IP-PABX TELEPHONY SYSTEM

(ASTERISK&SER BASED OPEN SOURCE)

1922

Submitted by:

Mian Mohammad Bilal **PGDIT**
Amjad Hussain **PGDIT**

Project Coordinator:

Munawar Tiwana
Quaid-I-Azam University.

CERTIFICATE

This is to certify that

Mian Mohammad Bilal	**PGDIT**
Amjad Hussain	**PGDIT**

have submitted the final year project on the "**Ip-Pabx telephony system(asterisk @ SER based open source)** " to the department of Computer Center, **Quaid-I-Azam University**, For partial fulfillment of the degree of Bachelor's of Post Graduate Diploma Of Information Tecnaology, according to the University Policies and Regulation.

Project Supervisor

Farooq Hussain

Iphonica (PVT) LTD

Project Coordinator

Munawar Tiwana
Quaid-I-Azam University.

Dedicated to

Our respected parents and teachers

ACKNOWLEDGEMENT

We are grateful to Almighty ALLAH., who bestowed our group with the required strength, patience and skill and paved the way towards consummation of this project through mutual and sustained efforts of all the members.

Our sincere thanks to **Mr. Farooq Hussain** our external advisor, who not only envisaged this project but was always available to guide us during all of its stages - from the beginning to the end. We admire his management, intellect and communication skills which made things simpler and achievable. It is because of his continuous supervision that the project proceedings went on smoothly and ended successfully yielding a "working product".

We are thankful to **Mr. Munawar Tiwana** and the faculty members of QAU who explicitly and implicitly supported us during the course of this project. It is imperative to thanks for imparting the information related to the project requirements well in time. We appreciate his role as a college spokesperson.

Last but not the least we would like to acknowledge the love, affection and support of our family members and friends. During the course of this project, we may have ignored them when they needed us but they kept on consoling us whenever we needed them and made our ambitions a reality.

PREFACE

Each passing day opens a new horizon. New technologies emerge and infuse into our daily life. One such example is the Information Technology which is bringing revolution after revolution in the life of a common person.

The rapid growth of the Internet in the past few years has promoted many aspects of web development such as real-time interactive systems. Public Switched Telephone Networks (PSTN) is no longer the only means for transmitting voice data. Internet telephony is booming and is becoming one of the fastest moving trends. It allows users to make phone calls to others using the Internet Protocol, just as if they were using an ordinary telephone.

An advantage of Internet Telephony is that it provides a more economical way for people to have interactive communication with friends or relatives overseas. Apart from making phone calls, facsimile and voice mails, Internet Phones can also provide video calls, file transmissions, whiteboard, chat rooms and E-mails. This is far better than traditional telephone systems.

Table of Contents

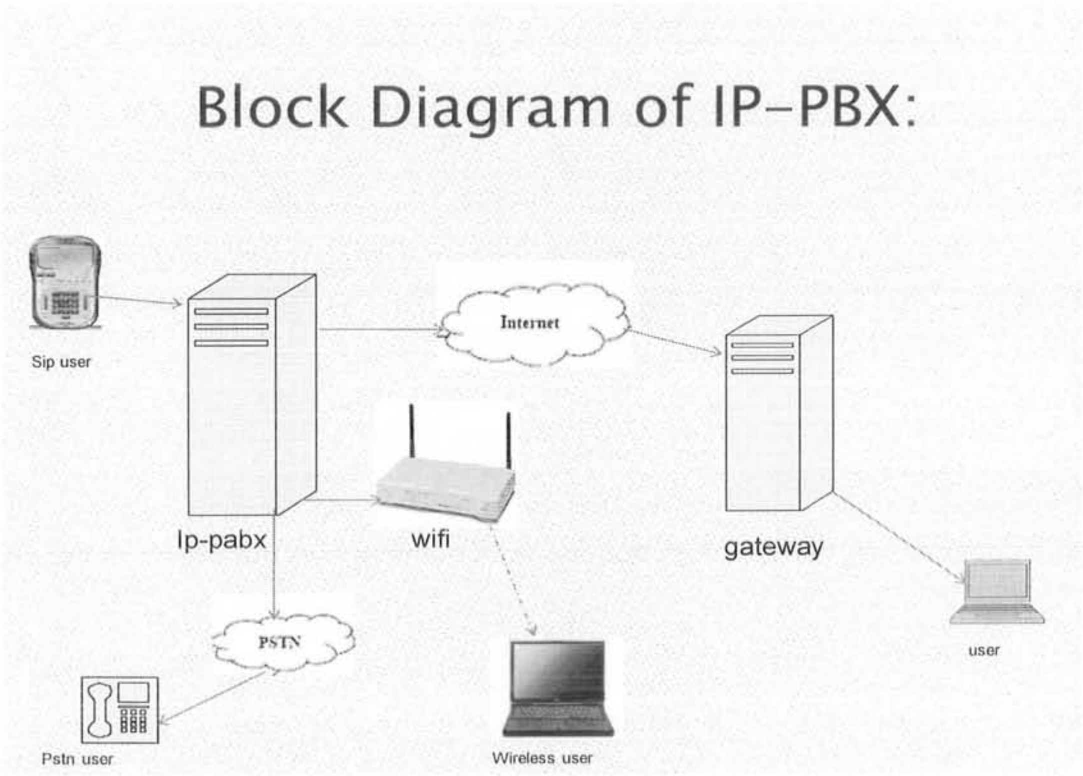
CHAPTER 1

System Introduction

IP-PBX stands for Internet Protocol-Private Branch Exchange. Generally every medium and large organization which cannot afford no of land lines (as PTCL in Pakistan case) for there offices so that they go for buying a PBX rather then buying or paying each line for there offices, the PBX will be an exchange for there organization and work fine as there own Private Exchange.

System Objectives

The objective of this Final Year Project is to implement a VoIP Asterisk gateway that enables communication between the telephone devices and the personal computers (PCs). With the system, users can communication with others across the Internet through ubiquitous access of the internet. One of the main functions of the system is to automatically receive phone calls and provide the conversion between the digital signal in computer and analog signal in ordinary telephone. .



CHAPTER 2

Literature Survey

Introduction to VOIP

Voice over IP, or Voice over Internet Protocol to use the full title, is simply a means of making telephone calls over a data network instead of over the traditional analogue public switched telephone network (PSTN).

The term VoIP describes the use of the Internet Protocol (IP) to transfer speech between two or more sites. Inherent in the term is the management of the protocol. In general, this means that the voice information is encoded into discrete digital packets and then transferred across an IP-based network.

There are many advantages to this method of telephony, primarily the cost savings that can be made by avoiding the use of the traditional PSTN. In addition to cost savings, the digital nature of VoIP allows easy administration, the implementation of additional services such as voicemail, and a reduction in the physical cabling required for new installations. However, due to the distributed nature of the internet the Quality of Service (QoS) can often suffer, and there are still a number of technical issues affecting the widespread adoption of VoIP.

Background of Internet Telephony

An Overview of Internet Telephony

Internet telephony or Voice over IP (VoIP) is a kind of technology that allows voice communication over IP data networks rather than the PSTN. The transmission of voice over IP requires conversion of an analog voice into a digital stream of data packets. The packets are then routed through the data network from one user to another, where they are converted back into voice. Figure: 2.1 depicts the process of converting analog speech signal into IP packets.

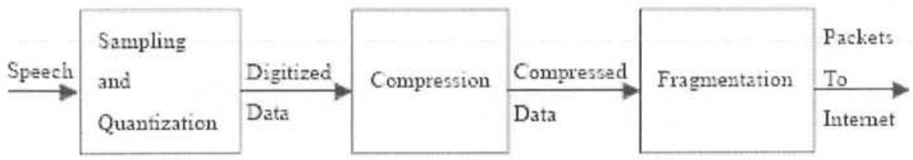


Figure: 2.1 Workflow of Internet Telephony

Evolution of IP Telephony

The first IP telephony software was produced by Vocal Tec in early 1995. By running a multimedia PC, the Vocal Tec Internet Phone lets users speak into their microphone and listen via their speakers. However, the connection was just a PC-to-PC type.

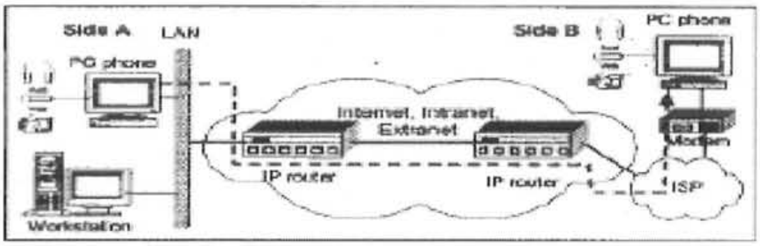


Figure: 2.2 Pc-to-Pc telephone Call

The next step in VoIP evolution was a gateway. In March of 1996, VocalTec announced it was working with an Intel Company (Dialogic Corporation) to produce the first IP telephony gateway.

Gateway Working

Conceptually, an Asterisk Telephone Gateway (shown in Figure: 2.3) works as follows. On one side, the gateway connects to the telephone world. On the other side, the gateway connects to the Internet world. The asterisk gateway takes the standard telephone signal, digitizes it, significantly compresses it, pocketsize it for the Internet using Internet Protocol (IP), and routes it to a destination.

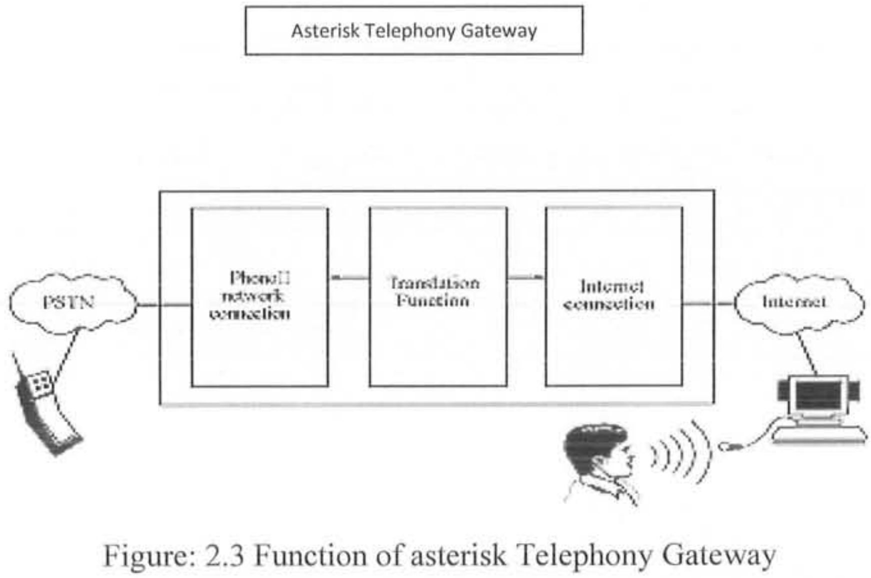


Figure: 2.3 Function of asterisk Telephony Gateway

A number of configurations can be built from this basic operation. Phone-to-PC or PC-to-phone operation (Fig.2.4) can take place with one gateway. A Phone-to-phone operation (Fig. 2.5) can occur with two gateways. In order to offer an international long distance service using gateways, for example, an organization or service provider can host one gateway in each country. The configuration costs are significantly less than a traditional circuit-switched service.

