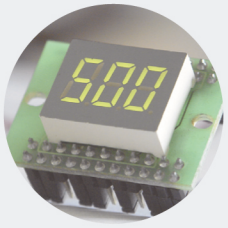
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO

MESTRADO EM ENGENHARIA ELECTROTÉCNICA - SISTEMAS ELÉCTRICOS DE ENERGIA






ANALYSIS AND TESTING OF VOIP-SUBSYSTEMS OF IPBRICK

SWAMY H K

Julho de 2017

POLITÉCNICO
DO PORTO

ANALYSIS AND TESTING OF VOIP-SUBSYSTEMS OF IPBRICK PORTUGAL, S.A.

SWAMY HK

Department of Electrical Engineering

Master's in Electrical Engineering – Power Systems

2017

Report prepared for the partial satisfaction of the requirements of DSEE Unit Course – Dissertation of the Master's in Electrical Engineering – Power Systems

Swamy HK, Nº 1150084, 1150084@isep.ipp.pt

Professor.Sergio Ramos , scr@isep.ipp.pt

Company: IP BRICK, Oporto, Portugal

Supervisor: Mr. Miguel Ramalhão, ramalhao@ipbrick.com

Department of Electrical Engineering

Master's in Electrical Engineering – Power Systems

2017

Where there is righteousness in the heart,
There is beauty in the character.
When there is beauty in the character,
There is harmony in the home.
When there is harmony in the home,
There is order in the nation.
When there is order in the nation,
There is peace in the world."

- ***APJ Abdul Kalam***

ACKNOWLEDGMENT

At the end, the last stage of many successful stages over the course of my academic career comes to an end and its pride and pleasure that I completed it. During the development of this thesis work, the support and encouragement of several people were fundamental. So I would like to thank all those who have left their footprints in my way.

This journey would not have been possible without the support of my professors, supervisors and mentors, family and friends. To my family thank you for encouraging me in all my pursuits and inspiring me to follow my dreams. I am especially grateful to my family and friends who supported me emotionally and financially. I always knew that you believed in me and wanted the best for me. Thank you for teaching me that my job in life was to learn to be happy, and understand myself only then could I know and understand others.

I would like to express my sincere gratitude to Instituto Superior de engenharia Do Porto and Ipbrick, oporto, portugal for the continuous support and encouragement. I express my sincere and special thanks to Prof. Teresa Alexandra Nogueira, Course Director ISEP and Mr. Miguel Ramalhao, Head of VOIP department and thesis Supervisor , IPBRICK Portugal S.A. for giving me an opportunity to use the facilities in College and Company premises to complete the Master's thesis respectively.

I also extend my gratitude to Prof. Sergio Ramos Thesis Supervisor in ISEP and Mr.Vittor Vidal team leader and Co-coordinator of VOIP department,IPBRICK, Oporto for their guidance and sincere co-operation. With magnificent pleasure my profound sense of gratitude to VOIP Department Engineers, Other department Engineers, and Co-workers of IPBRICK,Oporto for their valuable support and encouragement to do this Master's thesis. Thank you very much to everyone!

ABSTRACT

Technology in which communication done using IP(Internet Protocol) alternative for the traditional analog systems is voip (voice over internet protocol) .One of the emerging or attractive communication systems in this era is voip. Several technologies within the voip are emerging and more come to near future, services offered by this technology needs internet connections and/or telephone connections. It offers services such as making audio and video calls. VoIP is a medium that converts the analog signal to digital signals[1].

In this dissertation mainly focused on ipbrick voip-subsystems such as webrtc ,asterisk13.8 pbx server and kamilio sip proxy. These are free and open-source technologies available in the marketplace. Integration of these technologies provides real-time communication between the systems such as voice,video and text. Webrtc enables web browsers to have native support for real-time voice, video and data capabilities. As such, end-users do not require additional add-ons or plug-ins to utilize real-time voice and video communication. To allow webrtc to make calls to non-webrtc voip applications, A initiation protocol is needed and one such protocol is session initiation protocol (SIP),which is the standard protocol used for initializing, changing and terminating sessions for multimedia today. It is particularly known for its use in voip applications. Asterisk13.8pbx supports webrtc and it acts as media gateway.

In this thesis, we are evaluating and comparing previous and current versions of the ipbrick voip-subsystem which is previously having lab environment of ipbrick OS v6.1 with Asterisk v1.8 as a pbx server and webrtc application such as webrtc2sip and SIP Proxy software called Kamailio which connects two endpoints.

In this thesis we are testing ipbrick voip-subsystem with current version of ipbrick OS v6.2 with Asterisk v13.8 as a new pbx server and Webrtc application (SIPML5) on the browser side and a phone on the server side (softphone) establish a phone call between them. The SIP Proxy server Kamailio will act as a intermediary connection, connecting the two endpoints using websockets.

CONTENTS

CHAPTER 1

INTRODUCTION

The Present dissertation was elaborated within the scope of the Master's in Electrical Engineering in Power Systems. The project was developed in an IPBRICK lab environment, in the company of IPBRICK,Portugal,S.A.

1.1 IPBRICK

Mission

IPBRICK, S.A. is a company that has the mission to constantly innovate in communications solutions for companies.

Vision

Its vision is to be a reference company in the development and implementation of On Premises and Private Cloud solutions.

1.1.1 IPBrick.GT - Voice Communications

Integrated PBX features are managed through the IPBrick.GT graphical interface. Assign the VoIP addresses and the telephone extensions to the indicated Terminal. IPBrick allows the manipulation of numbers within the inbound and outbound scheme and incorporates queue management and flexible routing policies[2].

For out-of-LAN calls the data transport will be negotiated either directly with the recipient or will be diverted to a SIP provider. SIP providers offer VoIP gateways to bring calls to telephony endpoints (ISDN, PSTN, cellular radio). Inside your LAN, IPBrick.GT acts as the default gateway (SIP Proxy, VoIP Registrar) and allows as many incoming calls as you need.

The handset sends the calls through the connected PBX. This PBX behaves is transparent to IPBrick.GT, allowing a smart way of routing to negotiate calls via traditional or via VoIP to the Internet.

1.1.2 IPBrick OS v6.2 Release Notes

VoIP improvements:

•Adds support to SIP trunks over TCP (required for Skype for Business integration)

•Adds a new sub-menu "Telephony » Session Border Controller" to centralize SBC features

•Adds auto provisioning support for the models: Cisco 7905g, Cisco 7911G,Cisco 7912G, Cisco 7937G and Cisco 7942G

•Improves auto provisioning support for the models: Polycom soundpoint 300ip, Snom 300, Snom 360 and Snom 715

•Improves the detection of Denial of Service(DoS) attacks on internal networks

•Adds a new option to DISA, that allows to configure the Caller ID

•Adds a new option that allows to configure High Resolution Time on the fields billsec and duration, in the CSV file (Master.csv) of the records of the stored calls details

•Adds a new option that allows call screening

•Adds a new option that allows to block SIP DDIs with custom SIP messages (SIP Blacklisting)

•Adds a new page for IVR calls statistics (IVR surveys). This page presents features such as search for IVR calls, check the options dialled and export these records to csv

•Adds an option to timeout the agents' sessions. This timeout can be global or personalized by agent.

•The management of agents' sessions was improved for users, considering their access classes

•Adds an option to the user call using PIN# destination_number, even when the agents have passwords

•Adds an option to configure the separator (#) for calls using PIN# destination_number

•Support for BRI and PRI Telephony Cards was improved for better performance

•Adds support to monitor and login an agent with the same BLF

•Adds a new option "Allow global access by telephone" on voicemail which allows to enable/disable the access to phones, users, groups and attendance sequences

•Adds the possibility to define IPs on "VoIP domain alias"[3].

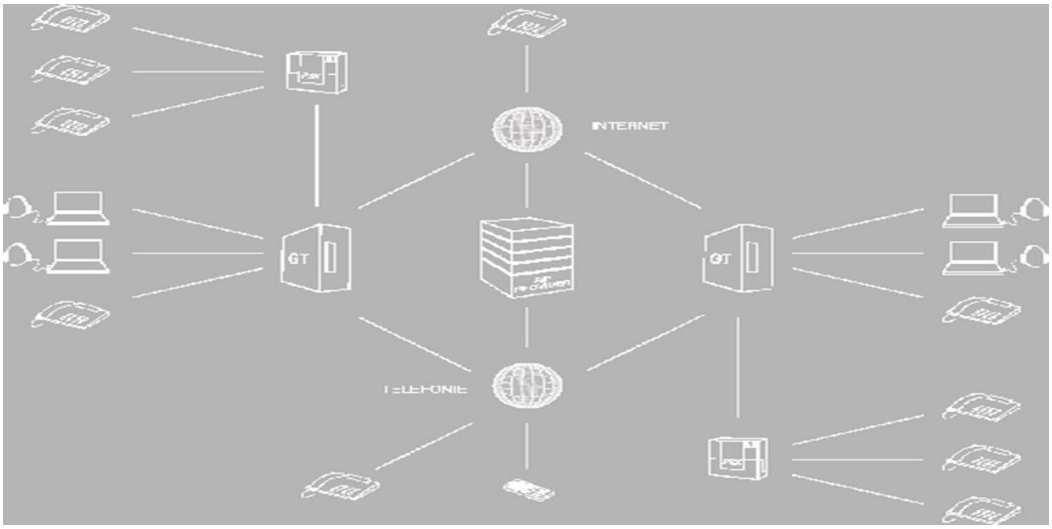


Figure 1.Ipbrick Network Architecture

1.2 Research objectives

- The main aim of this project is to analysing and evaluating ipbrick Voip-subsystem by comparing ipbrick OS V6.1 voip system with ipbrick OS v6.2 Voip sub-systems.
- Tested and analyzing ipbrick voip-subsystem with current version of ipbrick OS v6.2 compatibility with previous ipbrick OS v6.1.
- Research and Installing new asterisk pbx server which supports webrtc where previous asterisk server does not support it.
- Tested ipbrick pbx server by upgrading Asterisk server to new version and tested and validated ipbrick voip features.
- Research webrtc and tested webrtc sipml5 which connects calls via kamailio sip proxy to asterisk server.

On the technical side, lab setup requires three important components such as webrtc application (webphone),kamailio sip proxy and asterisk pbx as medium gateway and server side softphone.

WebRTC is a technology for transmitting real-time voice, video and data in web browsers released and open sourced by Google in 2011 [4].WebRTC enables web browsers to have native support for real-time voice, video and data capabilities. As such, end-users do not require additional add-ons or plug-ins to utilize real-time voice and video communication so it uses websockets for connecting to kamailio sip proxy which is an Open Source SIP Server released under GPL, able to handle thousands of call setups per second. Kamailio can be used to build large platforms for VoIP and realtime communications such as WebRTC, Instant messaging and other applications. Moreover, it can be easily used for scaling up SIP-to-PSTN gateways, PBX systems or media servers like Asterisk™[5].

Asterisk is a software implementation of a telephone private branch exchange (PBX) it allows attached telephones to make calls to one another and to connect to other telephone services such as the public switched telephone network (PSTN) and Voice over Internet protocol (VoIP) services.

1.3 Structure of the thesis

This section gives basic information about the thesis. This section covers the introduction and few basic concepts.

Chapter 1: Introduction

This section gives the introduction to the voip concepts and its evaluation and few concepts about ipbrick network architecture, ipbrickOS release and research objectives.

Chapter 2:Technology and Regulation

This section briefly explains the technologies and regulations used in thesis consists of webrtc,voip,sip,websockets.

Chapter 3:Technical Requirements

This section briefly explains the network design and configuration process of each and every node of network design. Implementation steps are also given in this section and also all the results collected in the implementation are done in this section.

Chapter 4:Conclusion and future work

Final Conclusions of the entire thesis are presented in brief here and based on those final conclusions few suggestions are given for network designers.

1.4 Who should read this report?

This report was written for those with a basic understanding of Voice over Internet Protocol (VoIP) technology or web development such as webrtc technology and sip technology. Who are interested in connecting their existing voip solutions for use in conjunction with web browsers, such as WebRTC technology.

CHAPTER 2

TECHNOLOGY AND REGULATION

2.1 Voice over Internet Protocol (VoIP) technology

VoIP is a technology that emerged in the beginning of 1990 and enables phone calls over the Internet protocol. The problems that existed when using telephones over an analog network also existed on the VoIP network. The biggest advantage using IP networks is the costs, as the toll charges can be skipped. Another big advantage is that it can carry 5 - 10 times the number of phone calls over the same bandwidth compared to PSTN.

Problems such as data loss, jitter and echo that could exist on the analog network now exist in VoIP calls. The biggest issue is that current IP networks were not intended to carry real time traffic. VoIP is trying to reconcile this problem, sending real-time data over a non-real-time network .

