

Encuestas por voz sobre IP - Aplicaciones

TCP/IP

Se desarrolla e implementa trabajo final para la cátedra de Aplicaciones TCP/IP a fin de llevar adelante una solución con los conceptos abordados en ella.

Introducción

Se trata del desarrollo e implementación de un sistema de *encuestas por voz, adquisición, procesamiento y visualización de los datos* correspondientes a los resultados de las encuestas en cuestión.

Sobre este documento

“**Encuestas por voz sobre IP**” se produce en el contexto de la cátedra Aplicaciones TCP/IP, de la carrera de Ing. en Telecomunicaciones, en la Universidad Nacional de Río Cuarto por Bibiana Rivadeneira.

Se encuentra bajo la licencia:



Este es un resumen legible por humanos (y no un sustituto) de la licencia. Advertencia.

Usted es libre de:

- **Compartir** – copiar y redistribuir el material en cualquier medio o formato
- **Adaptar** – remezclar, transformar y construir a partir del material para cualquier propósito, incluso comercialmente.
- La licenciente no puede revocar estas libertades en tanto usted siga los términos de la licencia

Bajo los siguientes términos:

- **Atribución** – Usted debe dar crédito de manera adecuada, brindar un enlace a la licencia, e indicar si se han realizado cambios. Puede hacerlo en cualquier forma razonable, pero no de forma tal que sugiera

que usted o su uso tienen el apoyo de la licenciante.

- No hay restricciones adicionales – No puede aplicar términos legales ni medidas tecnológicas que restrinjan legalmente a otras a hacer cualquier uso permitido por la licencia.

ÍNDICE DE CONTENIDOS

- Introducción.
- Sobre este documento.
- Objetivos.
- Resumen.
- Palabras clave.
 - Diagrama.
 - Tecnologías.
- Descripción.
 - Servidor.
 - Cliente.
 - Lógica.
- Diseño.
 - Capa 1: Docker.
 - Capa 2: Asterisk.
 - Capa 3: IVR, MiscApp.
 - Capa 4: Procesamiento y visualización.
- Procedimiento.
 - Requisitos previos.
 - Docker.
 - Git.
 - Asterisk + FreePBX + Módulos IVR.
 - Descargar y levantar la imagen.
 - Configuración básica en FreePBX.
 - Agregar extensiones y test básico.
 - Breve descripción de PJSIP.
 - Zoiper en Android.
 - Zoiper en Debian.
 - Misc Apps.
 - Importación de announcements.
 - Crear y configurar una aplicación IVR.
 - IVRs y announcements.
 - 1. Bienvenida.
 - 2. Género.
 - 3. Edad.
 - 4. Sabores.
 - Misc App.

- Adquisición de datos.
- Procesamiento de datos.
 - Logfile.
 - API.
- Visualización.
- Resultados.
 - API.
 - Visualización.
- Conclusión.
- Proyección.
- Referencias.

Objetivos

- Aplicar y reforzar conceptos de la pila de protocolos TCP/IP a una solución tecnológica.
- Desarrollar e implementar un sistema de encuestas por voz que permita adquirir, procesar y visualizar datos referentes a dichas encuestas.
- Emplear tecnologías de código abierto.

Resumen

Se trata de un sistema de **encuestas por voz**, más específicamente a través de llamadas telefónicas cursadas sobre el **protocolo de internet**, adquisición de los datos registrados durante las encuestas en cuestión, (DTMF) **procesamiento y visualización de los mismos en un portal web**.

Palabras clave

debian, docker, asterisk, freepbx, ivr, miscapp, sip, rtp, udp, python, flask, api, dash, wireshark

Diagrama

El sistema se representa mediante el siguiente diagrama simbólico:

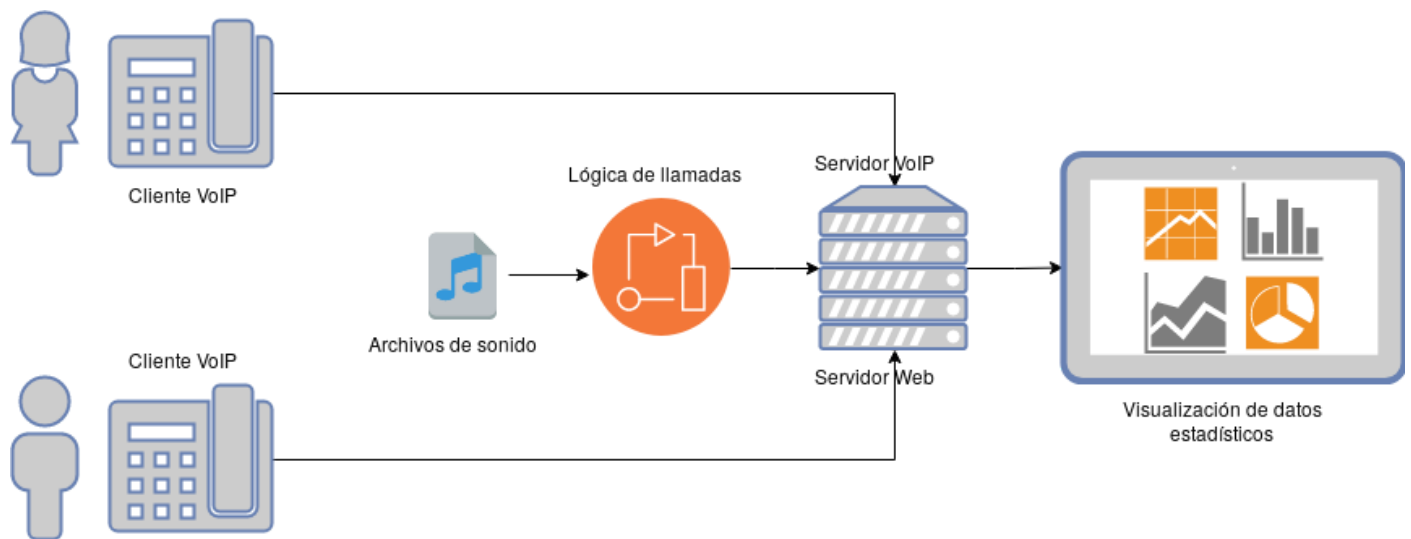


Diagrama simbólico sistema de encuestas por VoIP

- **Cliente:** Terminal telefónica para cursar la llamada de voz sobre IP y comenzar una interacción hombre-máquina.
- **Archivos de sonido:** Grabaciones predefinidas para los mensajes con los que interactúa el cliente.
- **Lógica de llamadas:** Los archivos de sonido se reproducen siguiendo una lógica de decisiones simples en función de las respuestas del cliente.
- **Servidor VoIP:** Cursa las llamadas por VoIP y realiza la adquisición de los datos resultados de las encuestas.
- **Servidor Web:** En el que se procesan los datos de los resultados y se generan gráficos amigables con estadísticas de los mismos, a visualizar en un portal web.

Tecnologías

Tecnologías a implementar en el sistema:

- **Docker client e imagen:** Contenedor con distribución [Debian](#), [Asterisk](#), [FreePBX](#), módulos [IVR](#).
- **Asterisk 15:** Distribución open source para soluciones VoIP y lógica por voz.
- **FreePBX 14:** Interfaz gráfica basada en web (open source) para administrar Asterisk.
- **Módulos IVR:** (*Interactive Voice Response*)
- **MiscApplications:** Módulo requerido para enrutar las llamadas a la encuesta.
- **Zoiper:** Cliente de VoIP disponible para GNU/Linux y Android, entre otros.
- **Flask API:** Framework open source escrito en Python para crear APIs.
- **Dash:** Biblioteca de Python para visualización amigable e interactiva de datos a partir de json.

Descripción

Se instala, configura e implementa un sistema de encuestas por voz sobre el protocolo de internet ([IVR](#) en

VoIP). Se describen a continuación las diferentes partes que componen el sistema, respondiendo al diagrama antes expuesto.

Servidor

Una distribución GNU/Linux [Debian](#) hostea al servidor [Asterisk versión 15](#) de VoIP. Se emplea [FreePBX](#) (*Interfáz gráfica web para administración de Asterisk*), versión 14.

Se emplean módulos [IVR](#) para brindar la posibilidad de interacción con mensajes automáticos de voz y reportes de respuestas de los usuarios, más específicamente de la señalización, de los tonos DTMF.

Cliente

Lógica

A partir de un conjunto de archivos de audio, se reproducen ante los siguientes eventos:

- **Llamada entrante:** Mensaje de bienvenida
- **Registrar una encuesta:** Mensaje de encuesta y las opciones correspondientes a la respuesta.
 - **Opción para registrar encuesta**
 - **Opción para repetir mensaje**
 - **Opción para salir**
- **Respuesta:** Mensaje de respuesta registrada, Mensaje de respuesta no registrada, mensaje de salida.