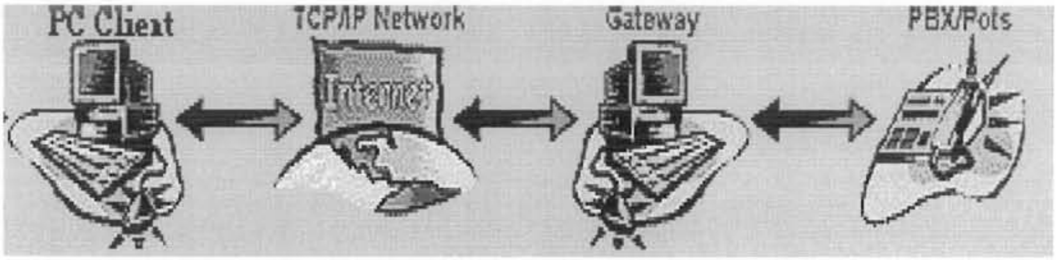


Figure: Phone-to-PC or PC-to-Phone operation

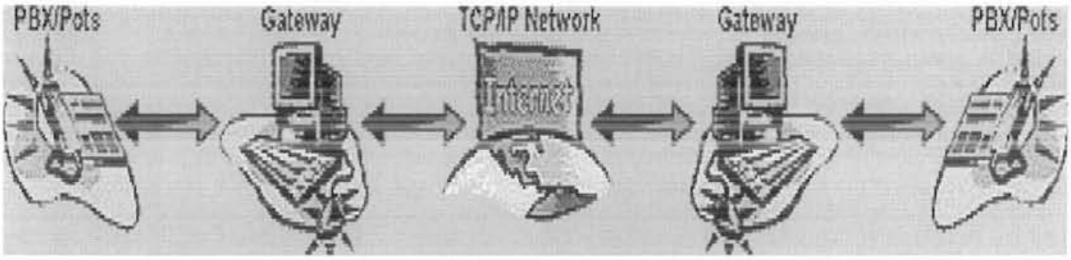


Figure: Phone-to-phone operation

WAYS OF USING VoIP

Computer-to-Computer

This is certainly the easiest way to use VoIP. You don't even have to pay for long-distance calls. There are several companies offering free or very low-cost software that you can use for this type of VoIP.

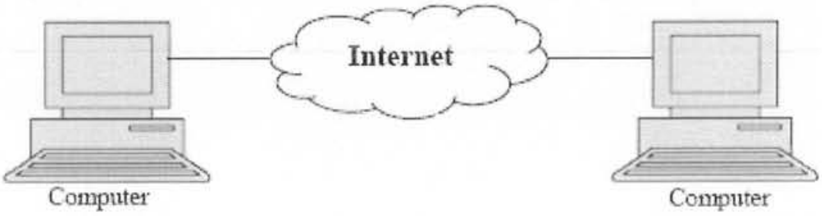


Figure: PC-to-PC

Computer-to-Telephone

This method allows you to call anyone (who has a phone) from your computer. Like computer-to-computer calling, it requires a software client. The software is typically free, but the calls may have a small per-minute charge.

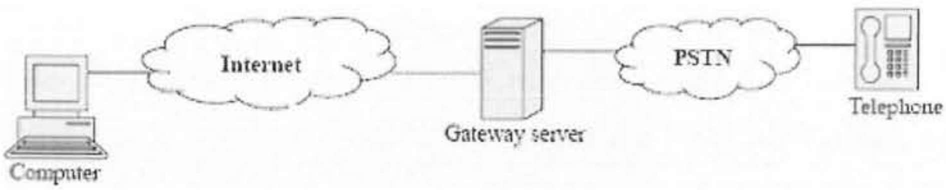


Figure: PC-to-Phone

Telephone-to-Computer

A few companies are providing special numbers or calling cards that allow a standard telephone user to initiate a call to a computer user. The caveat is that the computer user must have the vendor's software installed and running on his or her computer.

Telephone-to-Telephone

Through the use of gateways, you can connect directly with any other standard telephone in the world. To use the discounted services offered by several companies, you must call in to one of their gateways. Then, you enter the number you wish to call, and they connect you through their IP-based network. The downside is that you have to call a special number first. The upside is that the rates are typically much lower than standard long distance.

CHAPTER 3

Proposed System

So now what is the proposed system? As we have seen that analog PBX system has very constrains(major cost etc) and not much flexible, what the proposed system will provide, it replace the PBX (Private Branch Exchange, analog exchange) with the computer based exchange.

Features of Proposed System

Main Features

These are some main features which we are going to perform in our new telephony system, so that we fully entertained our users.

Audio Calls

This feature will provide a facility of a virtual room, by this they can connect to each other and can make a conference within sitting in their own room.

Video Calls

When ever a person is outside from his/her phone after some bells the call will automatically transfer to record the message and which will be then in goes into the respective users mail box as new voice mail message.

Instant Messaging

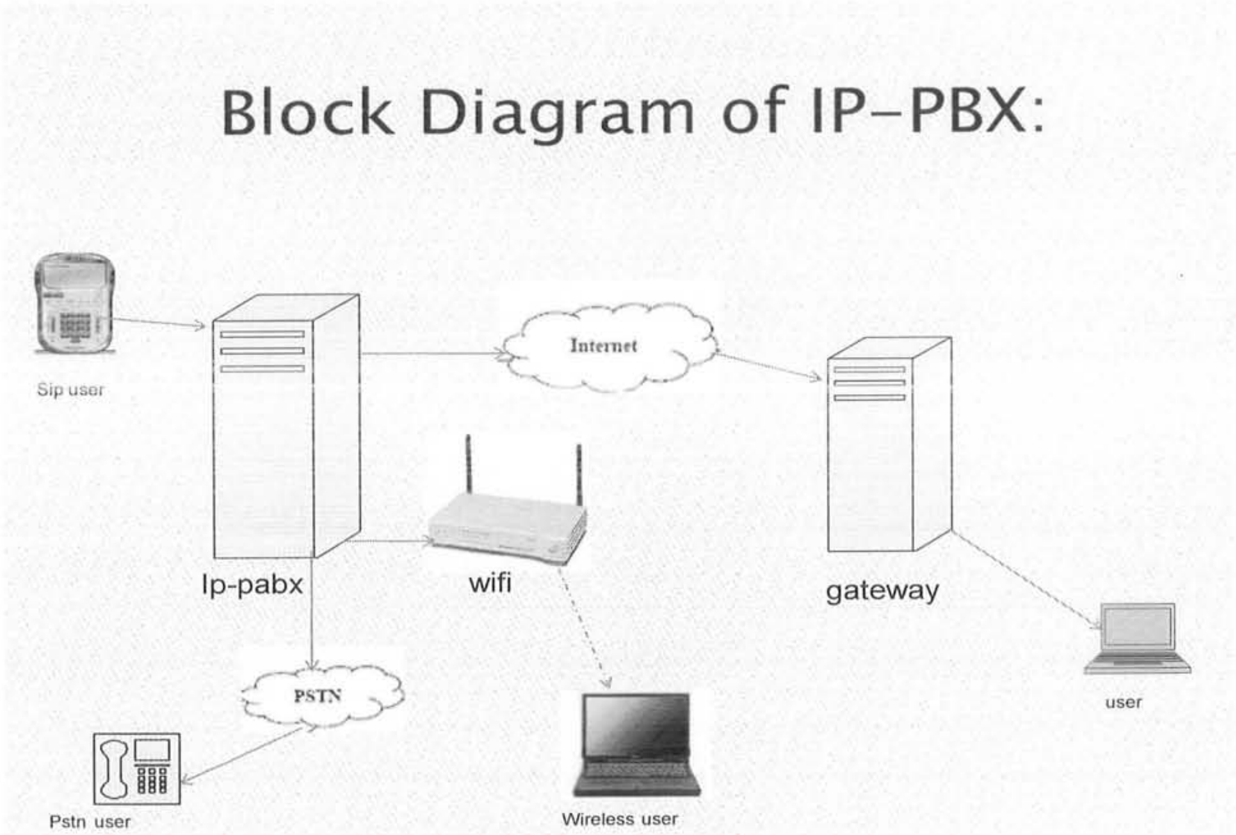
If a called party is not in office or there is a holiday then the call will be redirected to his personal mobile number

Some Calls Related Features

Here are some calls related features which will also in the proposed system, as in our life we use in the telephony system.

1- Call Detail Records
2- Call Forward on Busy
3- Call Forward on No Answer
4- Call Transfer
5- Call Waiting
6- Caller ID
7- Caller ID Blocking
8- Caller ID on Call Waiting

Block Diagram of IP – PBX



CHAPTER 4

System Analysis and Design

PBX

A PABX usually called a PBX is a Private Automatic Branch Exchange. A PABX is private because the enterprise owns it, not the telephone company. The telephone company can still be a supplier or service provider. Originally, PBX equipment was analog, more recent PBX equipment is digital. A PBX is cost attractive because it is less expensive to use a PBX than a separate phone line for every user in the enterprise and because it provides more services.

A PBX provides features and capabilities not available with direct connections to the Public Switched Telephone Network (PSTN.) A PBX moves telephone functions from the phone company to the enterprise. A PBX provides additional functions and features like

- Interactive voice response.
- Call waiting
- Conferencing
- Voice mail
- Paging
- Transferring calls

Introduction to Asterisk

Asterisk is open source Linux software for PBX. It runs on Linux and provides all of the features you would expect from a PBX.

Asterisk Architecture

Asterisk is middle ware that connects Internet and telephony technologies with Internet and telephony applications.

- VOIP
- SIP
- H.323
- IAX
- BGCP (for gateways and phone.)

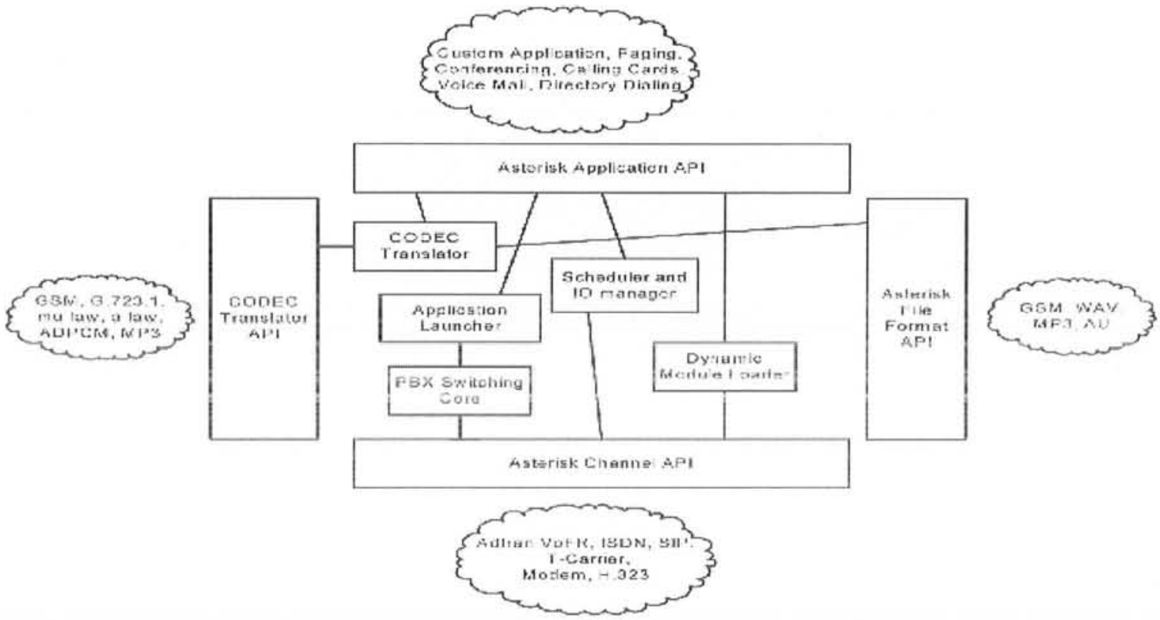
Asterisk can interoperate with almost all standards-based telephony equipment. Hardware to connect your Asterisk system is inexpensive. Asterisk supports traditional telephone technologies like ISDN PRI and T-Carrier including T1 and E-1. Telephony applications include calling, conferencing, call bridging, voicemail, auto attendant, and call parking and many others.

Dynamic Module Loader

Asterisk contains engines that perform critical functions. When Asterisk starts, the Dynamic Module Loader loads and initializes drivers. The drivers provide channel drivers, file formats, call detail recording back ends, codecs, and applications, among others.

Asterisk PBX Switching Core

The Asterisk PBX Switching Core accepts telephone calls from the interfaces. The Switching Core handles calls according to the instructions found in a dial plan. The PBX Switching Core uses the Application Launcher to ring phones, to connect to voicemail, or to dial out on outbound trunks.