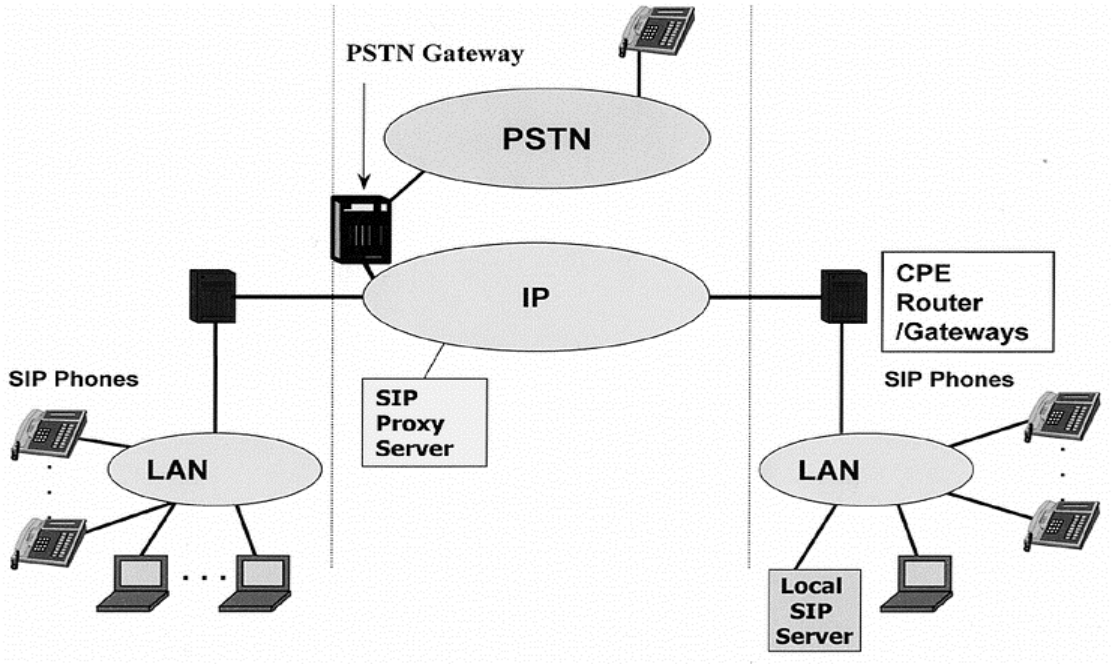


Figure 2. End to End Voip Technology

2.2 What is SIP and How Does It Work?

SIP (Session Initiation Protocol) is a protocol used in VoIP communications allowing users to make voice and video calls, mostly for free. I will keep the definition in this article to something simple and practical. If you want a more technical insight of SIP read its profile[6].

2.2.1 Why Use SIP?

SIP allows people around the world to communicate using their computers and mobile devices over the Internet. It is an important part of Internet Telephony and allows you to harness the benefits of VoIP (voice over IP) and have a rich communication experience.

But the most interesting benefit we derive from SIP is the cutting down of communication costs. Calls (voice or video) between SIP users are free, worldwide. There are no boundaries and no restrictive laws or charges. Even the SIP apps and SIP addresses are obtained free.

SIP as a protocol is also very powerful and efficient in many ways. Many organizations use SIP for their internal and external communication, centred on a PBX.

2.2.2 What is Required?

If you want to communicate through SIP, you need the following:

A SIP address/account. This is obtained free from many providers. Just register online; you can have yours now for free. Here are links to help you get a free SIP account.

- What is a SIP Address?

- Providers of Free SIP Addresses

- Registering for a SIP Address

A SIP client, this is a program that you install on your computer or mobile device. It contains softphone functionality and some other features and provides an interface for you to communicate. There are different types of SIP clients. Among the most common ones are the applications offered free by VoIP service providers, to use with their VoIP services. Some of them support SIP. But you have clients that are built for SIP and do not depend on any service. You can use them with any SIP account, and can even use them within PBX environment. Here is a list of the most popular free SIP clients on the market.

Here is how to configure a SIP client. An Internet connection with sufficient bandwidth for voice and video communication. Not much is required for voice communication, especially if you are using enhanced codecs for low bandwidth consumption, but you need solid bandwidth for voice communication. You will prefer a DSL connection.

Hearing and talking devices. You need what it takes to get your voice through and to hear what is being said, and what is being presented in images. A headset, earpieces, a microphone and a web cam for video communication.

Buddies to talk to. maybe this is the first item in the list that's checked. You may have buddies, but they need to be using SIP too, if you want the calls to be free. Share the SIP addresses just like you do phone numbers.

2.2.3 How SIP Works?

The figure 2.2 below describes a very basic SIP call flow case where SIP Phone A is the caller and SIP Phone B is the recipient . Phone A and Phone B connects the call with help of SIP Proxy.

Caller Phone A wants to initiate a call and sends a initial "INVITE" to Phone B. The "INVITE" contains signaling information in various headers such as phone numbers and SIP path. The "INVITE" also contains a Session Description Protocol (SDP) that describes which media settings that A supports and prefers, those media settings can be codecs and media addresses.

When Phone B receives the "INVITE", it will respond with a "100 Trying" SIP response which means that it has accepted the "INVITE" from Phone A and is processing it. When the phone at Phone B's starts to ring, a "180 Ringing" is sent back to notify Phone A.Callee Phone B answers the phone and responds with a "200 OK" SIP message to Phone A.

This also contains a SDP body with media settings that Phone B supports and prefers. It is important to note that Phone A and Phone B are negotiating which media settings will be used during the call. Finally Phone A replies with an "ACK" to acknowledge and confirm that the "200 OK" were received successfully. From this point, the two parties can speak to each other using the negotiated media parameters.

When one party ends the calls, a "BYE" message is sent to the other party, which responds with a "200 OK" confirming call termination.

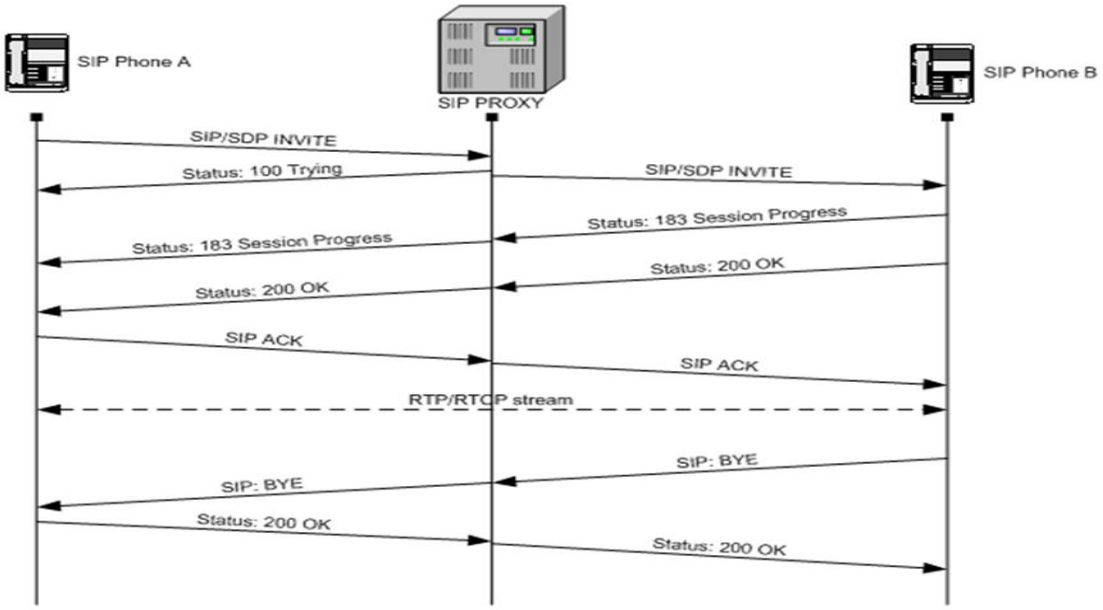


Figure 3.Basic scenario of how a successful SIP call can be established and terminated.

2.3 Kamailio SIP Proxy server

SIP user-agents can communicate directly with each other without the involvement of other devices. Despite this, an SIP Server is often added to create a more practical system which facilitates end-to-end communication utilized as a public service[7].

The Request For Comments (RFC) document 3261, SIP: Session Initiation Protocol, describes an SIP Proxy server as a entity that acts both as a server and a client, to make requests on behalf of other clients. The proxy server primarily handles routing, ensuring a request is sent to another entity closer to the intended user. The proxy also enforces policies and allows users to make calls.

The Kamailio® SIP server is a leading Open Source software for building SIP services such as a SIP proxy, SIP Presence Server, SIP location server and much more. With a rich configuration language, modularity and continuos development Kamailio is the choise for building enterprise as well as carrier solutions. Kamailio runs on Unix and Linux systems, ranging from embedded systems to large scale multi-core servers.

Kamailio is the result of over 10 years of development, in the SIP express router project, the Open SER project and the SIP Router project. Kamailio is the result of a merger of multiple projects, a collaboration process that started on November 4, 2008.

To understand the meaning of Kamailio in the SIP please see the Kamailio SIP Router releases pages.

Kamailio is released under GNU Public License v2 (GPLv2). Beginning with 3.0.0, the application includes parts of code under BSD license that can be used such as individual components.

Kamailio can be:

SIP proxy server

SIP registrar server

SIP location server

SIP application server

SIP dispatcher server

SIP Websocket server

Each of these parts is responsible for different functions. Where the proxy server can combine and provide one or multiple of the aforementioned functions.

2.4 What is Media Gateway and How asterisk does?

A media gateway is a translation device or service that converts media streams between disparate telecommunications technologies such as POTS, SS7, Next Generation Networks (2G, 2.5G and 3G radio access networks) or private branch exchange (PBX) systems. Media gateways enable multimedia communications across packet networks using transport protocols such as Asynchronous Transfer Mode (ATM) and Internet Protocol (IP).

Because the media gateway connects different types of networks, one of its main functions is to convert between different transmission and coding techniques. Media streaming functions such as echo cancellation, DTMF, and tone sender are also located in the media gateway.

Media gateways are often controlled by a separate Media Gateway Controller which provides the call control and signaling functionality. Communication between media gateways and Call Agents is achieved by means of protocols such as MGCP or Megaco (H.248) or Session Initiation Protocol (SIP). Modern media gateways used with SIP are often stand-alone units with their own call and signaling control integrated and can function as independent, intelligent SIP end-points.

Voice over Internet Protocol (VoIP) media gateways perform the conversion between Time-division multiplexing (TDM) voice to a media streaming protocol, such as the Real-time Transport Protocol, (RTP), as well as a signaling protocol used in the VoIP system.

Mobile access media gateways connect the radio access networks of a public land mobile network PLMN to a next-generation core network. 3GPP standards define the functionality of CS-MGW and IMS-MGW for UTRAN and GERAN based PLMNs.

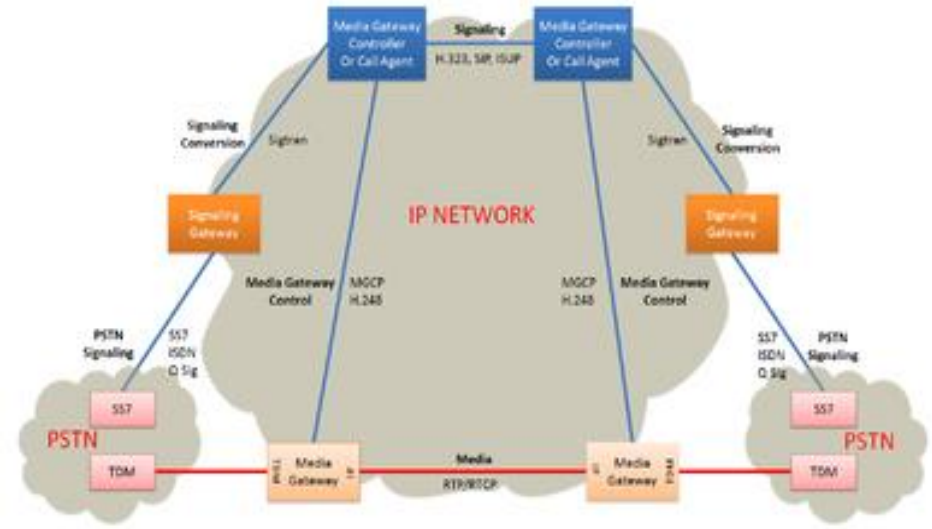


Figure4.Media gateways are used for transcoding media between PSTN and IP networks

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 other custom solutions. It is used by small businesses, large businesses, call centers, carriers and government agencies, worldwide. Asterisk is free and open source. Asterisk is sponsored by Digium.

Today there are more than one million Asterisk-based communications systems in use, in more than 170 countries. Asterisk is used by almost the entire Fortune 1000 list of customers. Most often deployed by system integrators and developers, Asterisk can become the basis for a complete business phone system, or used to enhance or extend an existing system, or to bridge a gap between systems.