Diseño

El diseño del sistema puede ser descrito en “capas” correspondientes a los módulos que componen el sistema de encuestas por voz sobre el protocolo de internet.

Se dockerizan las tecnologías a implementar sobre una distribución GNU/Linux.

Capa 1: Docker

Cliente docker, corriendo sobre un Debian 10, capa de abstracción para desplegar en contenedores las herramientas requeridas para el servidor de VoIP.

Capa 2: Asterisk

Servidor de VoIP que proporciona funcionalidades de una central telefónica (PBX) para llamadas de voz sobre el protocolo de internet.

Capa 3: IVR, MiscApp

Módulos de respuestas de voz interactiva y para desvío de llamada de extensión a la encuesta.

Capa 4: Procesamiento y visualización

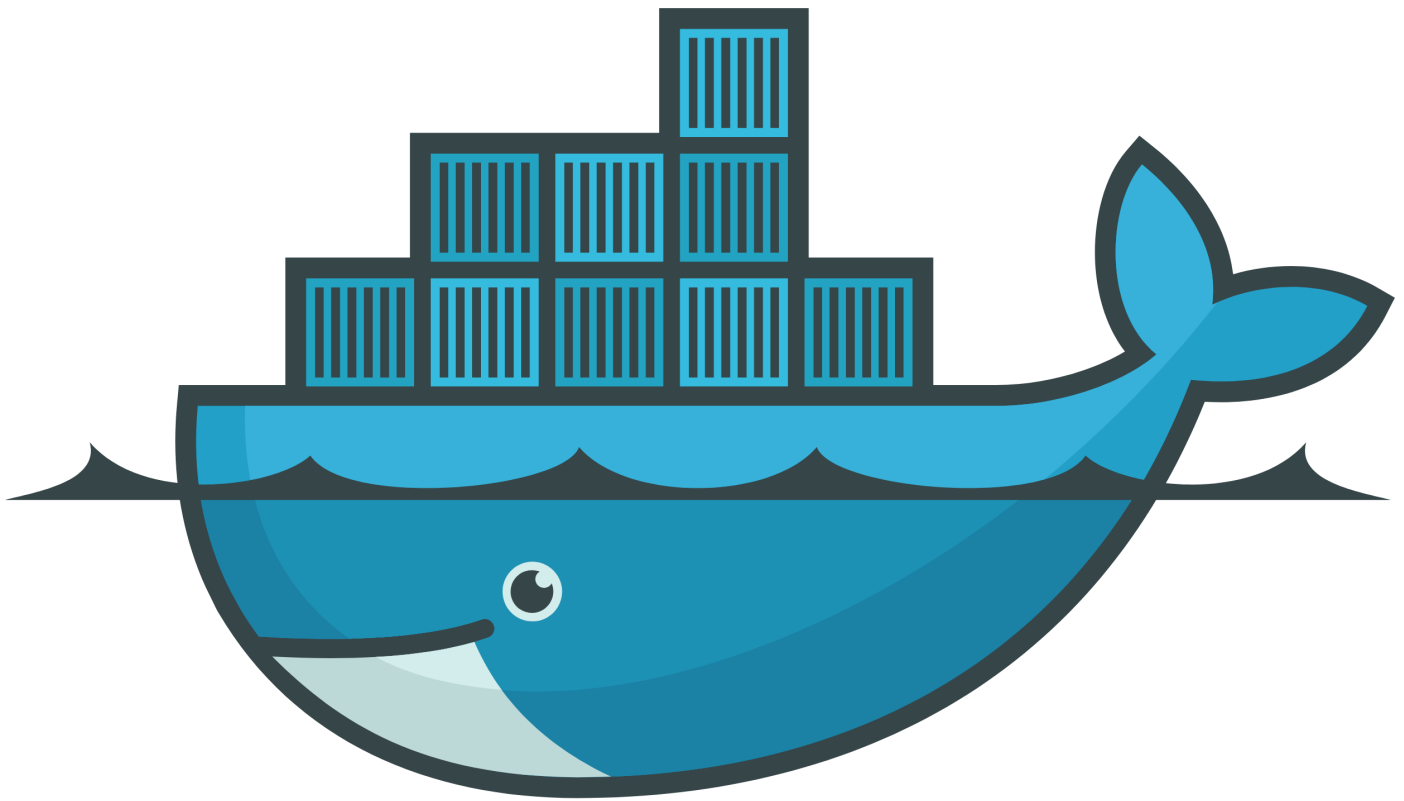
Se diseña e implementa una API con python y flask para servir los resultados de las encuestas, extraídos del logfile de Asterisk, mientras que se ejecuta una aplicación web con dash para visualizar los mismos.