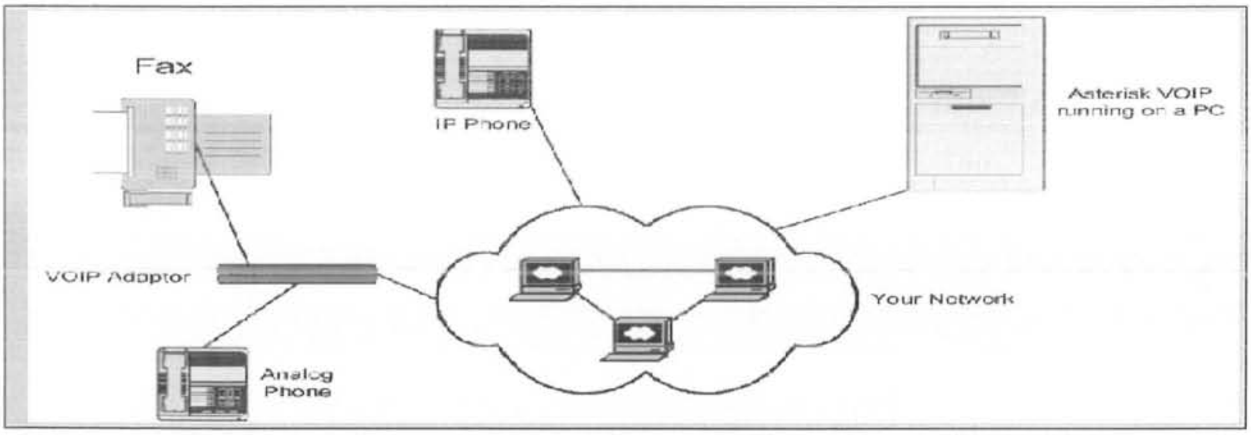


Working

VOIP (Voice over IP) systems like Asterisk can use a computer to send and receive telephone calls over a data network. Telephone calls are sent over the network as data using IP, the Internet Protocol. Telephone calls are sent from one IP phone to another IP phone as data.

Call Facility

An Asterisk system often services many IP telephones, as many as a thousand or more. Standard analog telephones or other devices like fax machines can be connected with an inexpensive adaptor. With such a system, anyone in the office can call anyone else in the office. Calling outside the office, for example anyone with a regular telephone, is described below

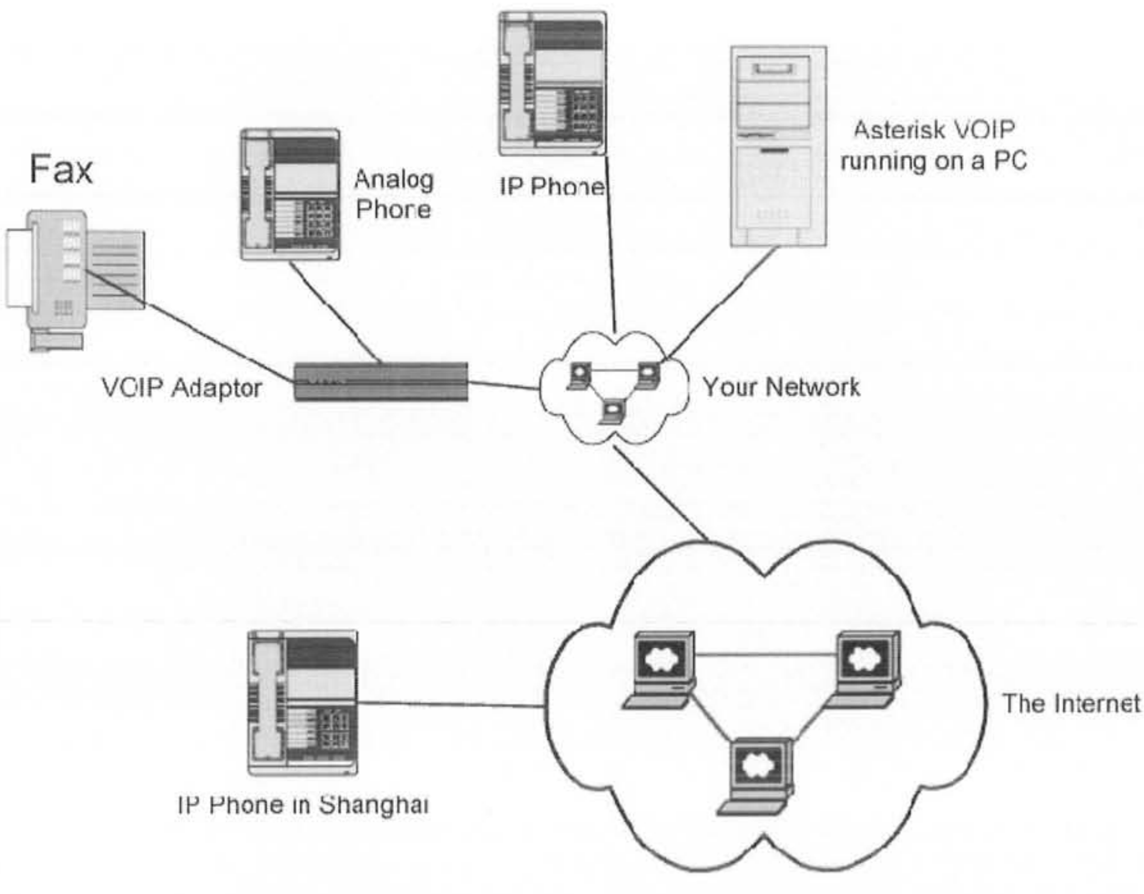


Connecting Your Asterisk Phone System to The Internet

If your Asterisk system is connected to the Internet, any VOIP enabled telephone that is connected to the Internet can be allowed to connect to your Asterisk System.

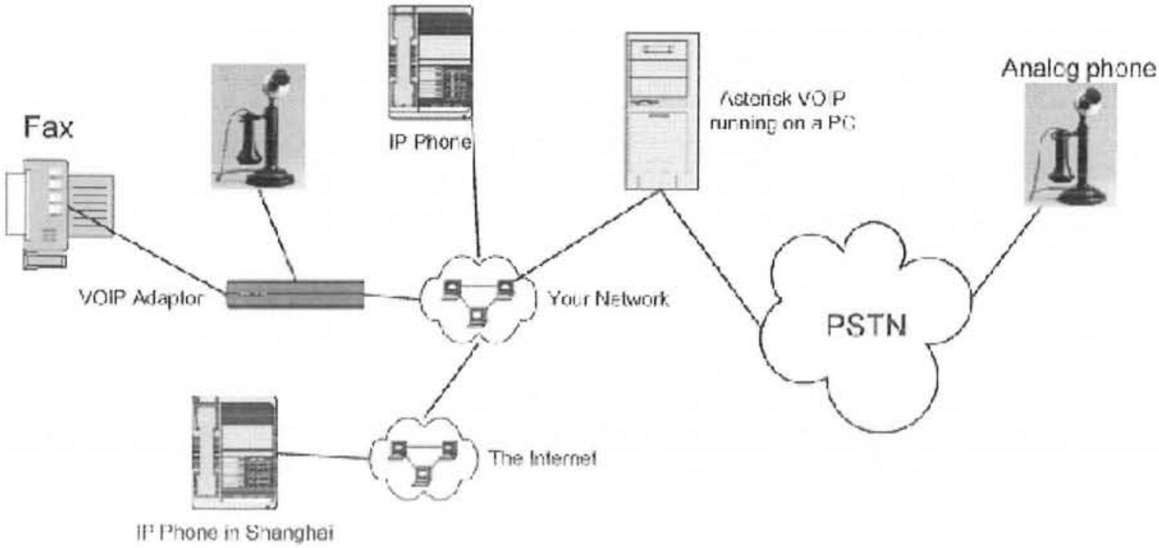
For example, you can have an Asterisk phone system in your office in New York and an office in Shanghai. Your Asterisk system in New York is connected to the Internet, and your Shanghai office is connected to the Internet. A phone in Shanghai connects to your New York

Asterisk System Over the Internet.



Connecting Your Asterisk System to PSTN

As shown in the following illustration, Asterisk users should be able to place calls to telephones connected to the PSTN. This requires a connection to the PSTN. Your Asterisk system has to be connected to the PSTN. This is easy to do. Asterisk users need a telephone number if calls are to be accepted from the PSTN. You have to rent telephone numbers from a telephone company..



System Requirements

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

Packages Need

Asterisk uses three main packages:

Asterisk program (asterisk)

Zapata telephony drivers (zaptel)

PRI libraries (libpri)

Other packages

asterisk-sounds (for Mysql)

mpg123(for Mp3)

Package Requirements

1- To compile Asterisk, you must install the GCC compiler (Version 3.x or later) and its dependencies.

2- Asterisk also requires bison, a parser generator program that replaces yacc, and ncurses for CLI functionality.

3- The cryptographic library in Asterisk requires OpenSSL and its development packages.

4- Zaptel requires libnewt and its development packages for the zttool program and the usb-uhci module for ztdummy. , Zaptel also requires the libpri package (again, even if you aren't using PRI circuits, we recommend that you install libpri along with zaptel).

The following sections discuss how to obtain, extract, compile, and install the asterisk, zaptel, libpri, and asterisk-sounds packages.

Obtaining the Source Code

The Asterisk source code can be obtained either through FTP or CVS.The Asterisk source code can be obtained from the Digium FTP server, located at ftp://ftp.digium.com. The easiest way to obtain the stable release is through the use of the program wget.

Note that we will be making use of the /usr/src/ directory to extract and compile the Asterisk source.

```
# cd /usr/src/
# wget --passive-ftp ftp.digium.com/pub/asterisk/asterisk-1.*.tar.gz
# wget --passive-ftp ftp.digium.com/pub/asterisk/asterisk-sounds-   *.tar.gz
# wget --passive-ftp ftp.digium.com/pub/zaptel/zaptel-*.tar.gz
# wget --passive-ftp ftp.digium.com/pub/libpri/libpri-*.tar.gz
```

Extracting the Source Code

With the help of following commands you can extract it.

```
# cd /usr/src/
# tar zxvf zaptel-*.tar.gz
# tar zxvf libpri-*.tar.gz
# tar zxvf asterisk-*.tar.gz
# tar zxvf asterisk-sounds*.tar.gz
```

Compiling packages

Compiling Zaptel

Compiling the Zapata telephony drivers for use with your Digium hardware is straightforward simply run make for either the 2.4 or 2.6 Linux kernels (the Makefile will determine the kernel version for you). Use these commands to compile Zaptel (replace version with your version of zaptel):

```
# cd /usr/src/zaptel-version
# make clean
# make
# make install
```

Compiling libpri

Libpri is used by various makers of Time Division Multiplexing (TDM) hardware, but even if you don't have the hardware installed it is safe to compile and install this library. You must compile and install libpri before Asterisk, as it will be detected and used when Asterisk is compiled. Here are the commands (replace version with your version of libpri):

```
# cd /usr/src/libpri-version
# make clean
# make
# make install
```

Compiling Asterisk

Once you've compiled and installed the zaptel and libpri packages (if you need them), you can move on to Asterisk. This section walks you through a standard installation and introduces some of the alternative make arguments that you may find useful. We'll also look at how you can edit the Makefile to optimize the compilation of Asterisk.

Standard Installation

Asterisk is compiled with gcc through the use of the GNU make program. Unlike many other programs, there is no need to run a configuration script for Asterisk. To get started compiling Asterisk, simply run the following commands (replace version with your version of Asterisk):

```
# cd /usr/src/asterisk-version
# make clean
# make
# make install
# make samples
```

Run the make samples command to install the default configuration files. Installing these files (instead of configuring each file manually) will allow you to get your Asterisk system up and running much faster.

Installing Additional Prompts

The asterisk-sounds package contains many useful professionally recorded prompts. It is highly recommended that you install it now, as we will be using some of the prompts from this package in later chapters. To do so, run the following commands:

```
# cd /usr/src/asterisk-sounds
# make install
```

Other Useful Add-ons

The asterisk-addons package contains code to allow the storage of Call Detail Records (CDRs) to a MySQL database and to natively play MP3s.

Common Compiling Issues

There are many common compiling issues that users often run into. Here are some of the more common problems, and how to resolve them.

Asterisk

First, let's take a look at some of the errors you may encounter when compiling Asterisk.

C compiler cannot create executables

If you receive the following error while attempting to compile Asterisk, you must install the gcc compiler and its dependencies:

Checking whether the C compiler (gcc) works... no configure:

Error: installation or configuration problem:

C compiler cannot create executables.make:
*** [editline/libedit.a] Error 1

The following packages are required for gcc:

gcc

glibc-kernheaders

cpp

binutils

glibc-headers

glibc-devel

These can be installed manually, by copying the files off of your distribution disks, or through the yum package manager, with the command yum install gcc.

bison: command not found

The following error may be encountered if the bison parser, which is required for parsing expressions in the extensions.conf file, is not found:

```
bison ast_expr.y -name-prefix=ast_yy -o ast_expr.c
make: bison: Command not found
Make: *** [ast_expr.c] Error 127
```

The following files are required in order to install Asterisk; they can be installed with the yum install bison command:

Bison

m4.