2.5 IPBRICK PRODUCT :IPBrick.GT - Appliance UCoIP

Figure 5.Ipbrick.GT server

IPBrick.GT completes the IPBrick.IC solution with hardware designed to enable optimized integration of Voice/Data. IPBrick.GT implements the much-needed integration between traditional telephony and new VoIP-based telephony, SIP phones and SIP providers.

IPBrick.GT implements a complete PSTN / VoIP gateway allowing direct connection to the PBX (ISDN E1 / Bri and analogous links), PSTN provider, LAN and Internet. Intelligent routing allows voice to flow both via VoIP and the traditional ISDN telephony network or the like. IPBrick integrates traffic shaping to prioritize and optimize telephone connections. The number of active and parallel calls will only be limited by the contracted bandwidth and appliance comes with iPBrick.IC preinstalled and ready to use.

2.5.1 IPBrick.GT – Features

1.VoIP communications based on standard Internet protocols (SIP, RTP)

2.Security in VoIP data and voice delivery through definable VPN tunnels

3.Smart negotiation of voice calls for external servers (VoIP-Servers), SIP Provider and the traditional telephone network

4.Isolation of the ISDN / Analog coupling to your PBX

5.A migration path from your ISDN / Analog phones and PBX to VoIP technology

6.High availability for VoIP companies

7.Comprehensive QoS application with integrated bandwidth control

2.5.2 Technical Specifications

Procesador: Celeron D

RAM: 512MB

HD: 40GB

LAN: 2x Fast-Ethernet, 2x Gigabit-Ethernet

ISDN: 2x E1

OS: IPBrick.IC

2.6 Web Real-Time Communication (WebRTC)?

WebRTC was announced and open sourced by Google in May 2011 and was an effort started by the industry to add real time video and voice communications capabilities to browsers. The specification is currently not finalized and is still undergoing work to standardize the browser APIs and relevant protocols an increasing number of web browsers announce their time-lines for implementation and support for the WebRTC technology.

Unlike other traditional real-time systems, such as SIP-based soft phones, WebRTC communication is directly controlled by a web server through a JavaScript (JS) API as shown in figure 2.1

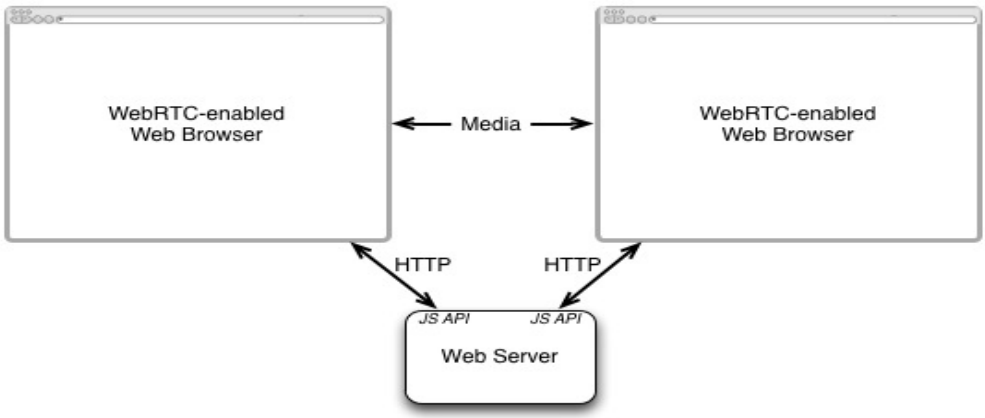


Figure 7: A simple WebRTC system

Browser vendors Google and Mozilla Foundation announced interoperability for WebRTC in February 2013 as they demonstrated a video where a video conference call was transmitted between the browsers.

Johnston, Yoakum and Singh believe that WebRTC will have a major impact on the way enterprises communicates and how it communicates with consumers. This makes WebRTC a very promising technology**.**

The WebRTC technology reuses existing standards such as mandatory Secure Real-time Transport Protocol (SRTP) for media transmission and mandatory Interactive Connectivity Establishment (ICE) traversal through Network Address Translations (NAT) and firewalls . The signaling method is unspecified and left to the developer to decide which is most suited.

Common methods when developing pure web applications are Hypertext Transfer Protocol (HTTP) and WebSockets (WS). The HTTP and WS protocols are responsible for exchanging the call control and transmitting session description information among the participants.

2.6.1 WebSockets - a bidirectional communication channel

WebSocket is a protocol specified by The Internet Engineering Task Force (IETF) and is a part of the HTML5 API mentioned in part 2.2.2.1. It creates a bidirectional client- server connection from the JavaScript code in the browser to the web server. Any data with a packet boundary can be sent over transmission Control Protocol (TCP) in either direction, once the connection is established[8].

The current attempts to provide real-time data communication on the web involves polling and server-side push technologies. These solutions often involve keeping two different connections running, which restricts the sent data to data that has been explicitly approved. Alternatively, a Web Socket connection on the other hand, allows the server to send data whenever it wants with lower latency than using long polling .

A client and server can establish a Web Socket connection by upgrading from the HTTP protocol to the Web Socket protocol in the initial handshake, see example below:

Browser request and server response

GET /text HTTP/1.1\r\n

Upgrade: WebSocket\r\n

Connection: Upgrade\r\n

Host: www.websocket.org\r\n

...\r\n

HTTP/1.1 101 WebSocket Protocol Handshake\r\n

Upgrade: WebSocket\r\n

Connection: Upgrade\r\n

...\r\n

Once the connection is established, data frames can be sent in the bidirectional communication channel.Where both text and binary frames, can be sent in either direction and at the same time.

2.6.2 Webrtc is enabled in chrome 55.0 Browser

We can enable and disable the webrtc in browsers but by default webrtc is enabled in chrome and Firefox [9].

1.Open browsers

2.Type about:config in address bar and enter press

3.Type media.peerconnection.enabled

4.By default webrtc is true

5.Check whether webrtc is enabled or not by going to www.whoer.net

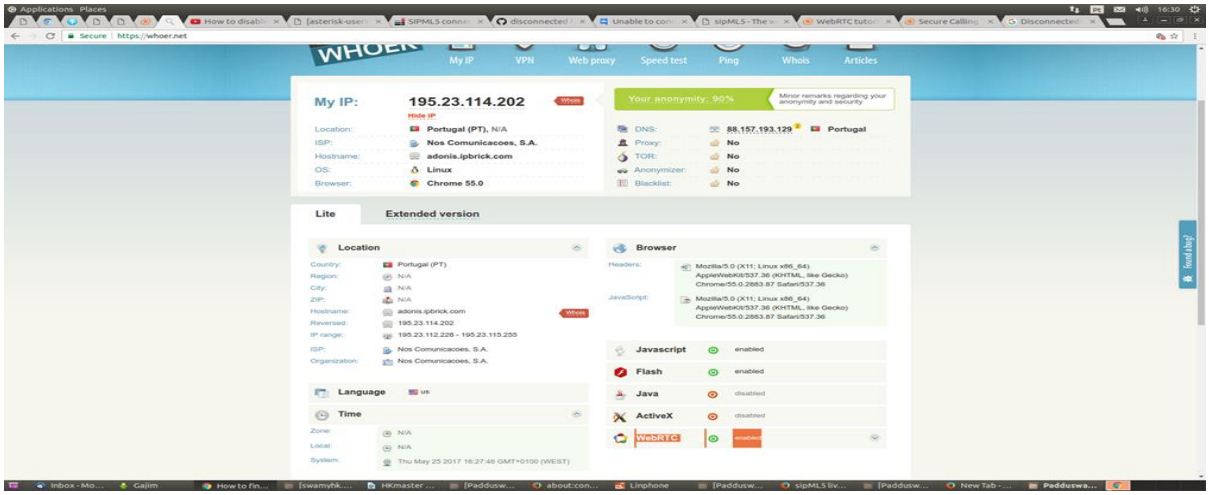


Figure 7.Webrtc enabled in browser

2.7 Interconnecting Web Browsers and Voice over Internet Protocol (VoIP)

WebRTC is independent of Web Sockets and you can use AJAX, a concept like server push, Web Sockets or plain HTTP for signaling. This is because the authors at IETF decided early that they did not want SIP to be a part of the browser. Developers therefore, had to do the signaling in JavaScript, the authors wanted to keep the signaling part outside the scope of the browser.

There are two ways to inter-work SIP systems and WebRTC enabled browsers. One can either do translation at the SIP Proxy server or implement SIP in JavaScript. SIP Proxies already supports several underlying transports such as TCP and User Datagram Protocol (UDP) and can be extended to support Web Sockets as another transport mechanism if needed.

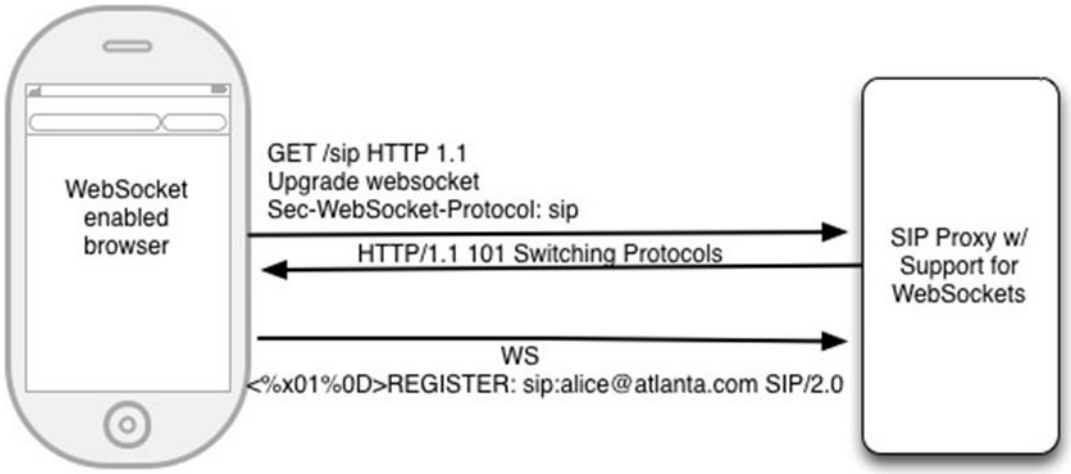


Figure 8.The flow for using SIP over Web Sockets

The SIP over Web Sockets protocol allows developers to implement SIP in JavaScript, whilst promoting end-to-end media paths if possible. Translating at the gateway requires both the signaling and media path go through the gateway. SIP over Web Sockets is not affected by NAT translations the same way as traditional SIP is, since it is sending data over TCP sockets.

By using SIP over Web Sockets one can communicate with the server software and bridge existing telecommunication SIP infrastructure.

2.8 SIP over Web Sockets

As mentioned before, prior to sending SIP requests, a SIP Web Socket Client connects to a SIP Web Socket Server and performs the connection handshake. Each SIP message must be carried within a single Web Socket message where a Web Socket message must not contain more than one SIP message[10]. First, let us define two new terms

- **SIP Web Socket Client** - A SIP entity capable of opening outbound connections to Web Socket servers and communicating using the Web Socket SIP sub-protocol.

- **SIP Web Socket Server** - A SIP entity capable of listening for inbound connections from Web Socket clients and communicating using the Web Socket SIP sub-protocol Examples of SIP Web Socket Clients implemented in web browsers is JsSIP, SIPML5 and sip-js.

SIP Proxy Servers with support for incoming Web Sockets connections on the market today are OfficeSIP, Kamailio, Asterisk and FreeSwitch.

CHAPTER 3

TECHNICAL REQUIREMENTS

3.1 Case Study:

3.1.1 Asterisk13.8 Pbx

Asterisk 13.8 pbx is an open source software, first released in 2016. Asterisk can be best described as a "back-to-back user agent" (B2BUA). B2BUAs are often implemented in media gateways[11].

Asterisk 13.8 provided us with the functionality to listen for incoming Web Socket connections and support for WebRTC. A major advantage with this software is that Asterisk 13.8 does support secure Web Sockets as of today.asterisk13.8 has been successfully implemented in our lab environment as shown in fig below.

3.1.2 Asterisk 13.8 Configuration:

Before doing the Asterisk installation, we have to update our system so that we have everything needed to compile plus the packages we need[12].

Go to your source

cd /usr/src/voip/

Apt -get update

cd /usr/src/voip/

apt -get build-essentials

apt-get -y --force-yes install postgresql-server-dev-9.1 libsnmp-dev unixodbc-dev libcurl3-dev libiksemel-dev libiksemel3 libssl-dev libtiff4-dev debhelper dpatch doxygen autotools-dev libncurses5-dev libfishsound1-dev libspeex-dev libxml2-dev subversion libspandsp-dev libjack-dev libresample1-dev libspeexdsp-dev liblua5.1-dev libneon27-dev libical-dev libgmime-2.6-dev libsrtp0-dev libopenais-dev libkqueue-dev libpopt-dev.

aptitude -y install libsnmp-dev libopenais-dev libsrtp-dev libgmime-2.6-dev libiksemel-dev libneon27-dev libical-dev libresample-dev libcurl4-openssl-dev libvorbis-dev libspeex-dev unixodbc-dev libltdl-dev portaudio19-dev.