Procedimiento

El entorno de desarrollo vive en una distribución GNU/Linux, Debian o basada en Debian.

Requisitos previos

Docker



docker

1. El primer paso es `eliminar instalaciones previas de docker`, en una terminal:

```
sudo apt-get remove docker docker-engine docker.io containerd runc  
sudo apt-get autoremove  
sudo apt-get autoclean
```

Si no había instalaciones existentes de docker, la terminal debería mostrarte esto:

```
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
Package 'docker-engine' is not installed, so not removed
```

2. Obtener algunas herramientas previas, en la terminal:

```
sudo apt-get update
```

Al finalizar, escribir:

```
sudo apt-get install \  
    apt-transport-https \  
    ca-certificates \  
    curl \  
    gnupg-agent \  
    software-properties-common
```

3. Agregar la GPG key oficial de docker:

```
sudo add-apt-repository \  
    "deb [arch=amd64] https://download.docker.com/linux/debian \  
    $(lsb_release -cs) \  
    stable  
OK
```

```
sudo apt-key fingerprint 0EBFCD88  
  
pub   4096R/0EBFCD88 2017-02-22  
      Key fingerprint = 9DC8 5822 9FC7 DD38 854A  E2D8 8D81 803C 0EBF CD88  
uid           Docker Release (CE deb) <docker@docker.com>  
sub   4096R/F273FCD8 2017-02-22
```

```
sudo add-apt-repository \  
    "deb [arch=amd64] https://download.docker.com/linux/ubuntu \  
    $(lsb_release -cs) \  
    stable"
```

4. Instalar docker-ce

```
sudo apt-get update  
sudo apt-get install docker-ce docker-ce-cli containerd.io
```


5. Elegir la versión de instalación

Listar las versiones disponibles para su instalación

```
apt-cache madison docker-ce
```

```
docker-ce | 5:18.09.6~3-0~debian-buster | https://download.docker.com/linux/debian buster/stable amd64 Packages
docker-ce | 5:18.09.5~3-0~debian-buster | https://download.docker.com/linux/debian buster/stable amd64 Packages
docker-ce | 5:18.09.4~3-0~debian-buster | https://download.docker.com/linux/debian buster/stable amd64 Packages
docker-ce | 5:18.09.3~3-0~debian-buster | https://download.docker.com/linux/debian buster/stable amd64 Packages
docker-ce | 5:18.09.2~3-0~debian-buster | https://download.docker.com/linux/debian buster/stable amd64 Packages
docker-ce | 5:18.09.1~3-0~debian-buster | https://download.docker.com/linux/debian buster/stable amd64 Packages
docker-ce | 5:18.09.0~3-0~debian-buster | https://download.docker.com/linux/debian buster/stable amd64 Packages
docker-ce | 18.06.3~ce~3-0~debian | https://download.docker.com/linux/debian buster/stable amd64 Packages
docker-ce | 18.06.2~ce~3-0~debian | https://download.docker.com/linux/debian buster/stable amd64 Packages
docker-ce | 18.06.1~ce~3-0~debian | https://download.docker.com/linux/debian buster/stable amd64 Packages
docker-ce | 18.06.0~ce~3-0~debian | https://download.docker.com/linux/debian buster/stable amd64 Packages
docker-ce | 18.03.1~ce-0~debian | https://download.docker.com/linux/debian buster/stable amd64 Packages
docker-ce | 18.03.0~ce-0~debian | https://download.docker.com/linux/debian buster/stable amd64 Packages
docker-ce | 17.12.1~ce-0~debian | https://download.docker.com/linux/debian buster/stable amd64 Packages
docker-ce | 17.12.0~ce-0~debian | https://download.docker.com/linux/debian buster/stable amd64 Packages
```

6. ¡Instalar docker!

Para este ejemplo de instalación se usa `5:18.09.6~3-0~debian-buster` (la primera opción). En el siguiente comando se reemplaza `<VERSION_STRING>` `docker-ce-cli=<VERSION_STRING>` la versión deseada:

```
sudo apt-get install docker-ce=<VERSION_STRING> docker-ce-cli=<VERSION_STRING>
containerd.io
```

```
sudo apt-get install docker-ce=5:18.09.6~3-0~debian-buster docker-ce-cli=5:18.
09.6~3-0~debian-buster containerd.io
```

Para verificar la instalación ingresar el siguiente comando en la terminal:

```
sudo docker run hello-world
```

Si todo está bien se muestra en la terminal:

```
Hello from Docker!
This message shows that your installation appears to be working correctly.
```

Git




```
sudo apt install git
```

Asterisk + FreePBX + Módulos IVR

Descargar y levantar la imagen

```
git clone https://github.com/flaviostutz/freepbx
cd freepbx
sudo docker-compose up -d
```

Ir a <http://localhost/>

 FreePBX Support

Welcome to FreePBX Administration!