Zaptel

You may also run into errors when compiling Zaptel. Here are some of the most commonly occurring problems, and what to do about them.

make: cc: Command not found

You will receive the following error if you attempt to build Zaptel without the gcc compiler installed:

```
make: cc: Command not found
make: *** [gendigits.o] Error 127
```

Be sure to install gcc and its dependencies.

FATAL: Module wctdm/fxs/fxo not found

The TDM400P cards require the PCI bus to be Version 2.2. If you attempt to load the Zapata telephony drivers with an older version, you may get the following errors:

When attempting to load the wctdm driver, you may see this error:

```
FATAL: Module wctdm not found
```

When attempting to load the wctdm or wcfxo driver, you may see an error such as this:

```
ZT_CHANCONFIG failed on channel 1: No such device or address (6)
FATAL: Module wctdm not found
```

resolve these errors is to use a newer motherboard that supports PCI

Version 2.2 .

Loading Asterisk

Asterisk can be loaded in a variety of ways. The easiest way is to start Asterisk by running the binary file directly from the Linux command-line interface.

CLI Commands

The Asterisk binary is, by default, located at /usr/sbin/asterisk. If you run /usr/sbin/asterisk, it will be loaded as a daemon. There are also a few switches you should be aware of that allow you to (re)connect to the Asterisk CLI, set the verbosity of CLI output, and allow core dumps if Asterisk crashes. To explore the full range of options, run Asterisk with the -h switch.

```
# /usr/sbin/asterisk -h
```

Here is a list of the most commonly used options:

-c (Console. This allows you to connect to the Asterisk CLI.)

-v (Verbosity. This is used to set the amount of output for CLI debugging.)

-g (Core dump. If Asterisk were to crash unexpectedly, this would cause a core file to be created for later tracing with gdb).

-r (Remote. This is used to reconnect remotely to an already running Asterisk process. (The process is remote from the standpoint of the console connecting to it but is actually a local process on the machine. This has nothing to do with connecting to a remote process over a network using a protocol such as IP, as this is not supported).

Directories Used by Asterisk

Asterisk uses several directories on a Linux system to manage the various aspects of the system, such as voicemail recordings, voice prompts, and configuration files. This section discusses the necessary directories, all of which are created during installation and configured in the asterisk.conf file.

/etc/asterisk/ (contains the Asterisk configuration files)

/usr/lib/asterisk/modules/ (Asterisk loadable modules)

/var/lib/asterisk (astdb file and a number of subdirectories)

The subdirectories within /var/lib/asterisk/ include:

agi-bin/ (contains your custom scripts)
firmware/
images/
keys/ (public/private key system to authenticate peers)
mohmp3/ (how the MP3 files are formatted as in MOH)
sounds/ (available voice prompts for Asterisk).

/var/spool/asterisk/ (subdirectories, including outgoing/, qcall/, tmp/, and voicemail/).

/var/run/ (contains the process ID (pid) information for all active processes)

/var/log/asterisk/ (contains Asterisk logs information)
/var/log/asterisk/cdr-csv (store the CDRs in comma-separated value (CSV) format).

Working with Interface Configuration Files

In this chapter, we're finally going to "get our hands dirty" and start building an Asterisk configuration. For the first few sections on FXO and FXS channels, we'll assume that you have the Digium Dev-Lite kit with one FXO and one FXS interface, which allows you to connect to an analog phone line (FXO) and to an analog phone (FXS).

The configuration we do in this chapter won't be particularly useful on its own, but it will be a kernel to build on. We're going to touch on the following files.

zaptel.conf

Here, we'll do low-level configuration for the hardware interface. We'll set up one FXO channel and one FXS channel.

zapata.conf

In this file, we'll configure Asterisk's interface to the hardware.

Extensions.conf

The dialplans we create will be extremely primitive, but they will prove that the system is working.

Sip.conf

This is where we'll configure the SIP user.

Iax.conf

This is where we'll configure incoming and outgoing IAX channels.

Meetme.conf

With the help this file audio conference rooms can be made.

Voicemail.conf

Voice mailboxes are here.

In the following sections, you will be editing several configuration files . You'll have to reload these files for your changes to take effect.

FXO and FXS Channels

The difference between an FXO channel and an FXS channel is simply which end of the connection provides the dial tone.

FXO (Foreign Exchange Office)

An FXO port does not generate a dial tone; it accepts one. A common example is the dial tone provided by your phone company. If your Asterisk server has a compatible FXO port, you can plug a telephone line from your Telephone Company ("PTCL") into this port. Asterisk can then use the PTCL line to place and receive telephone calls.

FXS (Foreign Exchange Station)

An FXS port provides both the dial tone and ringing voltage to alert the station user of an inbound call. if your Asterisk server has a compatible FXS port, you may plug an analog telephone into your Asterisk server, so that Asterisk may call the phone and you may place calls. An FXS port will be defined with FXO signaling.

Hardware Require

X101p card is used for as a single port FXO card to connect your IP-PBX to the CO (Central office). The mostly used is the Digium Hardware which is reliable.

Dialplan Configuration for Incoming Call

The following minimal dialplan makes use of the Echo( ) application to verify that bidirectional communications for the channel are working:

```
[incoming]

; incoming calls from the FXO port are directed to this context

from zapata.conf

exten => s,1,Answer( )

exten => s,2,Echo( )
```

Whatever you say, the Echo( ) application will relay back to you.

Dialing in

Now that the FXO channel is configured, let's test it. Run the zttool application and connect your PSTN line to the FXO port on your X101P. Once you have a phone line connected to your FXO port, you can watch the card come out of a RED alarm.

Now dial the PSTN number from another external phone (such as a cell phone). Asterisk will answer the call and execute the Echo( ) application. If you can hear your voice being reflected back, you have successfully installed and configured your FXO channel.

SIP and RTP

SIP is an application-layer signaling protocol that uses the well-known port 5060 for communications. SIP can be transported with either the UDP or TCP transport-layer protocols. Asterisk does not currently have a TCP implementation for transporting SIP messages, but it is possible that future versions may support it (and patches to the code base are gladly accepted). SIP is used to "establish, modify, and terminate multimedia sessions such as Internet telephony calls." SIP does not transport media between endpoints.

SIP was not the first, and is not the only, VoIP protocol in use today (others include H.323, MGCP, IAX, and so on), but currently it seems to have the most momentum with hardware vendors. The advantages of the SIP protocol lie in its wide acceptance and architectural flexibility (and, we used to say, simplicity!).

SIP Configuration

Here is a basic sip.conf file:

```
[general]
context=default
srvlookup=yes

[john]
type=friend
secret=welcome
qualify=yes          ; Qualify peer is no more than 2000 ms away
nat=no               ; this phone is not natted
host=dynamic         ; This device registers with us
canreinvite=no       ; Asterisk by default tries to redirect
context=internal     ; the internal context controls what we can do
```

The sip.conf file starts with a [general] section, which contains the channel settings and default options for all users and peers defined within sip.conf. You can override the default settings on a per-user/peer basis by configuring them within the user/peer definition.

Domain Name System Service records (DNS SRV records) are a way of setting up a logical, resolvable address where you can be reached. By using SRV records, you gain many of the advantages of DNS, whereas disabling them breaks the SIP RFC and removes the ability to place SIP calls based on domain names. (Note that if multiple records are returned, Asterisk will use only the first.) DNS SRV record lookups are disabled by default in Asterisk, but it's highly recommended that you turn them on. To enable them, set srvlookup=yes in the [general] section of sip.conf.

Client Configuration

While it would be impossible to show all the possible configurations for all the end devices that can communicate with Asterisk, we feel it beneficial to provide the configuration for at least one free soft phone, which you can use in determining if Asterisk is right for your organization. We've chosen to use X-ten's X-Lite client , which you can download from their web site (http://www.xten.com).

Go to SIP account Setting => Add

Dialplan Configuration for Internal Calls

Many SIP phones, both soft and hard, are multi-line phones. This means they can accept multiple incoming calls at the same time. Thus, to test your X-Lite soft phone you can simply call yourself, and the call will loop back from the Asterisk server and onto line two of the client. To call yourself, dial extension 100. If your preferred client doesn't support multi-line functionality, you can use extension 611 to enter the Echo( ) test application.

```
[internal]
exten => 100,1,Dial(SIP/john)
```

exten => 611,1,Echo( )

IVR (Interactive Voice Response)

Background ( )

Plays a file while accepting touch-tone (DTMF) digits

The most common use of the Background( ) application is to create voice menus (often called auto-attendants or phone trees). Many companies use voice menus to direct callers to the proper extensions, thus relieving their receptionists from having to answer every single call.

Background(filename1[&filename2...][,options[,language]])

Plays the specified audio file(s) while waiting for the user to begin entering an extension. Once the user begins to enter an extension, the playback is terminated. The filename should be specified without a file extension, as Asterisk will automatically find the file format with the lowest translation cost.

Valid options include one of the following:

skip

Causes the playback of the message to be skipped if the channel is not in the "up" state (i.e., hasn't yet been answered). If skip is specified, the application will return immediately should the channel not be off-hook.

no answer

Does not answer the channel before playing the specified file. Without this option, the channel will automatically be answered before the sound is played. Not all channels support playing messages before being answered.

The language argument may be used to specify a language to use for playing the prompt, if it differs from the current language of the channel.

Returns -1 if the channel was hung up, or if the given filename does not exist; otherwise, returns 0.

```
exten => 123,1, Answer( )
exten => 123,2,Background('exter-ext-of-person');
```

Tool

Audocity is the tool used for recoding the sound files for IVR .These sound files are sampled at 8 kHz sample/second and these files are in .WAV format .these files r located in /var/lib/asterisk/sound directory.

Interaction With IVR

An administrator can change the ivr as per their need from front-end .

Voice Mail Configuration

Voicemail

One of the most popular (or, actually, unpopular) features of any modern telephone system is voicemail. Naturally, Asterisk has a very flexible voicemail system. Some of the features of Asterisk's voicemail system include:

Unlimited password-protected voicemail boxes, each containing mailbox folders for organizing voicemail

Different greetings for busy and unavailable states

Default and custom greetings

The ability to associate phones with more than one mailbox and mailboxes with more than one phone

Email notification of voicemail, with the voicemail optionally attached as a sound file.

No, you really don't have to pay for this; and yes, it really does work.

Voicemail forwarding and broadcasts

Message-waiting indicator (flashing light or stuttered dial tone) on many types of phones

Company directory of employees, based on voicemail boxes

And that's just the tip of the iceberg! In this section, we'll introduce you to the fundamentals of a typical voicemail setup.

The voicemail configuration is defined in the configuration file called voicemail.conf . This file contains an assortment of settings that you can use to customize the voicemail system to your needs. Covering all the available options in voicemail.conf would be beyond the scope of this chapter, but the sample configuration file is well documented and quite easy to follow.

For now, look near the bottom of the file, where voicemail contexts and voicemail boxes are defined.

Creating Mailboxes

Inside each voicemail context, we define different mailboxes. The syntax for defining a mailbox is:

```
mailbox => password,name[,email[,pager_email[,options]]]
```

Let's explain what each part of the mailbox definition does:

mailbox

> This is the mailbox number. It usually corresponds with the extension number of the associated set.

password

> This is the numeric password the mailbox owner will use to access her voicemail. If the user changes her password, the system will update this field in the voicemail.conf file.

name

> This is the name of the mailbox owner. The company directory uses the text in this field to allow callers to spell usernames.

email

> This is the email address of the mailbox owner. Asterisk can send voicemail notifications (including the voicemail message itself) to the specified email box.

pager_email

> This is the email address of the mailbox owner's pager or cell phone. Asterisk can send a short voicemail notification message to the specified email address.

options

> This field is a list of options that sets the mailbox owner's time zone and overrides the global voicemail settings. There are nine valid options: attach, serveremail, tz, saycid, review, operator, callback, dialout, and exitcontext. These options should be in option=value pairs, separated by the pipe character (|). The tz option sets the user's time zone to a time zone previously defined in the [zonemessages] section of voicemail.conf, and the other eight options override the global voicemail settings with the same names.