5.Next step is to add the libsrtp library support. Do the following from your terminal CLI

[ipbrick]# cd /usr/src/voip

[ipbrick]# wget http://srtp.sourceforge.net/srtp-1.4.2.tgz

[ipbrick]# tar zxvf srtp-1.4.2.tgz

[ipbrick]# cd srtp

[ipbrick]# autoconf

[ipbrick]# ./configure

[iipbrick]# make

[ipbrick]# make install

[ipbrick]# cp /usr/local/lib/libsrtp.a/lib

[ipbrick]# cd ..

6.Compile another library Jansson

[ipbrick]#cd /usr/src/voip

[ipbrick]# wget http://www.digip.org/jansson/releases/jansson-2.5.tar.gz

[ipbrick]# tar zxvf jansson-2.5.tar.gz

[ipbrcik]# cd jansson-2.5

[ipbrick]# ./configure

[ipbrcik]# make

[ipbrick]# make install

[ipbrick]# cd ..

3.1.3 Now it's time for Asterisk itself(Install Asterisk):

1.[ipbrick]# mkdir /usr/src/voip/

[ipbrick]# wget http://downloads.asterisk.org/pub/telephony/asterisk/asterisk-certified-13.8-current.tar.gz

[ipbrick]# tar –xzvf asterisk-certified-13.8-current.tar.gz

[ipbrick]# /usr/src/voip/asterisk-certified-13.8-cert2

[ipbrick]# make clean

[ipbrick]# ./configure

[ipbrick]# make

[ipbrick]# make menuselect.makeopts

[ipbrick]# menuselect/menuselect –enable format_mp3 –enable res_config_mysql –enable app_mysql –enable app_saycountpl –enable cdr_mysql –enable EXTRA-SOUNDS-EN-GSM

3.1.4 Create directory and parent directory

[ipbrick]# mkdir -p /usr/src/voip/ asterisk-certified-13.8/etc/init.d

[ipbrick]# mkdir -p /usr/src/voip/ asterisk-certified-13.8 /etc/default

[ipbrick]# mkdir -p /usr/src/voip/asterisk-certified-13.8/etc/logrotate

[ipbrick]# mkdir -p /usr/src/voip/asterisk-certified-13.8/DEBIAN

make install DESTDIR=/usr/src/voip/ asterisk-certified-13.8

make config DESTDIR=/usr/src/voip/ asterisk-certified-13.8

make samples DESTDIR=/usr/src/voip/ asterisk-certified-13.8

3.1.5 Change ownerships rights

chown -R asterisk:asterisk/usr/src/voip/asterisk-13.8/var/log/asterisk

chown -R asterisk:asterisk /usr/src/voip/asterisk-13.8/var/run/asterisk

chown -R asterisk:asterisk /usr/src/voip/asterisk-13.8/var/lib/asterisk

chown -R asterisk:asterisk /usr/src/voip/asterisk-13.8/var/spool/asterisk

mkdir -p/usr/src/voip/asterisk-13.8/usr/share/asterisk/default-configs

mv/usr/src/voip/asterisk-13.8/etc/asterisk/*/usr/src/voip/asterisk-13.8/usr/share/asterisk/default-configs/

/usr/src/voip/asterisk-13.8/etc/logrotate.d/asterisk

chown -R root:asterisk /usr/src/voip/asterisk-13.8/etc/asterisk/

chmod 750 /usr/src/voip/asterisk-13.8/etc/asterisk/

3.1.6 Installing Scripts

echo "Package: asterisk

Priority: optional

Section: base

Maintainer: Support IPBrick <support@ipbrick.com>

Architecture: all

Depends: libspeex1, libpri1.4, dahdi

Replaces: asterisk-modules (<= 1:13.8~dfsg-3), asterisk-voicemail (<= 1:13.8~dfsg-3)

Provides: asterisk-voicemail

Conflicts: asterisk-voicemail

Version: 1:13.8

Description: Asterisk- certified-13.8.

" > /usr/src/voip/asterisk-13.8/DEBIAN/control

```
echo '#!/bin/bash
groupadd -g 116 asterisk
useradd -u 108 -g 65534 -s /bin/false -c "Asterisk PBX" -d /var/lib/asterisk asterisk
exit 0
' > /usr/src/voip/asterisk-13.8/DEBIAN/preinst
chmod 755 /usr/src/voip/asterisk-13.8/DEBIAN/preinst

echo '#!/bin/bash
mknod /var/log/asterisk/cdr-csv/Master.csv c 1 3
chown -R asterisk.asterisk /var/log/asterisk
insserv asterisk
exit 0
' > /usr/src/voip/asterisk-13.8/DEBIAN/postinst
chmod 755 /usr/src/voip/asterisk-13.8/DEBIAN/postinst

echo '#!/bin/bash
insserv -d -r asterisk
exit 0
' > /usr/src/voip/asterisk-13.8/DEBIAN/postrm
chmod 755 /usr/src/voip/asterisk-13.8/DEBIAN/postrm

cd /usr/src/voip/
mkdir -p /usr/src/voip/asterisk-sounds_13.8/var/lib/asterisk/
mkdir -p /usr/src/voip/asterisk-sounds_13.8/DEBIAN/
mv/usr/src/voip/asterisk13.8/var/lib/asterisk/sounds/usr/src/voip/asterisksounds_13.8/var/lib
/asterisk/
mkdir -p /usr/src/voip/asterisk-moh_13.8/var/lib/asterisk/
```

mkdir -p /usr/src/voip/asterisk-moh_13.8/DEBIAN/

mv/usr/src/voip/asterisk13.8/var/lib/asterisk/moh/usr/src/voip/asteriskmoh_13.8/var/lib

/asterisk/

mkdir -p /usr/src/voip/asterisk-pickup4_13.8/usr/lib/asterisk/modules/

mkdir -p /usr/src/voip/asterisk-pickup4-13.8/DEBIAN/

mv/usr/src/voip/asterisk13.8/usr/lib/asterisk/modules/app_directed_pickup4.so
/usr/src/voip/asterisk-pickup4-13.8/usr/lib/asterisk/modules

3.1.7 Installing Debian Packages

dpkg -b asterisk-13.8 asterisk-13.8.deb

dpkg --contents asterisk-13.8.deb

dpkg --info asterisk-13.8.deb

dpkg -i asterisk-13.8.deb

To initiate asterisk server use the following commands

/etc/init.d/asterisk start|stop|restart

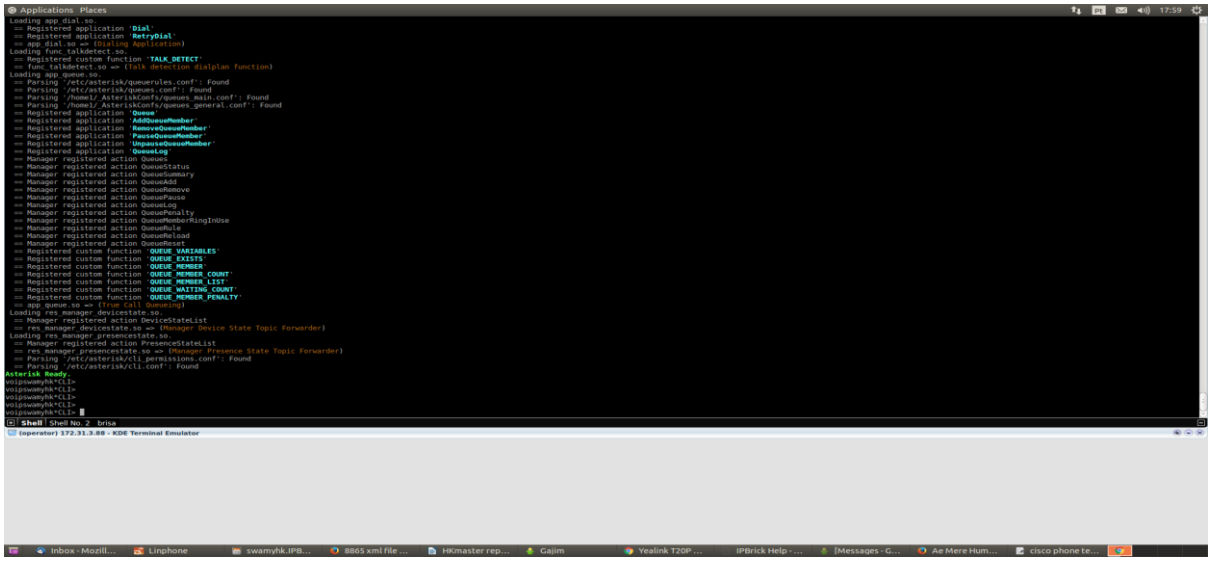


Figure 9 .Asterisk 13.8 successfully installed.

3.2 Asterisk13.8 Troubleshooting?

1.configure: error: *** termcap support not found (on modern systems, this typically means the ncurses development package is missing)

Soluiton: Run dpkg pacakge dpkg -i libncurses5-dev_5.9-10_amd64.deb

2.libncurses5-dev depends on libtinfo-dev (= 5.9-10); however:

Package libtinfo-dev is not installed.

dpkg: error processing libncurses5-dev (--install):

dependency problems - leaving unconfigured

Solution:Run dpkg package dpkg -i libtinfo-dev_5.9-10_amd64.deb

3.Configure: error: *** uuid support not found (this typically means the uuid development package is missing)

Solution: Run dpkg Package dpkg -i uuid-dev_2.20.1-5.3_amd64.deb

4.Configure: *** Please install the 'libxml2' development package

solution: Run dpkg Package dpkg -i libxml2_2.8.0+dfsg1-7+wheezy5_amd64.deb and dpkg -i libxml2-dev_2.8.0+dfsg1-7+wheezy5_amd64.deb

5.configure: error: *** Asterisk now uses SQLite3 for the internal Asterisk database.

Solution: Run dpkg package dpkg -i libsqlite3-dev_3.7.13-1+deb7u2_amd64.deb and dpkg -i libsqlite3-0_3.7.13-1+deb7u2_amd64.deb

6.Checking for the ability of -lsrtp to be linked in a shared object... no

configure: WARNING: ***

configure: WARNING: *** libsrtp could not be linked as a shared object.

configure: WARNING: *** Try compiling libsrtp manually. Configure libsrt

configure: WARNING: *** with ./configure CFLAGS=-fPIC --prefix=/usr

configure: WARNING: *** replacing /usr with the prefix of your choice.

configure: WARNING: *** After re-installing libsrtp

configure: WARNING: *** configure script.

configure: WARNING: ***

configure: WARNING: *** If you do not need SRTP support re-run configure

configure: WARNING: *** with the --without-srtp option.

Solution: Run dpkg package dpkg –I libsrtp0-dev_1.4.4+20100615~dfsg-2+deb7u1_amd64.deb

7.dpkg: dependency problems prevent configuration of subversion:

subversion depends on libsvn1 (= 1.6.17dfsg-4+deb7u10); however:

Version of libsvn1:amd64 on system is 1.6.17dfsg-4+deb7u6.

Solution:Run dpkg package dpkg -i libsvn1.1.6.17dfsg-4+deb7u10

3.3 Test the call connectivity between two endpoints

Two Phones are registered in asterisk console and acquiring ip address and port address as shown in figure 8**.**

Commands used: ipbrick:~# asterisk -rvvv

ipbrick*CLI> sip reload

ipbrick*CLI> sip show peers

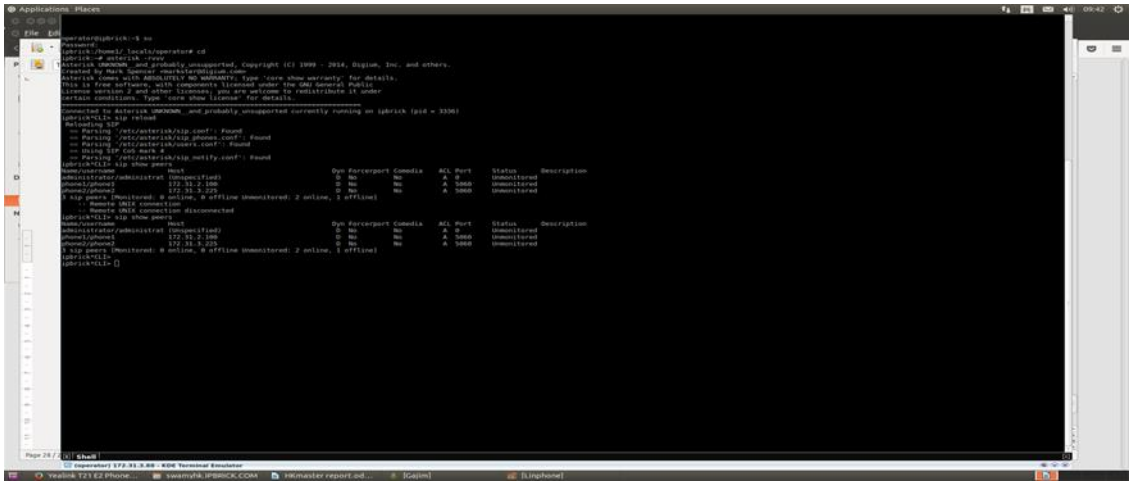


Figure 10.Phones registered in asterisk cli

3.3.1 Phones are created in IPBRICK web interface

IPBRICK is updated with newest version and update with fix-update4 and version 6.2.