Initial Setup

Please provide the core settings that will be used to administer and update your system

Administrator User

Username

Admin user name

Password

Admin password

Confirm Password

Admin password

System Notifications Email

Notifications Email address

Email Address

System Identification

System Identifier

VoIP Server

System Updates

Automatic Module Updates

Enabled

Email Only

Disabled

Automatic Module Security Updates

Enabled

Email Only

Send Security Emails For Unsigned Modules

Enabled

Disabled

Check for Updates every

Saturday

Between 4am and 8am

Setup System



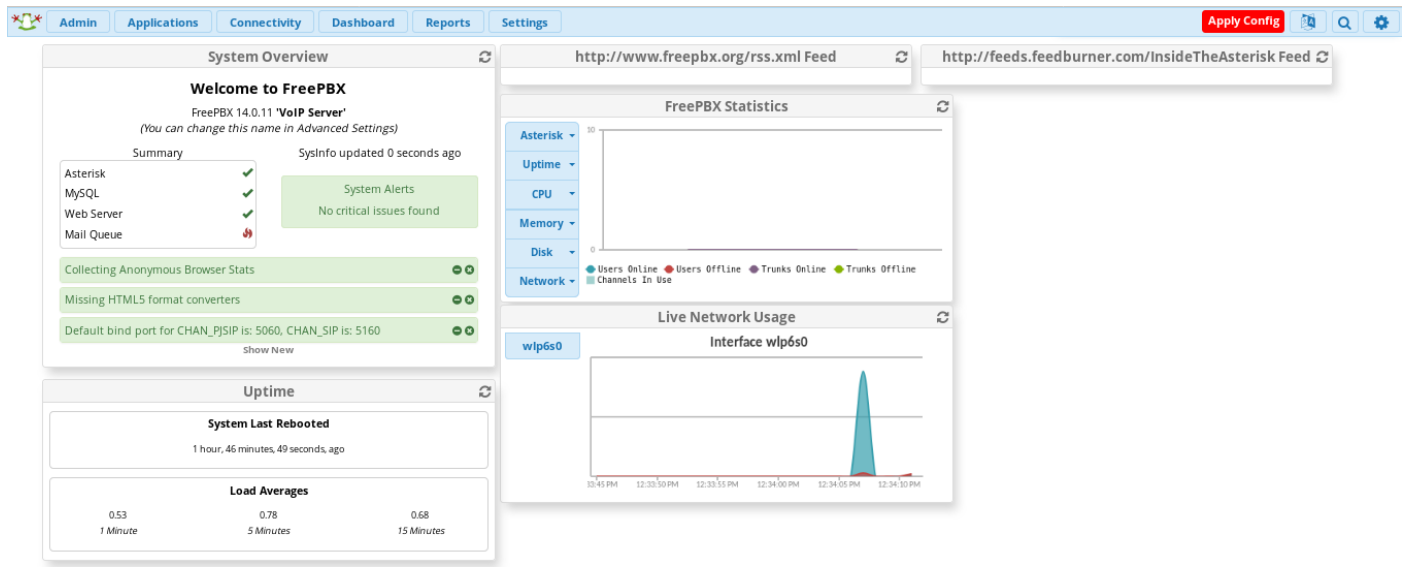
FreePBX is a registered trademark of
Sangoma Technologies Inc.
FreePBX 14.0.11 is licensed under the GPL.
Copyright© 2007-2019



FreePBX

Configuración básica en FreePBX

Setear un nombre de usuario y contraseña, email e iniciar la configuración.



FreePBX is a registered trademark of
Sangoma Technologies Inc.
FreePBX 14.0.11 is licensed under the GPL
Copyright© 2007-2019



Configuración FreePBX

Agregar extensiones y test básico

Se agregan dos extensiones PJSIP (usuarios) en la red de área local para llevar a cabo un test de funcionamiento inicial del servidor.

Breve descripción de PJSIP

PJSIP es una librería de código abierto para la comunicación multimedia, escrita en `C` que implementa los protocolos SIP, SDP, RTP, entre otros.