A typical mailbox definition might look something like this:

```
101=>1234,JoePublic,jpublic@somedomain.com,jpublic@pagergateway.net,
tz=central|attach=yes
```

Continuing with our dialplan from the last chapter, let's set up voicemail boxes for John and Jane. We'll give John a password of 1234 and Jane a password of 4444 (remember, these go in voicemail.conf, not extensions.conf):

```
[default]
101 => 1234,John Doe,john@asteriskdocs.org,jdoe@pagergateway.tld
102 => 4444,Jane Doe,jane@asteriskdocs.org,jane@pagergateway.tld
```

Adding Voicemail to the Dialplan

Now that we've created mailboxes for Jane and John, let's allow callers to leave messages for them if they don't answer the phone. To do this, we'll use the VoiceMail( ) application.

The VoiceMail( ) application sends the caller to the specified mailbox, so that he can leave a message. The mailbox should be specified as mailbox@context, where context is the name of the voicemail context. The mailbox number can also be prefixed by the letter b or the letter u. If the letter b is used, the caller will hear the mailbox owner's busy message. If the letter u is used, the caller will hear the mailbox owner's unavailable message (if one exists).

Let's use this in our sample dialplan. Previously, we had a line like this in our [internal] context, which allowed us to call John:

```
exten => 101,1,Dial(${JOHN},,r)
```

Now, let's change it so that if John is busy (on another call), it'll send us to his voicemail, where we'll hear his busy message (don't forget that the Dial( ) application sends the caller to priority n+101 if the dialed line is busy):

```
exten => 101,1,Dial(${JOHN},,r)
exten => 101,102,VoiceMail(b101@default)
```

Next, let's add an unavailable message that the caller will be played if John doesn't answer the phone within 10 seconds. Remember, the second argument to the Dial( ) application is a timeout. If the call is not answered before the timeout expires, the call is sent to the next priority. Let's add a 10-second timeout, and a priority to send the caller to voicemail if John doesn't answer in time:

```
exten => 101,1,Dial(${JOHN},10,r)
exten => 101,2,VoiceMail(u101@default)
```

```
exten => 101,102,VoiceMail(b101@default)
```

If we add these two new priorities and a timeout argument to the Dial( ) application, callers will get John's voicemail (with the appropriate greeting) if John is either busy or unavailable. A slight problem remains, however, in that John has no way of retrieving his messages. Let's remedy that.

Accessing Voicemail

Users can retrieve their voicemail messages, change their voicemail options, and record their voicemail greetings by using the VoiceMailMain( ) application. In its typical form, VoiceMailMain( ) is called without any arguments. Let's add extension 500 to the [internal] context of our dialplan so that internal users can dial it to access their voicemail messages:

```
exten => 500,1,VoiceMailMain( )
```

Music On Hold Configuration (MOH)

MusicOnHold(class)

Plays hold music specified by class, as configured in musiconhold.conf. If omitted, the default music class for the channel will be used. You can use the SetMusicOnHold( ) application to set the default music class for the channel.

Returns -1 on hangup; otherwise, does not return.

```
; transfer telemarketers to this extension to keep them busy
exten => 123,1,Answer( )
exten => 123,2,Playback(tt-allbusy)
exten => 123,3,MusicOnHold(default)
```

/etc/asterisk/musiconhold.conf

```
[classes]

; on Debian boxes files are in /usr/share/asterisk/mohmp3

; on other boxes, files are in /var/lib/asterisk/mohmp3

default => quietmp3:/usr/share/asterisk/mohmp3

loud => mp3:/usr/share/asterisk/mohmp3

podcasts => mp3:/usr/share/asterisk/mohmp3/podcasts
```

Call Transfer

Dial(tech/username:password@hostname/extension,ring-timeout,flags)

> Allows you to connect together all of the various channel types. Any valid channel type (such as SIP, IAX2, H.323, MGCP, Local, or Zap) is acceptable to Dial( ), but the parameters that need to be passed to each channel will depend on the information the channel type needs to do its job. For example, a SIP channel will need a network address and user to connect to, whereas a Zap channel is going to want some sort of phone number.

t

> Permits the called party to transfer a call by pressing the # key. Please note that if this option is used, reinvites are disabled, as Asterisk needs to monitor the call to detect when the called party presses the # key.

T

> Permits the caller to transfer a connected call by pressing the # key. Again, note that if this option is used, reinvites are disabled, as Asterisk needs to monitor the call to detect when the caller presses the # key.

EXAMPLE:

exten = > 786, 1, Dial (SIP/PTD, 20, t)

It means that PTD is having privilege to transfer the call

Asterisk CDR with MySQL Configuration
Asterisk can store CDR records in a MySQL database, as an alternative to CSV text files and other database formats.

Way to Download cdr_mysql
Due to Mysql client libraries licensing, the Mysql billing application is no longer an integrated part of the Asterisk standard distribution. It is now located in the asterisk-addons CVS directory.

Compile
Run "make clean", then "make", then "make install" in the addons directory you created. If make fails due to complaining about a missing "asterisk.h" file you can either copy this file from your asterisk directory, or create a soft link ("ln -s ...") for /usr/src/asterisk that points to your asterisk source directory.

Asterisk Config cdr_mysql.conf

Configuration of database for billing with CDR records

This is a sample configuration that assumes the SQL server is local.

```
;
; Note - if the database server is hosted on the same machine as the
; asterisk server, you can achieve a local Unix socket connection by
; setting hostname=localhost
;
; port and sock are both optional parameters. If hostname is specified
; and is not "localhost", then cdr_mysql will attempt to connect to the
; port specified or use the default port. If hostname is not specified
; or if hostname is "localhost", then cdr_mysql will attempt to connect
; to the socket file specified by sock or otherwise use the default socket
; file.
;
[global]
 hostname=localhost
 dbname=asteriskcdrdb
 password=password
 user=asteriskcdruser
 ;port=3306
 ;sock=/tmp/mysql.sock
 ;userfield=1
```

Configurator and Call Monitor

As before, the precise explanation of the Asterisk which is very important to understand the Asterisk world is given.

Now in this section, the main part that how all these things are maintain, how the asterisk server configure, monitor and maintain billing (CDR) through our own custom Configurator plus Monitor.

General

So how we are maintaining those entire things? We are using the Asterisk Real Time which is still developing, basically with the help of real time all the data in the conf files will be then maintain with the use of tables using mysql database.

Asterisk Real-Time Architecture

With the help asterisk real time architecture, the data can be switch to the data base tables, with the help of simple switch statement in extconfig.conf file. The detail procedure is given below for each.

Configuring for real time place in res_mysql.conf

[general]

dbhost = localhost

dbname = asterisk

dbuser = root

dbpass = shahid

dbport = 3306

dbsock = /var/lib/mysql/mysql.sock

All Asterisk RealTime Configuration

In this section we have how to switch extensions.conf to the database.

Extconfig.conf

Add the following line, swapping your own personal values if you wish

extensions => mysql,asterisk,extensions_table

You can change mysql to odbc if you want to use odbc.
You can change asterisk to be the name of your database.
You can change extensions_table to be the name of the extensions table we will create below.

Extensions.conf

[test]

switch => Realtime/[context](optional)@[family](optional)[/options]

This tells Asterisk that any call into the 'test' context are to be switched to RealTime using the context "mycontext" and the family name "realtime_ext".

Custom Configurations

Extconfig.conf

[settings]

sipusers=> mysql,asterisk,sip_buddies

sippeers=> mysql,asterisk,sip_buddies

extensions => mysql,asterisk,extensions_table

voicemail => mysql,asterisk,voicemail_user

Sip.conf

[general]

[global]

[EMPLOYEE]

switch=>Realtime/@sippeers

switch=>Realtime/@sipusers

Extensions.conf

[general]

[globals]

[EMPLOYEE]

switch=>Realtime/EMPLOYEE@extensions

switch=>Realtime/INCOMMING_INTERNAL@extensions

switch=>Realtime/LOCAL@extensions

switch=>Realtime/NATIONAL@extensions

switch=>Realtime/MOBILE@extensions

e.g LOCAL context in the data base allow a user to place a local call and have entries something like this.

exten => _XXXXXXX,1,Dial(Zap/1/${EXTEN})

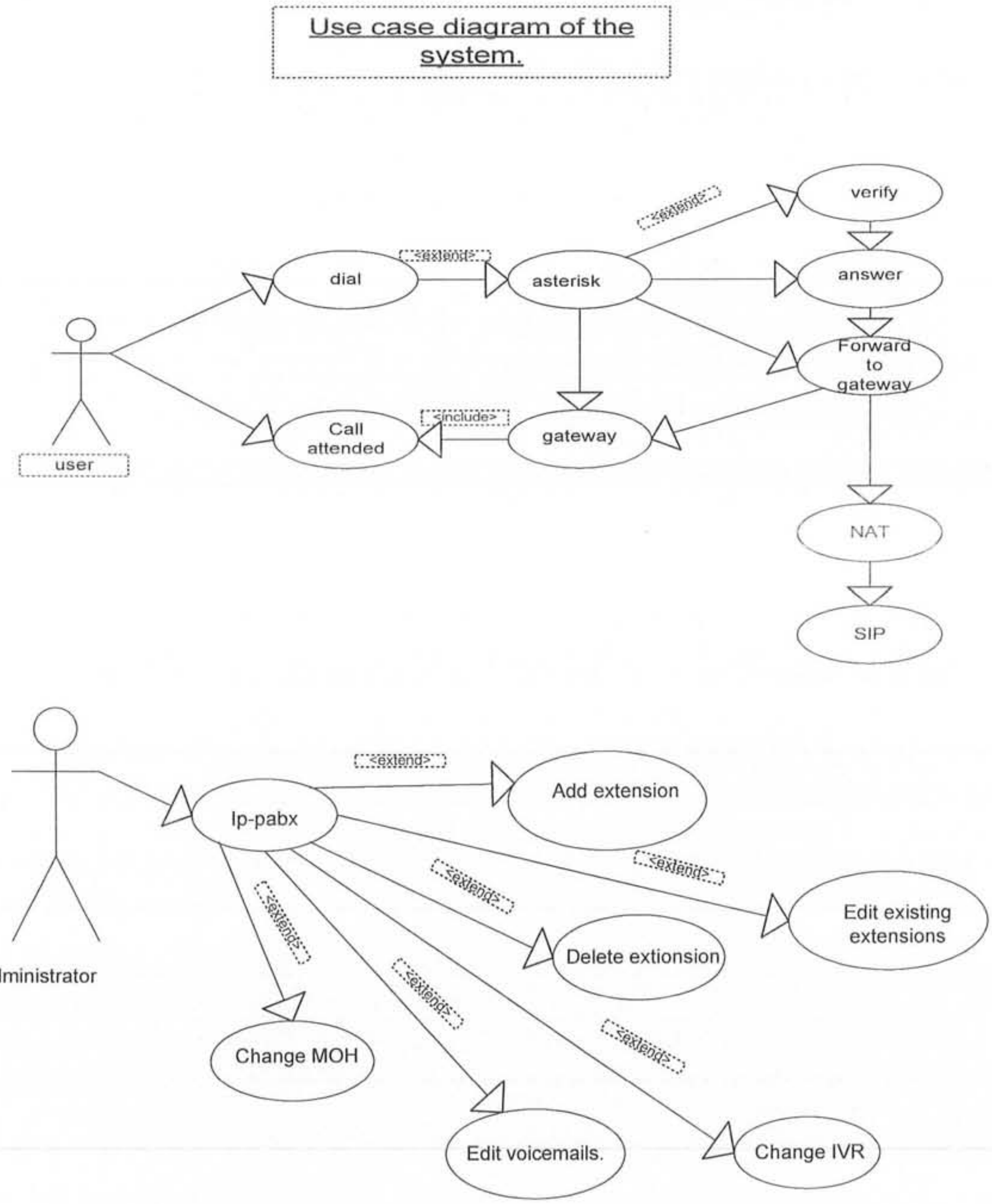
_XXXXXXX _ shows this is pattern and X is from 0-9