In (Figure 11 and 12) it is possible to see the registered IPBrick VoIP clients (IP phones, workstations + softphone). In section Machine Management you find the description of the menu to insert the VoIP machines.

It is also possible to register phones in:

Advanced Configurations - Telephony - Registered Phones

This option is valid, if it isn't necessary to attribute a specific IP address to the phone. It is possible to add a phone just by filling the field relating the name and the access password. This assuming that DNS is working correctly**.**

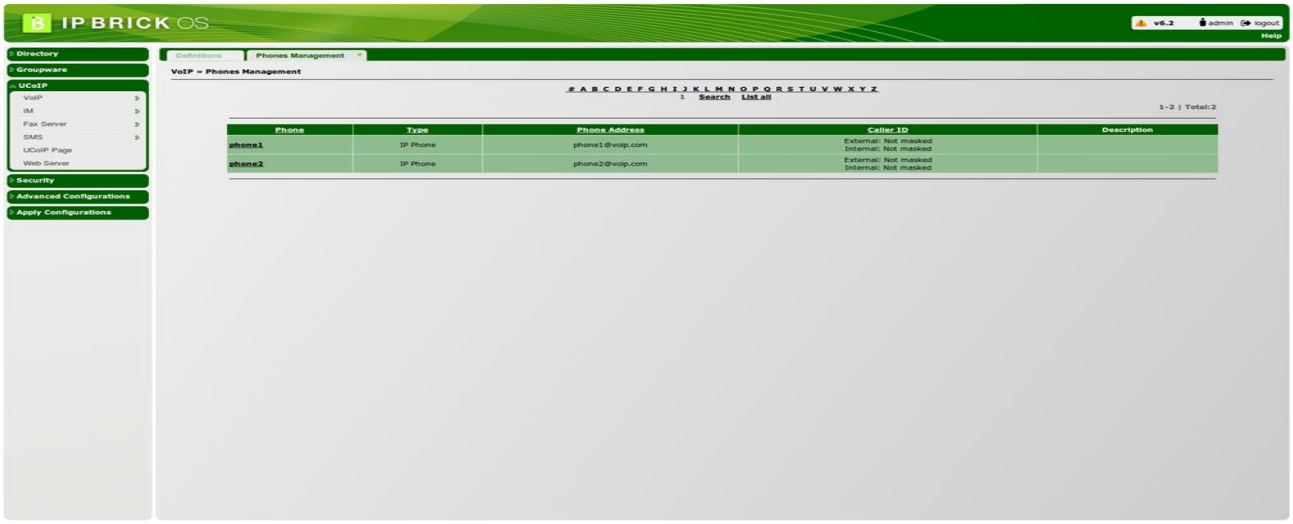


Figure 11.IPbrick web interface

3.3.2 Phone1 Ipbrick webinterface configuration

Phone1 has been created successfully with filling the necessary fields in phone management in ipbrick web interface as shown in below figure 12.

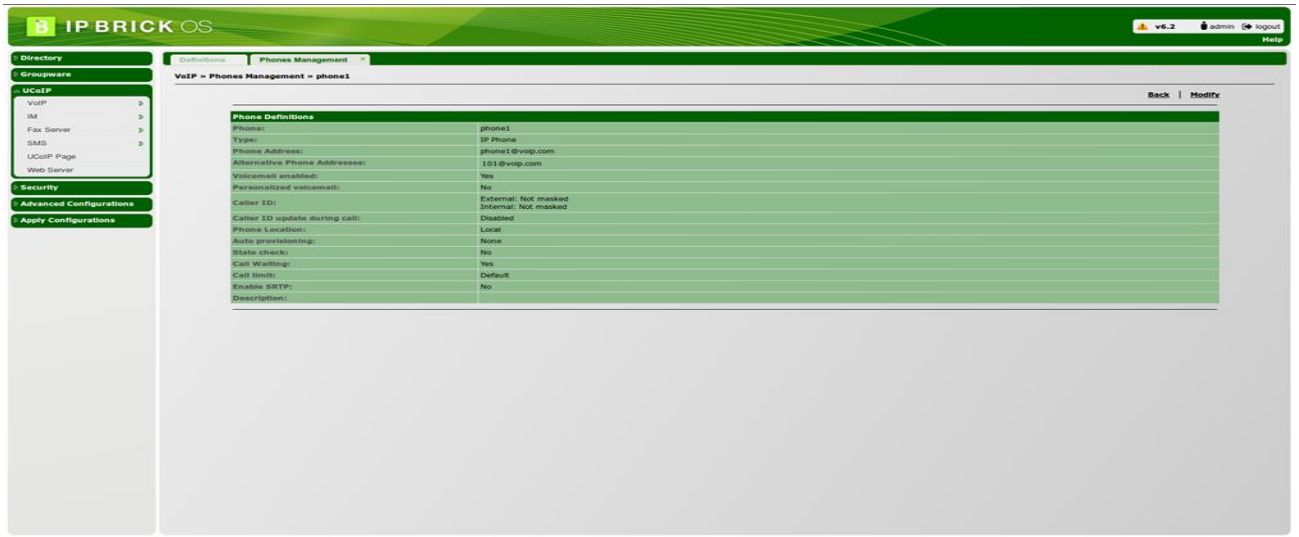


Figure 12.Phone1 configuration

3.3.3 Phone2 Ipbrick web interface configuartions

Phone2 has been created successfully with filling the necessary fields in phone management in ipbrick web interface as shown in below figure 13.

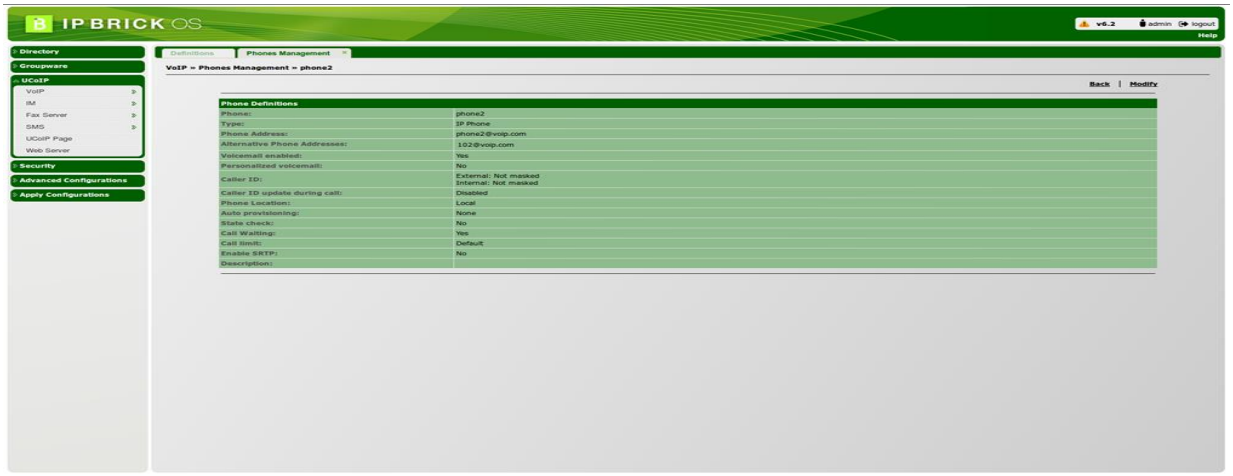


Figure13.Phone2 Configuration

3.4 IPBrick phone configuration Template(Extensions.conf)

The configuration file "extensions.conf" contains the "dial plan" of Asterisk, the master plan of control or execution flow for all of its operations. It controls how incoming and outgoing calls are handled and routed. This is where you configure the behavior of all connections through your PBX.

The content of "extensions.conf" is organized in sections, which can be either for static settings and definitions, or for executable dialplan components in which case they are referred to as contexts. The settings sections are general and globals and the names of contexts are entirely defined by the system administrator. A special type of contexts are macros, label by a userdefined name prefixed with macro-. These are reusable execution patterns, like procedures in a programming language. Every section in extensions.conf starts with the name of the section contained within square brackets.

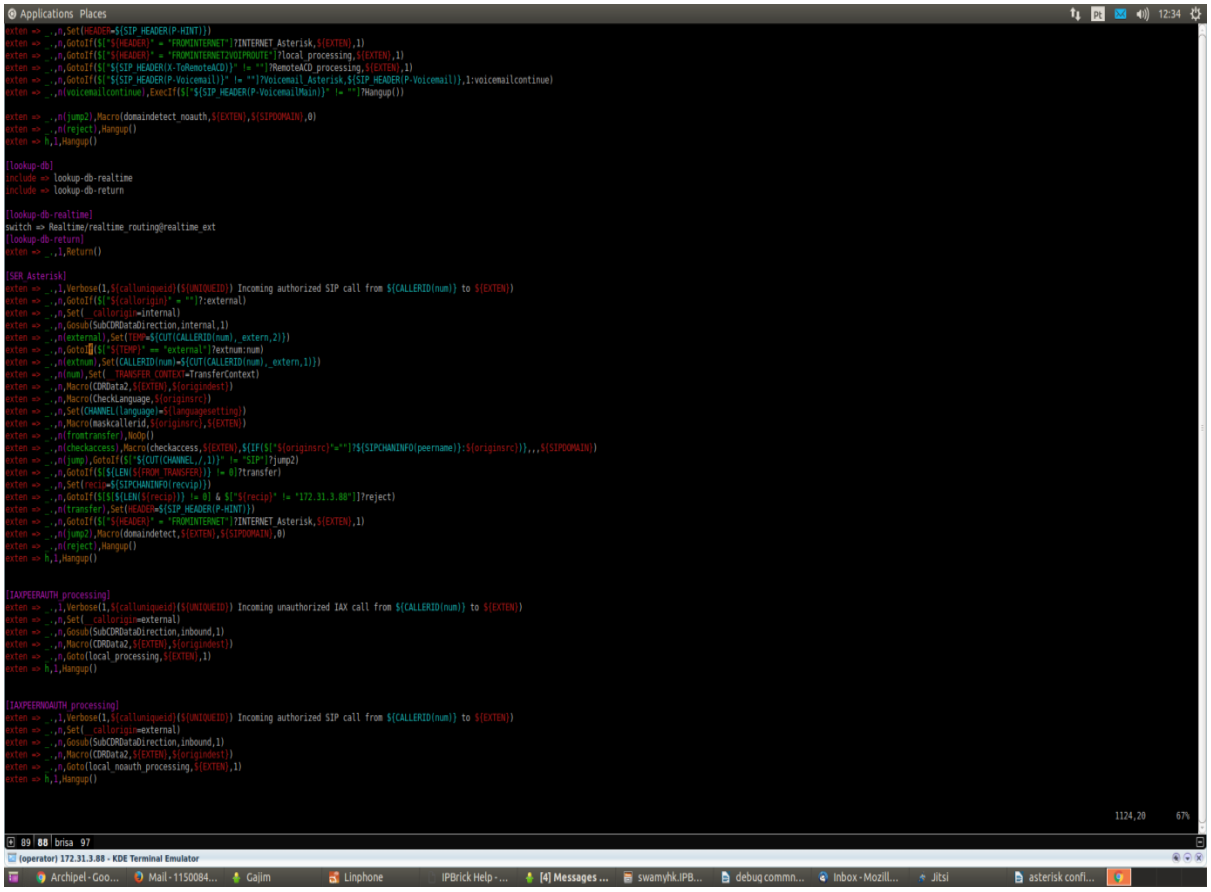


Figure 14.Extensions configurations

3.5 IPBrick Phone autoprovisioning Template (Sip_phones.conf)

This is the ipbrick auto provisioning template created automatically when something has been modified in web interface.

