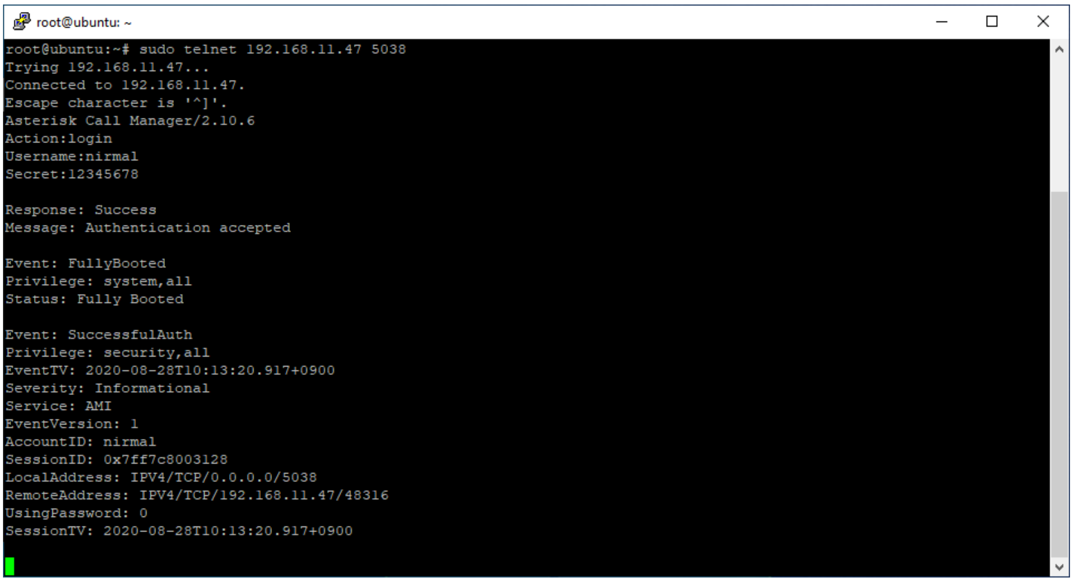
**Implement Originate**

1. Go to asterisk AMI Console

   #telnet 192.168.11.47 5038

2. In asterisk AMI control Login

```
Action:login
Username:nirmal
Secret:12345678
```




3. Initiate Command

```
Action: Originate
Channel: SIP/james
Context: phones
Exten: 100
Priority: 1
```