Zap/1 mean using channel 1

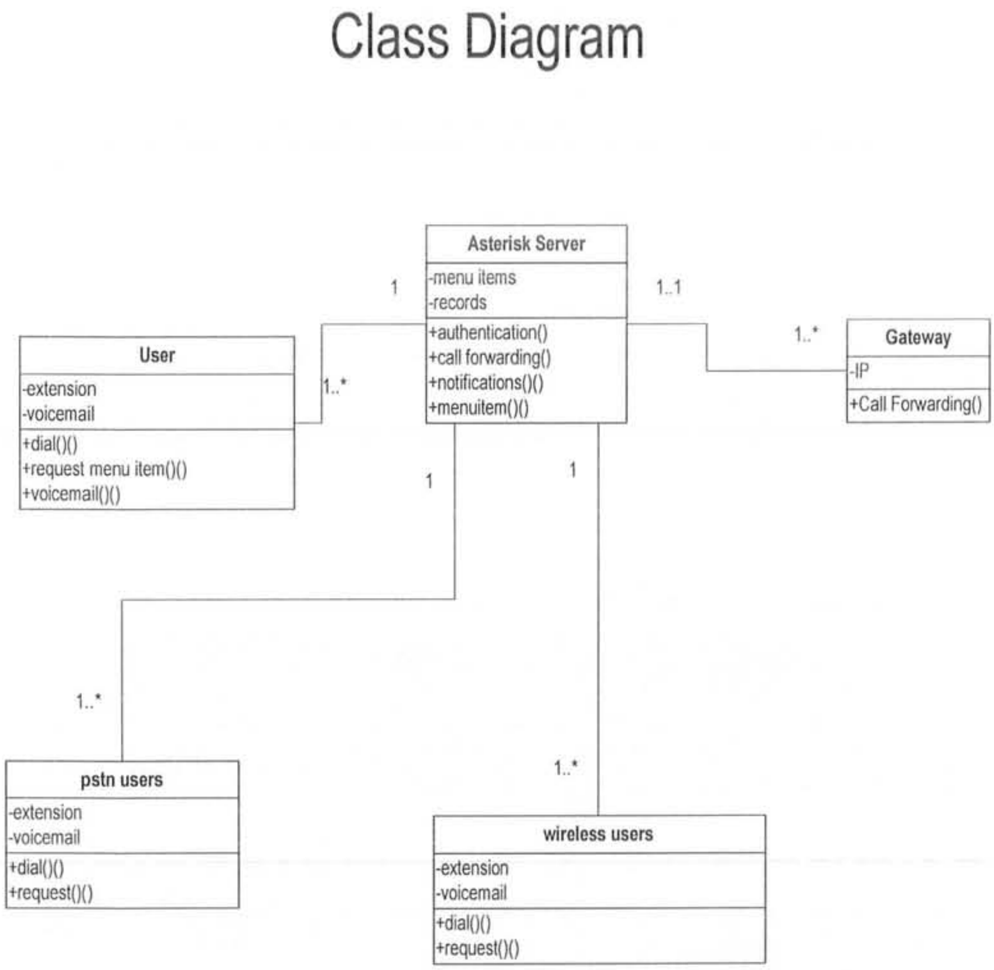
${EXTEN} variable sets when user enter any local number

UML DIAGRAMS

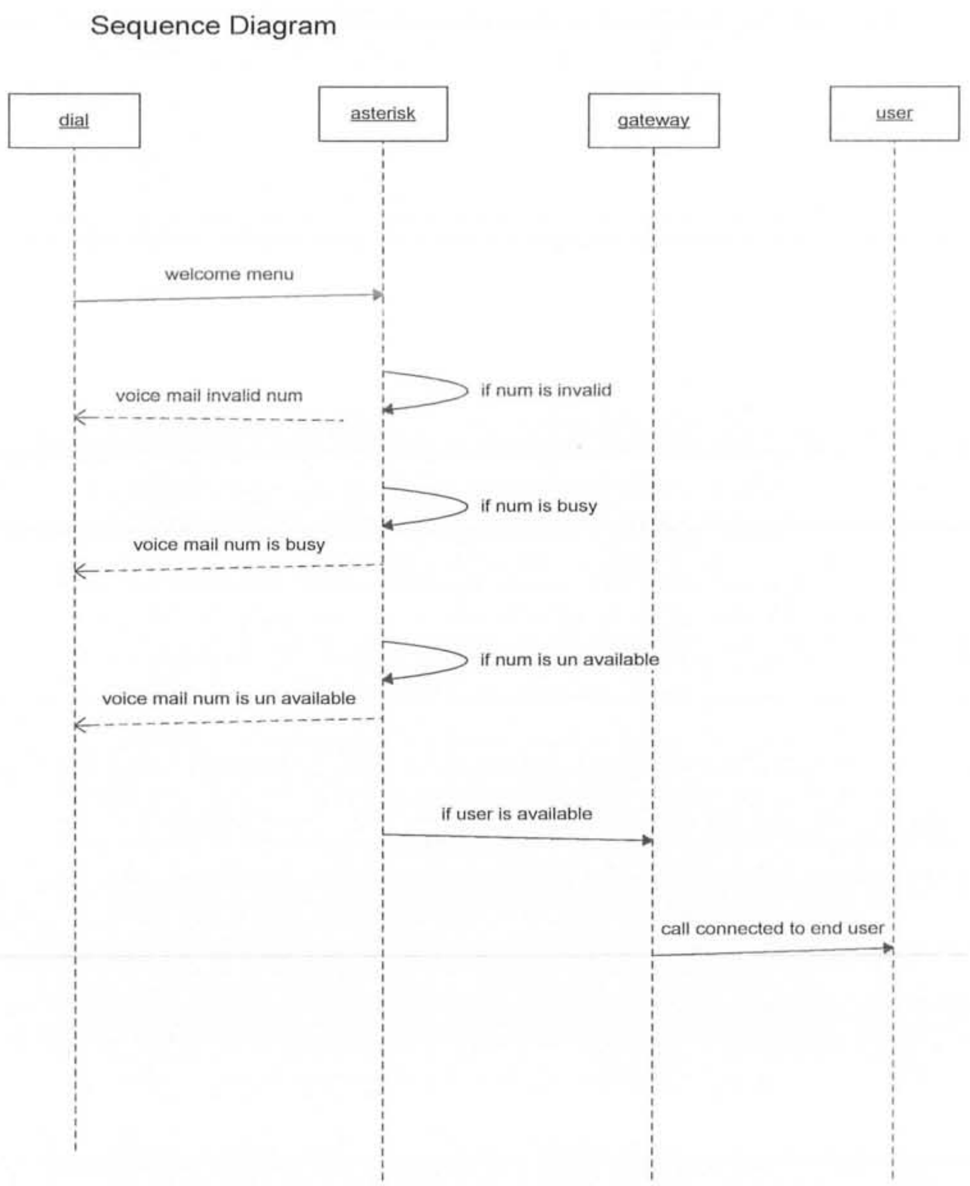
Use Case Diagram



Class Diagram



Sequence Diagram



CHAPTER 5

System Development

Configuring IPphones Soft & IP Deskphone

Configuring IP Phones for Use with Asterisk

Asterisk allows people to communicate using the internet. It is one of the cheapest ways to talk. The PC clients connect to each other through an Asterisk server which uses a Linux/Unix OS.

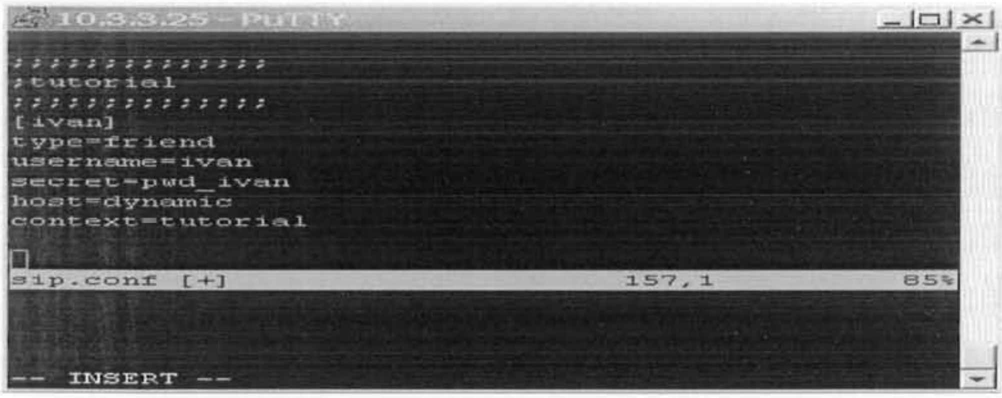
To Register SIP Users:

All users must be registered in sip.conf and all valid extensions must be declared in extensions.conf

Follow these steps for:

-Registering a user in sip.conf
-Registering a number in extensions.conf

1. Go to the /etc/asterisk directory on your Asterisk server. Open sip.conf and register the following test user.



Note that ALL characters after ';' (semi-colomn) are interpreted by Asterisk as a comment.

[ivan] This means we are registering user 'ivan'

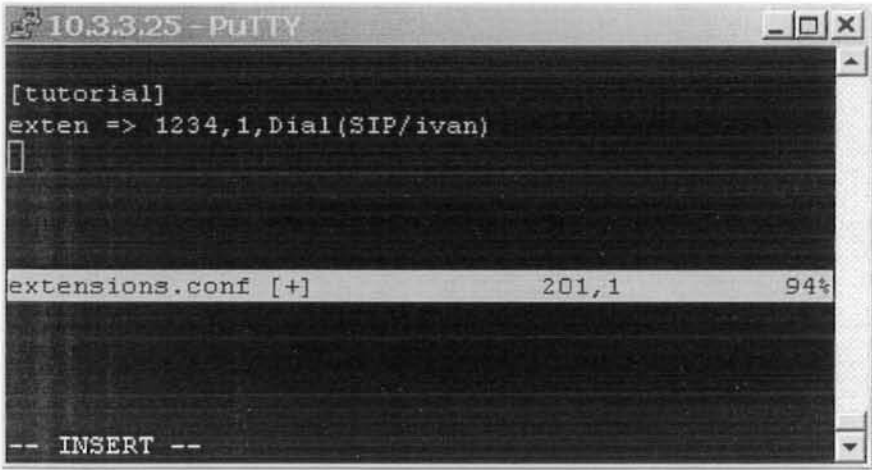
type=friend this means the user can place or receive calls. For INBOUND calls only, use 'peer' as type. For outbound calls only use 'user' as type.

username=ivan This declares that our user will be named 'ivan'

secret=pwd_ivan This creates the password for the user to login/authenticate on Asterisk host=dynamic This sets dynamic IP for the host. You may also define this as a static IP context=tutorial This defines the dial context for the user which in this case is tutorial.

This completes the user registration process on Asterisk. Now let us register an extension for the user.

2. Go to /etc/asterisk and open extensions.conf



[tutorial] This registers the context 'tutorial'

exten => 1234,1,Dial(SIP/ivan) when dialing number 1234, Asterisk will first Dial the user xlite through SIP protocol. The command is: exten => number, priority, Dial(protocol/user). The priority determines the sequence in which the extensions will be executed.

Start by registering the second user in the same way in /etc/asterisk/sip.conf

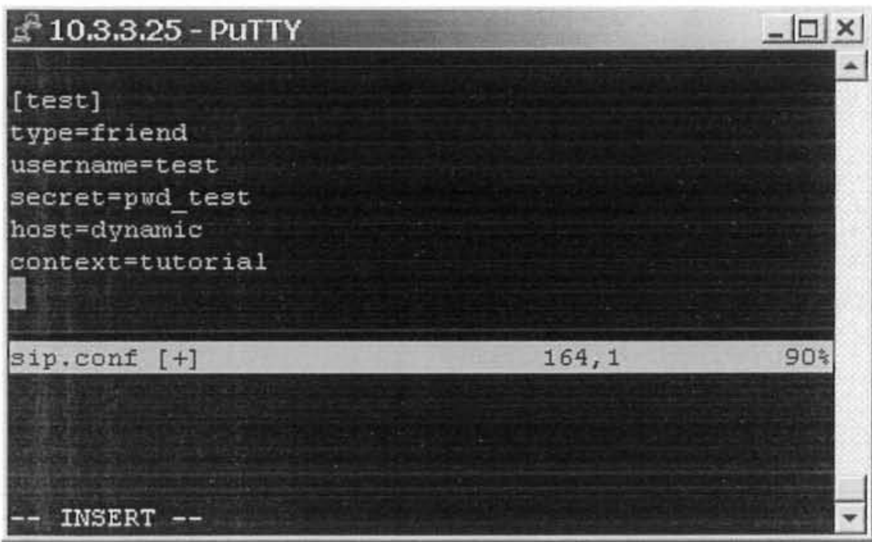
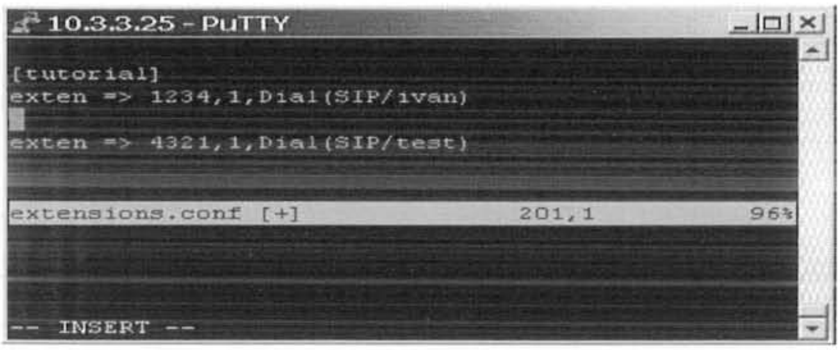


Figure: showing sip.conf

Register the extension(4321) in /etc/asterisk/extensions.conf in the same context = tutorial.