Figure15.Ipbrick Phone configuration

3.5.1 Phone1 is registerd through Linphone softphone (Account)

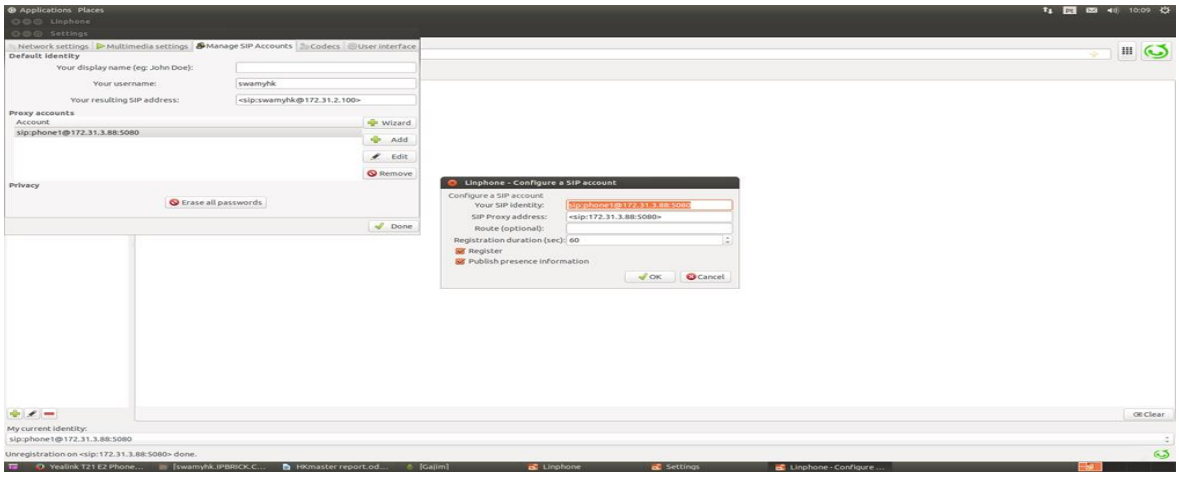


Figure16.Phone1 is registered in softphone

3.5.2 Phone2 Is registerd with yealink hardphone(Account)

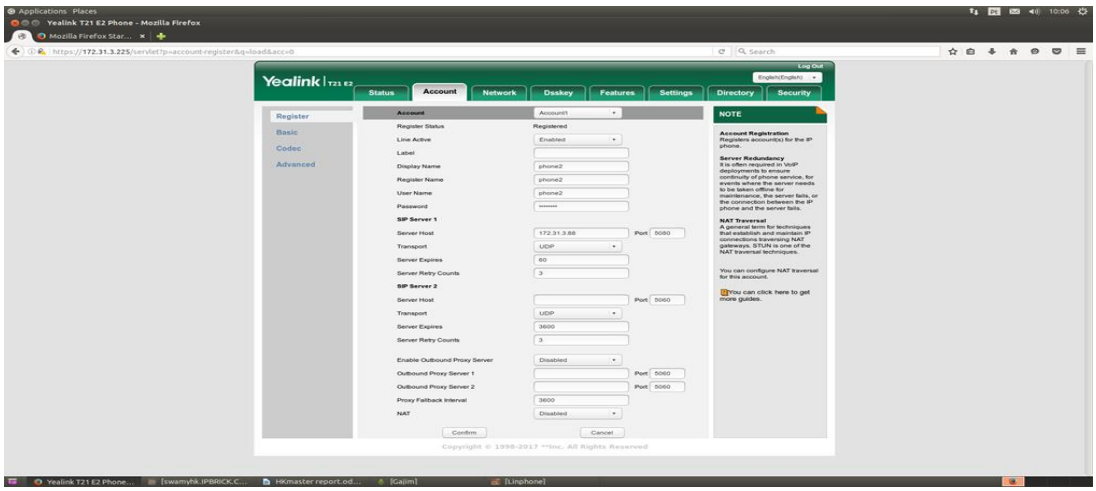


Figure17.Phone2 is registered in hardphone

3.6 SIP call running in asterisk CLI

Below figures show once we make a call to other endpoint ,call has been placed into the asterisk pbx and sip call is running through ipbrick phone template(context=SER_Asterisk).

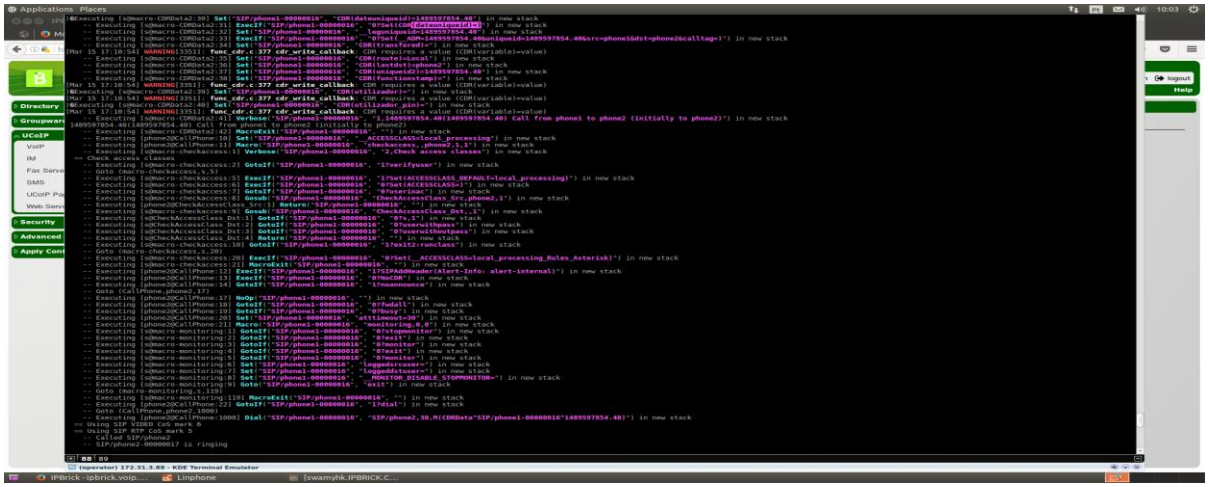


Figure 18.Call placed in asterisk cli

3.6.1 Call setup (initiation)

Call has been connected between source phone1 to destination phone2 and call state shows call setup.

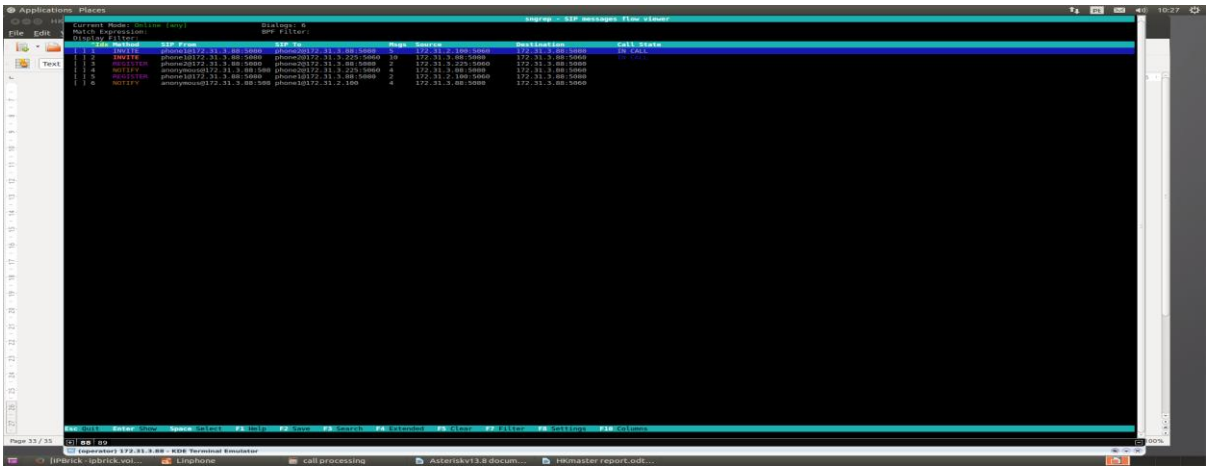


Figure19. sip call setup

3.6.2 Call has been established

Call has been established between two phones and call state shows INCALL means calls connected.

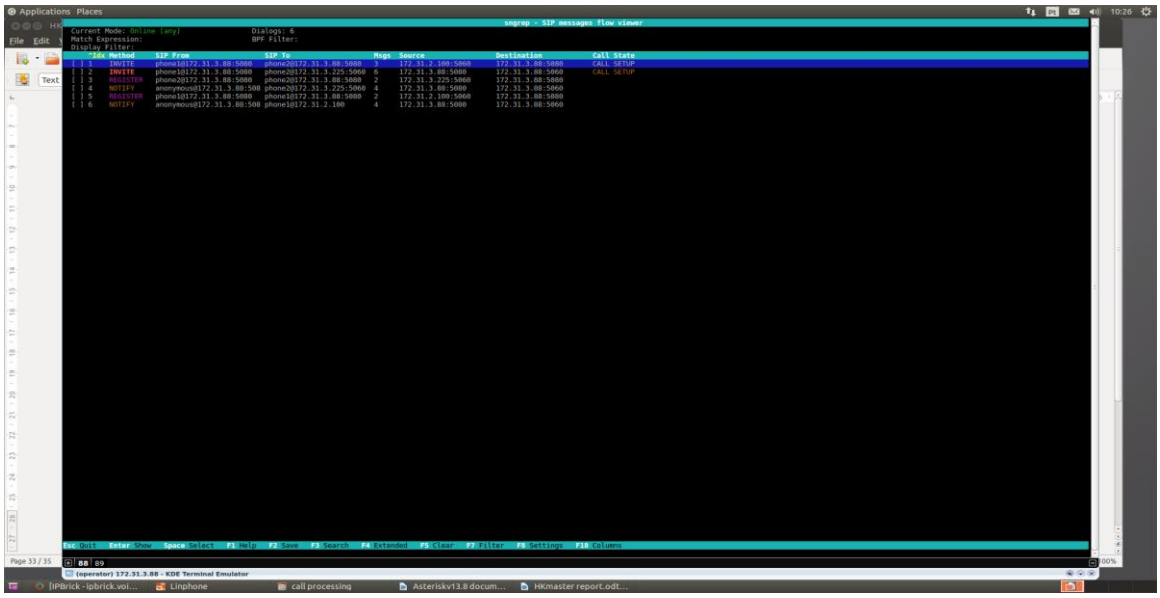


Figure 20.sip call Established

3.6.3 Call completed .

Call has been completed when one of the parties disconnects the call and call state shows completed.

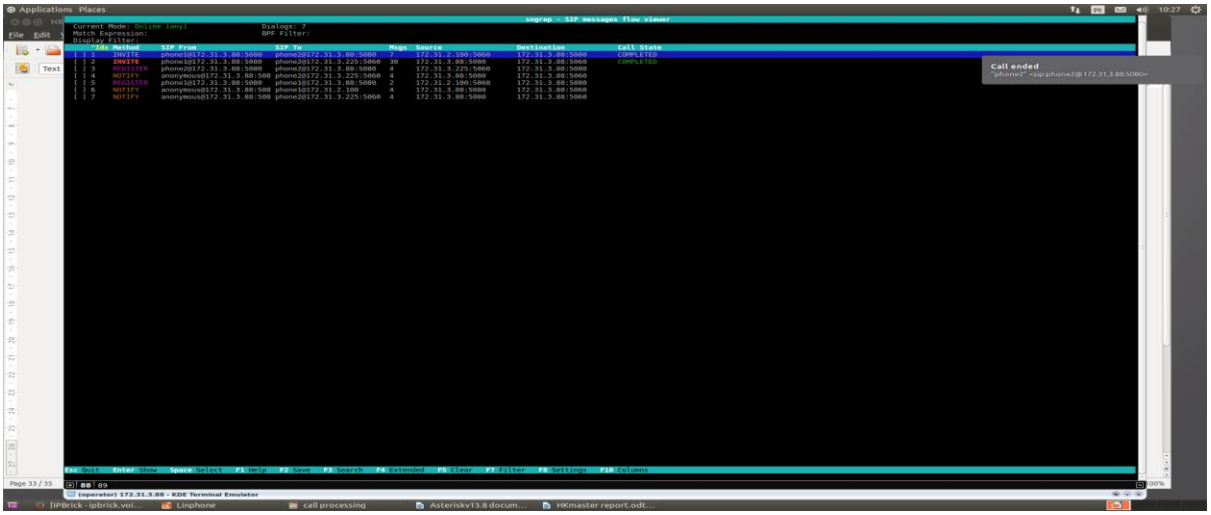


Figure 20.sip call Established

3.7 Asterisk Configuration for webrtc

To enable WebRTC support in Asterisk 13.8 (in a Linux environment), according to Asterisk documentation [53], one needs to do the following.

• Build and install SRTP; download and configure SRTP and its development headers.

3.7.1 Asterisk http configuration file

• Configure WebSocket module; enable the WebSocket module in the configuration file http.conf and ensure it is running on port 8088.

The configuration sample file is by default located at /etc/asterisk/http.conf

A very basic configuration of http.conf could be as follows:

[general]

enabled=yes

bindaddr=0.0.0.0

bindport=8088

tlsenable=yes

tlscertfile=/path Location/ *pem files

tlsprivatekey=/path Location/ *pem files

Below figure shows the configuration files of http.conf.