Now when user 'ivan' or any other user from the tutorial context dials 4321, the user 'test' will be called.

3. The final step is to register the user to a compatible softphone

Next Section Shows How to Configure the Softphone

Configuration of X-Lite Softphone

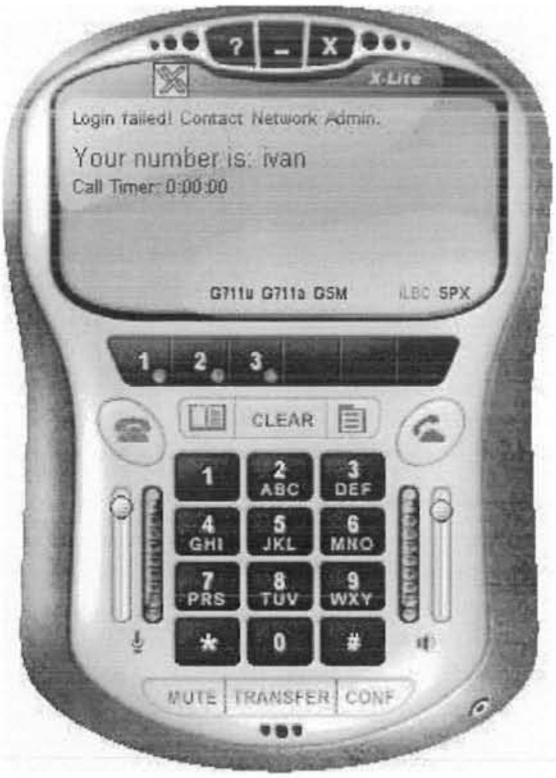


Figure: X-Lite SIP Phone

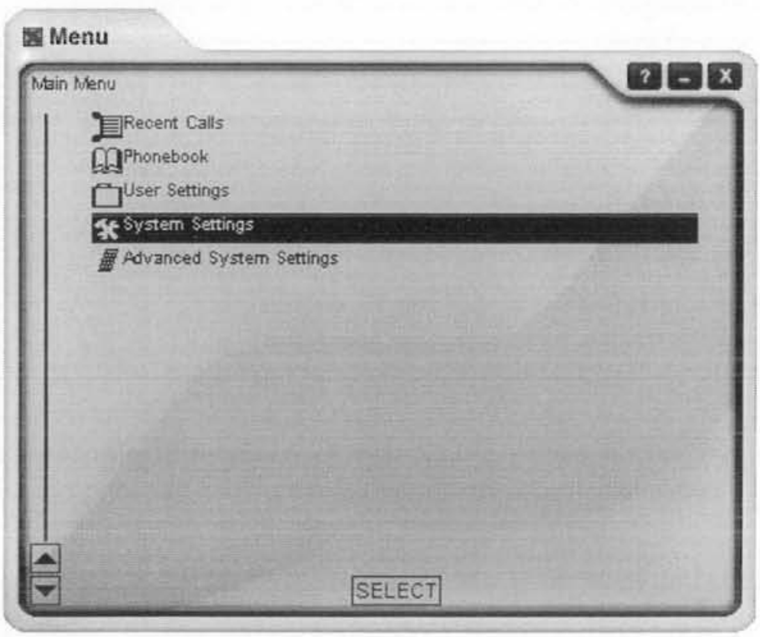
X-Lite: Configuring

A "softphone" is a piece of software that you download to your computer, with free trial enables we experience VoIP technology and experiment with set up and call quality. We will also need a headset, When you have downloaded it, you must configure it - this is a fairly straightforward process and is described briefly below.

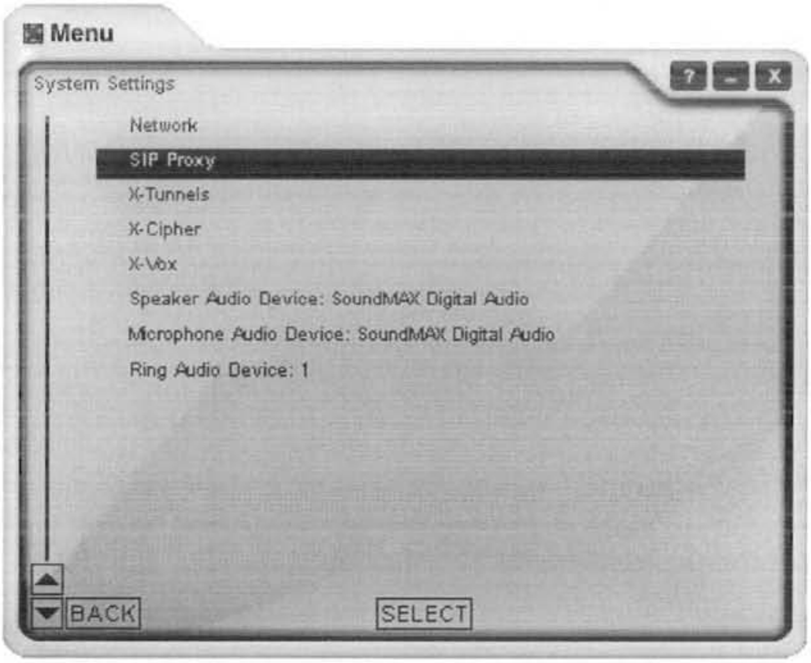
1. Press the menu button on the main display. This is the button to the right of the "clear" button in the middle of the window, as shown.

Figure: Showing X-Lite Softphone

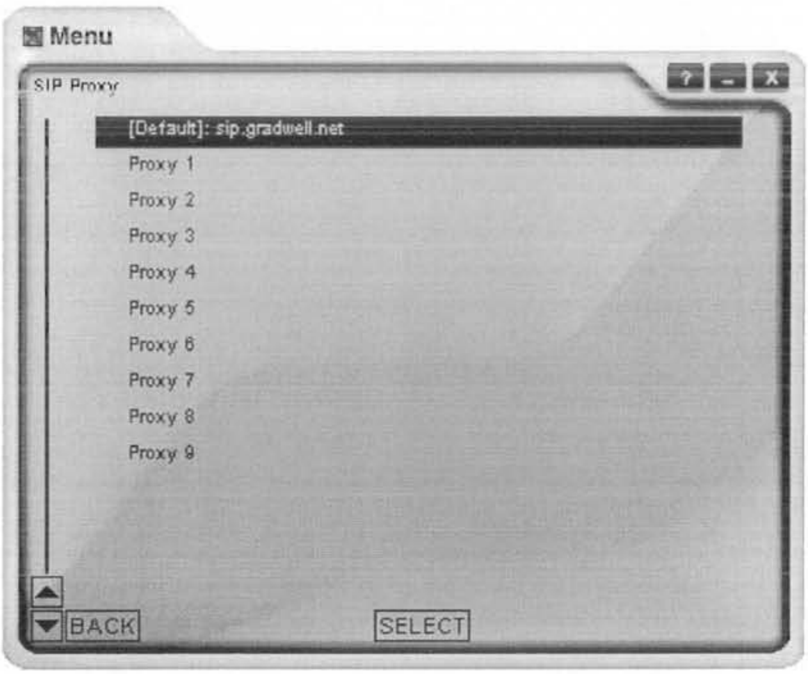
2. Select "System Settings" from the configuration window that appears.
3. Click on the System Setting as shown in the figure



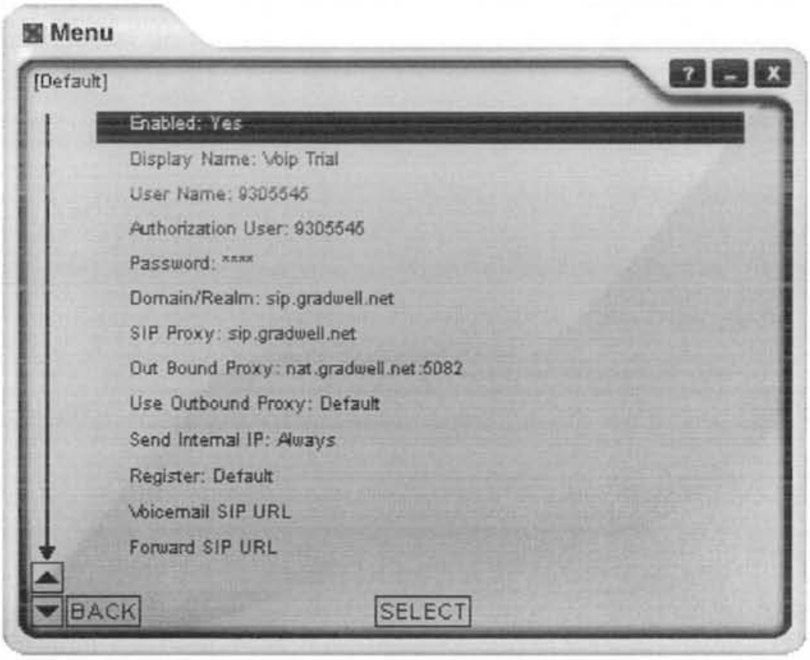
4. Select "SIP Proxy" from the next menu screen.



5. Select the item labeled "[Default]" from the list of SIP servers.



6. Enter your name, your user name and authorization user, both of which are your VoIP extension (nnnnnnn), and your password, which is specified in your trial confirmation email. Finally, set the Domain/Realm and SIP Proxy to sip.gradwell.net and the Outbound Proxy to nat.gradwell.net:5082.



7. Return to the main menu by pressing BACK three times.
8. Select Advanced System Settings, Audio Settings, and finally Silence Settings.
9. Double click Transmit Silence, double click Yes, and then click BACK.
10. Close the menu window using the cross sign in the upper right corner.
11. You will also need to set the following "System Settings | Network" You should have an entry for the Primary DNS Server and if you are using the Outbound Proxy enter the details Nat.gradwell.net:5082 into the "Out bound SIP Proxy" field.

Setting up Asterisk

Asterisk must (of course) recognize your softphone.
In the /etc/asterisk/sip.conf add the following lines

[Phone3]
type = friend
host = dynamic
defaultip = 192.168.1.103
dtmfmode = rfc2833
context = sip
callerid = "Phone3" <3>

[Phone4]
type = friend
host = dynamic
defaultip = 192.168.1.104
dtmfmode = rfc2833
context = sip
callerid = "Phone4" <4>

In the /etc/asterisk/extensions.conf add the following lines

exten => 3,1,Dial(SIP/Phone3,20,tr)
exten => 4,1,Dial(SIP/Phone4,20,tr)

Phone3 will now be reachable by dialing extension 3, it will ring for maximum 20 seconds. The same is true for Phone4

Troubleshooting

If things did not work the first time, you can track down the trouble using the Asterisk console. This can be used to display SIP messages coming through the system, as well as steps executed in interpreting the dial plan instructions.

:

```
asterisk –vvvvr
```

This will attach to the console with verbose mode set. Give the command

```
sip debug
```

to enable SIP debugging. You can turn off SIP debugging using the command

```
sip no debug
```

Setting up an X-Lite client on Asterisk

1) Adding a client entry to Asterisk's SIP configuration

edit the SIP configuration file /etc/asterisk/sip.conf

% sudo vi /etc/asterisk/sip.conf

enter an entry for each X-Lite client, for example

[12345] ; X-Lite client 12345

type=friend

secret=blah

auth=md5

nat=yes ; we assume clients are behind NAT

host=dynamic ; and have dynamic IP addresses

reinvite=no ; if so, we need to make them

canreinvite=no ; always go through Asterisk qualify=1000

dtmfmode=inband

callerid="Fred Flintstone" <12345>

disallow=all

allow=gsm ; add whatever other codecs we fancy

context=theflintstones ; use a context that exists ;-)

save the changes

2) Adding extensions for X-Lite clients

edit the Extensions configuration file /etc/asterisk/extensions.conf

% sudo vi /etc/asterisk/extensions.conf

;enter one catch-all extension, for example

[theflintstones] ; Our context for X-Lite clients

;

; Catch all five digit numbers, no leading zeroes

exten => _[123456789]XXXX,1,NoOp("call for "${EXTEN})

exten => _[123456789]XXXX,2,Dial(SIP/${EXTEN},60,tr)

exten => _[123456789]XXXX,3,Congestion

; alternatively, enter one specific extension for each client

; save the changes

3) Reload the new configuration

on the Asterisk console, reload the configuration files

switch1*CLI> reload

That's done!

CHAPTER 6

Conclusion and Recommendations

Conclusion

VoIP Telephony with Asterisk using Session Initiation Protocol (SIP)

We concluded that:

Asterisk has come a long way and has a longer way to go. It has become the de facto standard for voice switching and PBX functions in the open source environment. Host processing of TDM and DSP has made it possible to use Asterisk with inexpensive hardware in order to build flexible PBX platforms.

Applications of ASTERISK

PBX

Asterisk allows you to create a PBX that rivals the features and functionality of traditional telephony switches. Other PBXs are expensive, proprietary, and now passé. Asterisk is cost-effective, low-maintenance, and flexible enough to handle all voice and data networking.

Interactive Voice Response (IVR)

Asterisk's flexible IVR capability allows a user to interact with a database using a menu of pre-recorded voice-clips. Using MySQL and other popular databases, Asterisk can interact with the caller through touch tone inputs, record responses, query databases, and utilize AGI scripts to perform specific tasks. For example, a customer can authenticate a pre-paid calling card with a PIN queried from a database.

Media Server

Asterisk augments existing PBXs and Gateways with select features for either PSTN or IP protocols. Acting as an adjunct to a legacy system or soft switch, Asterisk can extend features and functionality by providing voicemail and conferencing services.

VoIP and Protocol Gateway

Asterisk's broad support of both traditional TDM and VoIP protocols permits the construction of lexible gateways between different channel types. Using Asterisk, protocol converters, translating between T1, E1, PRI, SIP, IAX, GR-303, MGCP, FXS, and many others.

VoIP Switch

Asterisk can act as a soft switch in addition to acting as a traditional TDM switch, allowing it to control a variety of devices including phones, gateways, media servers, and other Asterisk servers. It can handle virtually any VoIP protocol, including SIP, IAX, H.323, MGCP, and Skinny. Asterisk collects call detail records and provides a variety of billing options (including Open Settlement Protocol) and may be configured to carry media (especially useful for SIP+NAT situations) or to have devices send media directly to one another.

Future of Asterisk (*)

Areas for Improvement

Like any other software project, there is a number of areas in which Asterisk needs more attention.

Integrating Fax services, while trivial in most PBX platforms, is still undergoing development in Asterisk.

The user interface is a command line, which is very flexible, but at the same time hard for some people to use.

Performance of Asterisk has not been explored much. While the claim is that Asterisk can scale well, there is no test data showing that Asterisk can support large number of users without performance penalty. There is no easy way to cluster Asterisk installations to scale up and match the performance of legacy PBX systems.

Flexible APIs make it possible to write any PBX related application, but doing so is out of reach for the average telecommunications expert. It still takes a software person to create the fully functional Asterisk installation for a real enterprise.

BIBLIOGRAPHY

Internet

http://www.asterisk.org

http://www.digium.com

http://www.digium.com/handbook-draft.pdf

http://www.iaxtel.com

http://www.cornfed.com/iax.pdf

http://www.voip-info.org/wiki-asterisk

http://linuxdevices.com/articles/AT8678310302.html

Software

Asterisk@Home http://asteriskathome.sourceforge.net/
Asterisk http://www.asterisk.org/
Asterisk@Home Handbook http://asteriskathome.sourceforge.net/handbook/
Asterisk@Home support http://www.voip-info.org/tiki-index.php?
Asterisk Support http://www.voip-info.org/tiki-index.php?page=Asterisk

SoftPhones

Xlite http://xten.com/index.php?menu=products&smenu=xlite
SipXPhone http://www.sipfoundry.org/sipXphone/index.html

SER (Sip Express Router)

CHAPTER 1

General Information

About SIP Express Router (SER)

SIP Express Router (SER) is an industrial-strength, free VoIP server based on the Session Initiation Protocol (SIP, RFC3261). It is engineered to power IP telephony infrastructures up to large scale. The server keeps track of users, sets up VoIP sessions, relays instant messages and creates space for new plug-in applications. Its proven interoperability guarantees seamless integration with components from other vendors, eliminating the risk of a single-vendor trap. It has successfully participated in various interoperability tests in which it worked with the products of other leading SIP vendors.

ser can be also used with contributed applications. Currently, serweb, a ser web interface, SIPSak diagnostic tool and SEMS media server are available. Visit our site, http://www.iptel.org/, for more information on contributed packages.

About iptel.org

iptel.org is a know-how organization spun off from Germany's national research company FhG Fokus. One of the first SIP implementations ever, low-QoS enhancements, interoperability tests and VoIP-capable firewall control concepts are examples of well-known FhG's work.

Feature List

Based on the latest standards, the SIP Express Router (SER) includes support for registrar, proxy and redirect mode. Further it acts as an application server with support for instant messaging and presence including a 2G/SMS and Jabber gateway, a call control policy language, call number translation, private dial plans and accounting, ENUM, authorization and authentication (AAA) services. SER runs on Sun/Solaris, PC/Linux, PC/BSD, IPAQ/Linux platforms and supports both IPv4 and IPv6. Hosting multiple domains and database redundancy is supported.

*

Speed - With ser, thousands of calls per seconds are achievable even on low-cost platforms. This competitive capacity allows setting up networks which are inexpensive and easy to manage due to low number of devices required. The processing capacity makes dealing with many stress factors easier.

*

Flexibility - SER allows its users to define its behavior. Administrators may write textual scripts which determine SIP routing decisions, the main job of a proxy server. They may use the script to configure numerous parameters and introduce additional logic. For example, the scripts can determine for which destinations record routing should be performed, who will be authenticated, which transactions should be processed statefully, which requests will be proxied or redirected, etc.

*

Extensibility - SER's extensibility allows linking of new C code to ser to redefine or extend its logic. The new code can be developed independently on SER core and linked to it in run-time. The concept is similar to the module concept known for example in Apache Web server.

*

Portability. ser has been written in ANSI C. It has been extensively tested on PC/Linux and Sun/Solaris. Ports to BSD and IPAQ/Linux exist.

*

Interoperability. ser is based on the open SIP standard. It has undergone extensive tests with products of other vendors both in iptel.org labs and in the SIP Interoperability Tests (SIPIT).

Use Cases

This section illustrates the most frequent uses of SIP. In all these scenarios, the SIP Express Router (SER) can be easily deployed as the glue connecting all SIP components together, be it soft-phones, hard-phones, PSTN gateways or any other SIP-compliant devices.

Added-Value ISP Services

To attract customers, ISPs frequently offer applications bundled with IP access. With SIP, the providers can conveniently offer a variety of services running on top of a single infrastructure. Particularly, deploying VoIP and instant messaging and presence services is as easy as setting up a SIP server and guiding customers to use Windows Messenger. Additionally, the ISPs may offer advanced services such as PSTN termination, user-driven call handling or unified messaging all using the same infrastructure.

PC 2 Phone

Internet Telephony Service Providers (ITSPs) offer the service of interconnecting Internet telephony users using PC softphone or appliances to PSTN. Particularly with long-distance and international calls, competitive pricing can be achieved by routing the calls over the Internet.

PBX Replacement

Replacing a traditional PBX in an enterprise can achieve reasonable savings. Enterprises can deploy a single infrastructure for both voice and data and bridge distant locations over the Internet. Additionally, they can benefit of integration of voice and data.

CHAPTER 2

INTRODUCTION TO SER

Request Routing and SER Scripts

The most important concept of every SIP server is that of request routing. The request routing logic determines the next hop of a request. It can be for example used to implement user location service or enforce static routing to a gateway. Real-world deployments actually ask for quite complex routing logic, which needs to reflect static routes to PSTN gateways, dynamic routes to registered users, authentication policy, capabilities of SIP devices, etc.

SER's answer to this need for routing flexibility is a routing language, which allows administrators to define the SIP request processing logic in a detailed manner. They can for example easily split SIP traffic by method or destination, perform user location, trigger authentication, verify access permissions, and so on.

Conditional Statements

A very useful feature is the ability to make routing logic depend on a condition. A script condition may for example distinguish between request processing for served and foreign domains, IP and PSTN routes, it may split traffic by method or username, it may determine whether a request should be authenticated or not, etc.

CHAPTER 3

Server Operation

Recommended Operational Practices

Operation of a SIP server is not always easy task. Server administrators are challenged by broken or misconfigured user agents, network and host failures, hostile attacks and other stress-makers.

Keeping track of messages is good

Frequently, operational errors are discovered or reported with a delay. Users frustrated by an error frequently approach administrators and scream "even though my SIP requests were absolutely ok yesterday, they were mistakenly denied by your server".

Real-time Traffic Watching

Looking at SIP messages in real-time may help to gain understanding of problems. Though there are commercial tools available, using a simple, text-oriented tool such as ngrep makes the job very well thanks to SIP's textual nature.

CHAPTER 4

Application Writing

ser offers several ways to couple its functionality with applications. The coupling is bidirectional: ser can utilize external applications and external applications can utilize ser. An example of the former direction would be an external program determining a least-cost route for a called destination using a pricing table. An example of the latter case is a web application for server provisioning. Such an application may want to send instant messages, query all current user's locations and monitor server health. An existing web interface to ser, serweb, actually does all of it.

The easiest, language-independent way of using external logic from ser is provided by exec module. exec module allows ser to start external programs on receipt of a request. The programs can execute arbitrary logic and/or affect routing of SIP requests. A great benefit of this programming method is it is language-independent. Programmers may use programming languages that are effective or with which they are best familiar. Section

gives additional examples illustrating use of the exec module.

Another method for extending ser capabilities is to write new modules in C. This method takes deeper understanding of ser internals but gains the highest flexibility. Modules can implement arbitrary brand-new commands upon which ser scripts can rely on. Guidelines on module programming can be found in ser programmer's handbook available from iptel.org website.

To address needs of applications wishing to leverage ser, ser exports parts of its functionality via its built-in "Application FIFO server". This is a simple textual interface that allows any external applications to communicate with the server. It can be used to send instant messages, manipulate user contacts, watch server health, etc. Programs written in any language (PHP, shell scripts, Perl, C, etc.) can utilize this feature. How to use it is shown in Section 4.2.

Using exec Module

The easiest way is to couple ser with external applications via the exec module. This module allows execution of logic and URI manipulation by external applications on request receipt. While very simple, many useful services can be implemented this way. External applications can be written in any programming language and do not be aware of SIP at all. ser interacts with the application via standard input/output and environment variables.

CHAPTER 5

Complementary Applications

Serctl command-line tool

serctl is a command-line utility which allows to perform most of management tasks needed to operate ser: adding users, changing their passwords, watching server status, etc. Usage of utility is as follows:

Example 5-1. serctl usage

usage:

* subscribers *

serctl add <username> <password> <email> .. add a new subscriber (*)

serctl passwd <username> <passwd> .... change user's password (*)

serctl rm <username> .... delete a user (*)

serctl mail <username> .... send an email to a user

serctl alias show [<alias>] .... show aliases

serctl alias rm <alias> .... remove an alias

serctl alias add <alias> <uri> .... add an aliases

* access control lists *

serctl acl show [<username>] .... show user membership

serctl acl grant <username> <group> .... grant user membership (*)

serctl acl revoke <username> [<group>] .... grant user membership(s) (*)

* usrloc *

serctl ul show [<username>]................ show in-RAM online users

serctl ul rm <username> .... delete user's UsrLoc entries

serctl ul add <username> <uri> .... introduce a permanent UrLoc entry

serctl showdb [<username>] .... show online users flushed in DB

* server health *

serctl monitor .... show internal status

serctl ps .... show runnig processes

serctl fifo .... send raw commands to FIFO

Commands labeled with (*) will prompt for a MySQL password.

If the variable PW is set, the password will not be prompted.