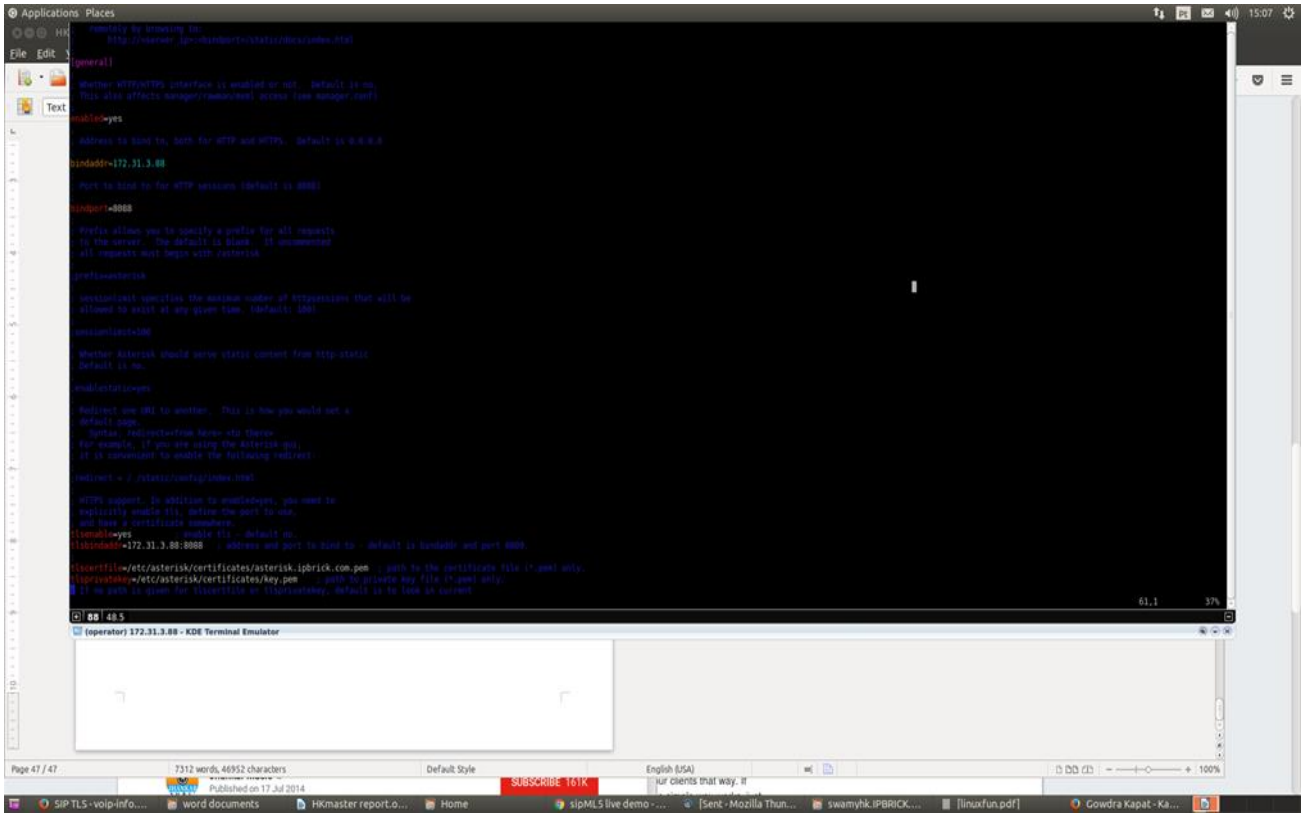


Figure 22.http configuration file

3.7.2 http and https server enabled on asterisk console.

1. Uncomment the line "enabled=yes" in /etc/asterisk/http.conf to enable Asterisk's built in micro HTTP server.

2. If you want Asterisk to actually deliver simple HTML pages, CSS, javascript, etc. you should uncomment "enablestatic=yes"

3. Adjust your "bindaddr" and "bindport" settings as appropriate for your desired accessibility.

4. Adjust your "prefix" if appropriate, which must be the beginning of any URI on the server to match. The default is blank, that is no prefix and the rest of these instructions assume that value.

Figure 23.http and https enabled.

3.7.3 Asterisk SIP configuration file

Configuration file for Asterisk SIP channels, for both inbound and outbound calls.

Starting with Asterisk: The global option "port" in 1.0.X that is used to set which port to bind to has been changed to "bindport" to be more consistent with the other channel drivers and to avoid confusion with the "port" option for users/peers.

Starting with Asterisk: The previously deprecated options "insecure=very" and "insecure=yes" have now been removed. "insecure=invite,port" is the equivalent of "insecure=very".

• Configure chan_sip module

– Add WebSocket as a transport layer the sip.conf configuration file

– Enable RTP Audio-Visual Profile with Feedback (AVPF) in the sip.conf configuration file

– Enable encryption in the sip.conf configuration file to support media encryption.

Figure 24.Sip Configuration

*Configure sip.conf

[general]

useragent=IPBrick

sdpsession=IPBrick

callerid=anonymous

shrinkcallerid=no

nat=route

realm=172.31.3.88

transport=udp,ws,wss

websocket_enabled = true

autodomain=yes

```
domain=voip.com
domain=ipbrick.voip.com
domain=webphone.voip.com
domain=172.31.3.88
t38pt_udptl = yes
Default context for unauthenticated calls not coming through SER
context=NO_CONTEXT
bindport = 5060      ; Port to bind to (SIP is 5060)
udpbindaddr = 172.31.3.88 ; Address to bind to (all addresses on machine)
srvlookup=yes
disallow=all
allow=gsm
allow=ulaw
allow=alaw
allow=h264
allow=h263p
allow=h263
allow=h261
allow=vp8
videosupport=yes
rtptimeout=600
rtpholdtimeout=700
allowsubscribe=yes
notifyringing=yes
notifyhold=no
notifycid=no
useclientcode=yes
```

limitonpeers=yes

rtcachefriends=yes

defaultexpirey=3600

registerattempts=0

maxexpirey=3600

qualifyfreq=60 ; State check frequency (Default: 60)

dtmfmode=auto

sendrpid=no

trustrpid=no

outboundproxy=172.31.3.88:5060

3.7.4 Asterisk rtp configuration file.

The rtp.conf file controls the Real-time Transport Protocol (RTP) ports that Asterisk uses to generate and receive RTP traffic. The RTP protocol is used by SIP, H.323, MGCP, and possibly other protocols to carry media between endpoints [13].

The default rtp.conf file uses the RTP port range of 10,000 through 20,000. However, this is far more ports than you're likely to need, and many network administrators may not be comfortable opening up such a large range in their firewalls. You can limit the RTP port range by changing the upper and lower bound limits within the rtp.conf file.

For every bidirectional SIP call between two endpoints, five ports are generally used: port 5060 for SIP signaling, one port for the data stream and one port for the Real-Time Control Protocol (RTCP) in one direction, and an additional two ports for the data stream and RTCP in the opposite direction.

UDP datagrams contain a 16-bit field for a Cyclic Redundancy Check (CRC), which is used to verify the integrity of the datagram header and its data. It uses polynomial division to create the 16-bit checksum from the 64-bit header. This value is then placed into the 16-bit CRC field of the datagram, which the remote end can then use to verify the integrity of the received datagram. Setting rtpchecksums=no requests that the OS not do UDP checksum creating/checking for the sockets used by RTP. If you add this option to the sample rtp.conf file, it will look like this:

*Configure rtp.conf

[general]

rtpstart=10000

rtpend=20000

icesupport=yes

stunaddr=stun.l.google.com:19302

Figure 25.rtp configuration

3.8 Generate Let's Encrypts certs for IPBrick

To enable HTTPS on your website, you need to get a certificate (a type of file) from a Certificate Authority (CA). Let’s Encrypt is a CA. In order to get a certificate for your website’s domain from Let’s Encrypt, you have to demonstrate control over the domain. With Let’s Encrypt, you do this using software that uses the ACME protocol, which typically runs on your web host.

Certbot is an easy-to-use automatic client that fetches and deploys SSL/TLS certificates for your webserver and automatically enable HTTPS on your website with EFF's Certbot, Certbot was developed by EFF and others as a client for Let’s Encrypt and was previously known as “the official Let’s Encrypt client” or “the Let’s Encrypt Python client.” Certbot will also work with any other CAs that support the ACME protocol[14].

While there are many other clients that implement the ACME protocol to fetch certificates, Certbot is the most extensive client and can automatically configure your webserver to start serving over HTTPS immediately. for Apache, it can also optionally automate security tasks such as tuning ciphersuites and enabling important security features such as HTTP → HTTPS redirects, OCSP stapling, HSTS, and upgrade-insecure-requests.

Certbot is part of EFF's larger effort to encrypt the entire Internet. Websites need to use HTTPS to secure the web. Along with HTTPS Everywhere, Certbot aims to build a network that is more structurally private, safe, and protected against censorship[6].

To generate a certificate you must do the following:

1. Install certbot-auto (eg. on /home1/_locals/operator)

    wget https://dl.eff.org/certbot-auto

    chmod a+x certbot-auto

2.SSLCertificateFile

This certificate file produced and normally reside inside the below folders.

/etc/letsencrypt/live/ucoip.domain.com/cert.pem

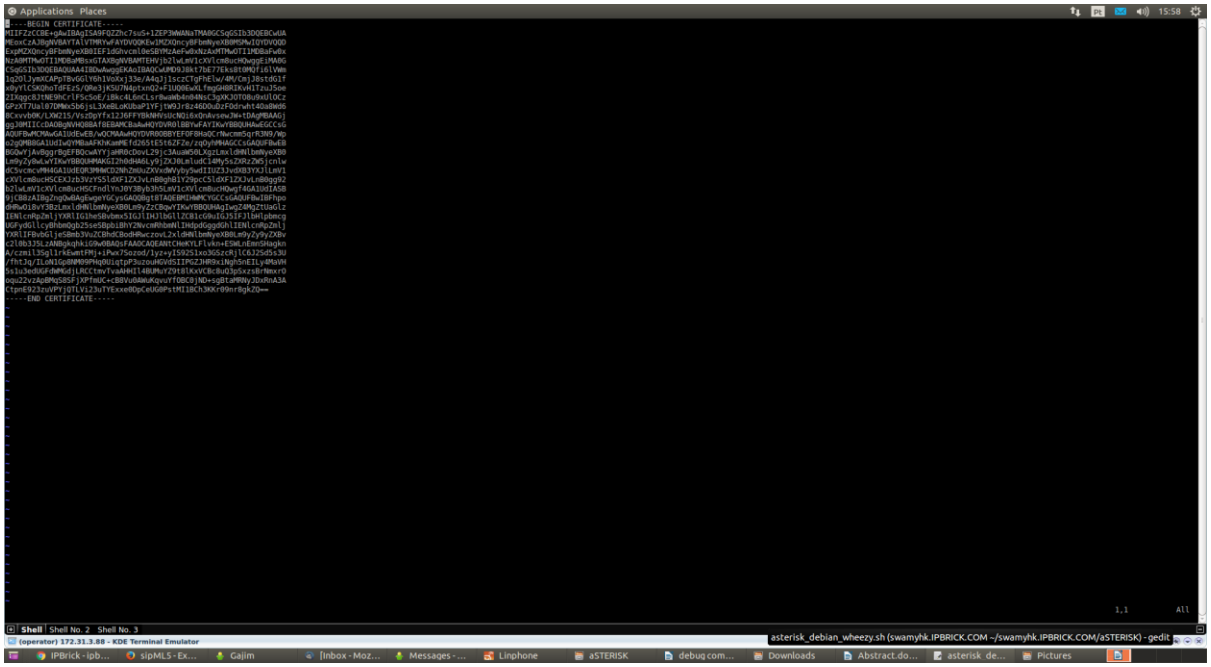


Figure 26. SSLCertificate file

3.SSLPrivateKeyFile

/etc/letsencrypt/live/ucoip.domain.com/privkey.pem

Figure 27.SSL Private key file

3.9 Configurations of SIPML5 webrtc web application?

This is the world's first open source (BSD license) HTML5 SIP client entirely written in javascript for integration in social networks (FaceBook, Twitter, Google+), online games, e-commerce websites, email signatures... No extension, plugin or gateway is needed. The media stack rely on WebRTC[15].

The client can be used to connect to any SIP or IMS network from your preferred browser to make and receive audio/video calls and instant messages.

The list of supported features

Works on Chrome, Firefox, IE, Safari, Opera and Bowser

Audio / Video call

Screen/Desktop sharing from Chrome to any SIP client

Instant messaging

Presence

Call Hold / Resume

Explicit Call transfer

Multi-line and multi-account

Dual-tone multi-frequency signaling (DTMF) using SIP INFO

Click-to-Call

SIP TelePresence (Video Group chat)

3GPP IMS standards

Now,head on to your Google Chrome browser and type
https://www.doubango.org/sipml5/ and click live demo

If you're not NATting, then just put two [] like that and the ICE/STUN will not be used to manage RTP and you call will be connected faster as well. Be sure the stun you use on your server side is the same used on SIPML5 as well.

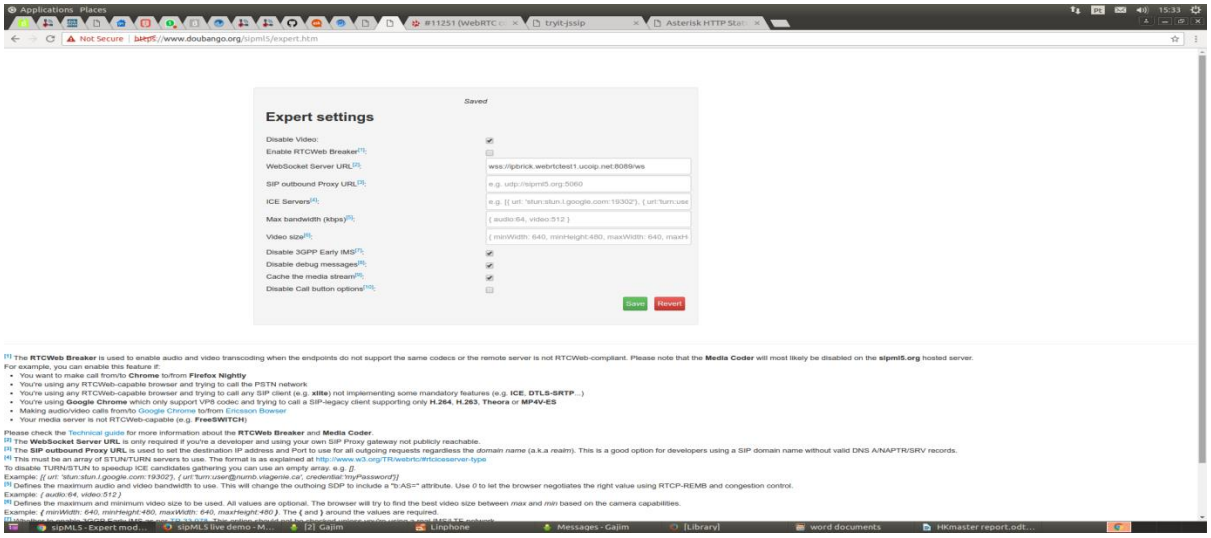


Figure 28. Sipml5 expert settings tab

[1]The RTCWeb Breaker is used to enable audio and video transcoding when the endpoints do not support the same codecs or the remote server is not RTCWeb-compliant. Please note that the Media Coder will most likely be disabled on the sipml5.org hosted server.

For example, you can enable this feature if:

- You want to make call from/to Chrome to/from Firefox Nightly.
- You're using any RTCWeb-capable browser and trying to call the PSTN network.

You're using any RTCWeb-capable browser and trying to call any SIP client (e.g. xlite) not implementing some mandatory features (e.g. ICE, DTLS-SRTP...)

You're using Google Chrome which only support VP8 codec and trying to call a SIP-legacy client supporting only H.264, H.263, Theora or MP4V-ES

Making audio/video calls from/to Google Chrome to/from Ericsson Bowser

Your media server is not RTCWeb-capable (e.g. FreeSWITCH)

Please check the Technical guide for more information about the RTCWeb Breaker and Media Coder.

[2] The WebSocket Server URL is only required if you're a developer and using your own SIP Proxy gateway not publicly reachable.

[3] The SIP outbound Proxy URL is used to set the destination IP address and Port to use for all outgoing requests regardless the domain name (a.k.a realm). This is a good option for developers using a SIP domain name without valid DNS A/NAPTR/SRV records.

[4] This must be an array of STUN/TURN servers to use. The format is as explained at http://www.w3.org/TR/webrtc/#rtciceserver-type.

To disable TURN/STUN to speedup ICE candidates gathering you can use an empty array. Example:[{url:'stun:stun.l.google.com:19302'}

[5] Defines the maximum audio and video bandwidth to use. This will change the outhoing SDP to include a "b:AS=" attribute. Use 0 to let the browser negotiates the right value using RTCP-REMB and congestion control.

Example: { audio:64, video:512 }

[6] Defines the maximum and minimum video size to be used. All values are optional. The browser will try to find the best video size between max and min based on the camera capabilities.

Example: { minWidth: 640, minHeight:480, maxWidth: 640, maxHeight:480 }. The { and } around the values are required.

[7] Whether to enable 3GGP Early IMS as per TR 33.978. This option should not be checked unless you're using a real IMS/LTE network.

If earlyIMS is disabled then, authentication will be done as per 3GPP TS 24.229 - 5.1.1.2.2.

[8] Whether to disable debug messages. SIPML5 supports #4 debug levels: INFO, WARN, ERROR and FATAL. Default level is INFO. Check this option to set the level value to ERROR.

[9] Whether to reuse the same media stream for all calls. If your website is not using https then, the browser will request access to the camera (or microphone) every time you try to make a call. Caching the media stream will avoid getting these notifications for each call.

[10] Whether to add options (Audio, Video, Screen share) in the the call button.

Go back to the other tab which the webphone is on, enter the SIP extension detailed you created above, or follow and modify per example below

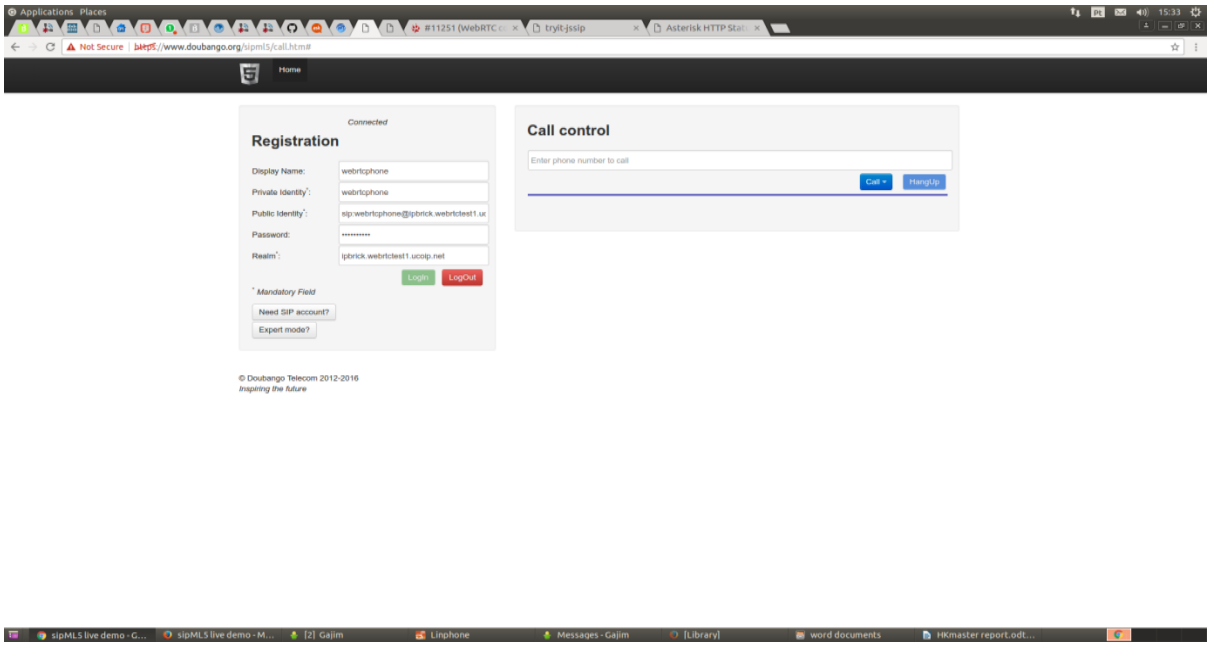


Figure 29.Sipml5 registration tab

Do the same above steps and configure your other web sip extension say webrtcphone on other browser. Register both extensions on your asterisk by simply hitting the ‘Login’ button on your screen.

You should be connected to your asterisk server if you have followed above steps.Now you should be able to make call between your test extensions.

3.10 Testing Call Between Web application and asterisk?

Call is initiated from browser side webclient sipml5 and directed through asterisk via websockets and asterisk redirects linephone softphone.

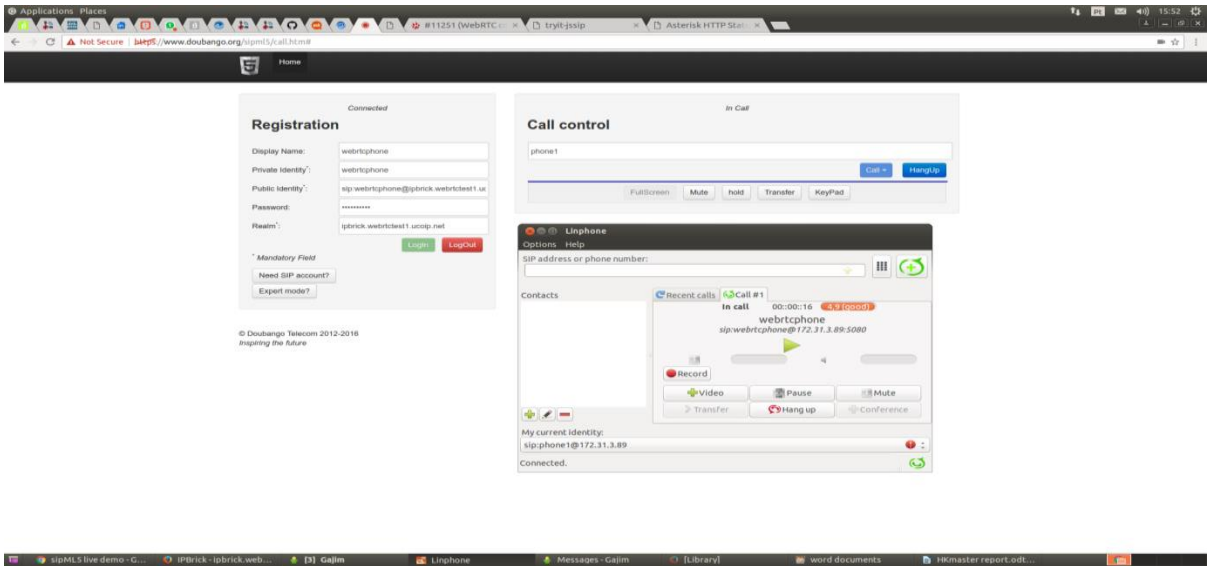


Figure 30.call flow between sipml5 webphone and asterisk server

CHAPTER 4

CONCLUSION

Voip is one of the emerging technologies used for communication where one can communicate with whoever they intend to in the world by using appropriate voip service provider. Due to advantages,features and number of voip providers voip has become less expensive.So the main concern about this technology is compatibility. The main objective of this project is voip subsystem compatibility in which updates and patches are made from ipbrick previous version v6.1 to ipbrick current verisonv6.2.Additional dependencies are added to implement asterisk13.Ipbrick previous version V6.1 uses asterisk 1.8 as its pbx server but it does not support webrtc technology where webrtc works only with websockets called WS and WSS.. The main advantage of WebRTC is its browser support accessible from JavaScript which means that you can provide a web client to end users without the need to download/install any additional software or browser plugin.

Asterisk 13 and above versions supports websockets, so we made changes in voip subsystem by installing new asterisk server with version 13.8 and tested all the voip features by using Kamailio sip proxy server and webrtc technology. tested successfully between asterisk and webrtc using sipml5 and sip legacy endpoint.

Finally we analysed and tested all the current voip system features which are compatible and achieved successfully as shown in graphical obtained results.

4.1 Future Works

Asterisk v13 provides two channel sip drivers now i have tested webrtc with asterisk channel driver called chan_sip. I will test webrtc with another channel sip driver called Pjsip and made a connection between webrtc and kamailio near future.

REFERENCES

1. https://en.wikipedia.org/wiki/ Voice_over_IP (online visited 5/01/2017)

2. http://www.ipbrick.com/ipbrick-gt/ (online visited 11/01/2017)

3. www.ipbrick.com/wp-content/uploads/2016/05/v6.2_release-notes.pdf (online visited 27/01/2017)

4. https://webrtc.org/ (online visited 10/02/2017)

5. http://www.asterisk.org/ (online visited 23/02/2017)

6. https://en.wikipedia.org/wiki/Session_Initiation_Protocol (online visited 01/03/2017)

7. https://www.kamailio.org/w/ (online visited 17/03/2017)

8. https://en.wikipedia.org/wiki/WebSocket (online visited 30/03/2017)

9. www.whoer.net/ (online visited 05/04/2017)

10. https://www.resiprocate.org/WebRTC_and_SIP_Over_WebSockets (online visited 14/04/2017)

11. https://en.wikipedia.org/wiki/Asterisk_(PBX) (online visited 21/04/2017)

12. http://www.ipcomms.net/sample-device-configurations/41-asterisk/181-install-asterisk-13-on-ubuntu-debian(online visited 03/05/2017)

13. https://www.voip-info.org/wiki/view/Asterisk+config+rtp.conf (online visited 17/05/2017)

14. https://letsencrypt.org/ (online visited 26/05/2017)

15. https://www.doubango.org/sipml5/ (online visited 04/06/2017)