En el menú `Aplicattions -> Extensions`, en la pestaña `PJSIP Extensions` `Add Extension`.
Se setean parámetros para dos usuarios de testing:

All Extensions **Custom Extensions** **DAHDI Extensions** **IAX2 Extensions**

Quick Create Extension **Delete**

Extension	Name	CW	DND	FMFM	CF	CFB	CFU	Type	Actions
No matching records found									

List Extensions

- Add New Custom Extension
- Add New DAHDI Extension
- Add New IAX2 Extension
- Add New PJSIP Extension**
- Add New Chan_SIP Extension
- Add New Virtual Extension

GeneralVoicemailFind Me/Follow MeAdvancedOther

— Add Extension

This device uses **CHAN_SIP** technology listening on **0.0.0.0:5060**

User Extension ?

Display Name ?

Outbound CID ?

Secret ?

33a3a0c20c91eb00292e7eddf689d0e6

— User Manager Settings

Link to a Default User ?

Create New User

Username ?

☐ Use Custom Username

Password For New User ?

e4bb4c7370da97b66f2f289187dae74a

Groups ?

Select Some Options

Usuario 1:

- User Extension: 101
- Display Name: bibiana
- Outbound CID:
- Secret: 101

Usuario 2:

- User Extension: 102
- Display Name: lucas
- Outbound CID:
- Secret: 102

Para averiguar la dirección IP de la LAN que corresponde al servidor de VoIP:

```
ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
```


```







inet 127.0.0.1/8 scope host lo
    valid_lft forever preferred_lft forever
inet6 ::1/128 scope host
    valid_lft forever preferred_lft forever
2: enp9s0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast state
DOWN group default qlen 1000
    link/ether 74:86:7a:fe:37:56 brd ff:ff:ff:ff:ff:ff
3: wlp6s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state U
P group default qlen 1000
    link/ether 9c:2a:70:d4:07:95 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.29/24 brd 192.168.1.255 scope global dynamic noprefixroute
wlp6s0
    valid_lft 83313sec preferred_lft 83313sec
    inet6 fdc8:94bb:2fce:7c00:62f0:587d:3bb1:5034/64 scope global dynamic nopr
efixroute
    valid_lft 6943sec preferred_lft 3343sec
    inet6 fe80::cebd:1b6c:b719:3da4/64 scope link noprefixroute
    valid_lft forever preferred_lft forever

```

La interfáz de red es la **wlp6s0** y su dirección IPv4 es **192.168.1.29**. Que corresponde al *dominio* para los clientes.

Zoiper en Android

Se emplea el cliente de VoIP  para ambos usuarios y se inicia una llamada.







18:24

Account setup

Username @ PBX/VoIP provider

101@192.168.1.29

Password

101|



For example K23Rdw32

Create an account



18:24

Account setup

Fill in your hostname and select your provider from the list

hostname or provider

192.168.1.29

This could be called 'Domain', 'SIP Server', 'Registrar' or 'SIP Proxy'. For example 'sip.example.com' or '123.21.123.32:5060'.

Or you can just search for the name of your provider. May be we know the settings.

Next



18:24

Account setup



My provider/PBX requires an authentication username or outbound proxy

Authentication username

Outbound proxy

Skip



Seleccionar la opc Skip



18:25

Account setup

Please choose between the following configurations



SIP TLS

Not found

☐ SIP TCP

Not found

☒ SIP UDP

Found

☐ IAX UDP

Not found

Finish



18:25



Cuentas

SIP

IAX



101@192.168.1.29

Cuenta activada





Zoiper en Debian

Se realizan los mismos pasos de configuración

The screenshot shows the Zoiper5 web interface. On the left, a sidebar titled 'Accounts' lists a single SIP account: '102@192.168.1.29' with a green checkmark. Below the list is a green 'Add' button. The main panel displays the configuration for the selected account. At the top, the account name '102@192.168.1.29' is shown in orange, along with 'Unregister' and 'Advanced' links. Below this is a section titled 'SIP Credentials' with fields for 'Domain' (192.168.1.29), 'Username' (102), and 'Password' (masked with three dots). Another section, 'Optional SIP credentials', contains two unchecked checkboxes: 'Use auth. username' and 'Use outbound proxy', followed by an 'Outbound proxy' input field.

Misc Apps

Se requiere el módulo [miscapps](#) para dirigir las llamadas hacia una extensión, a la encuesta.

```
git clone https://github.com/FreePBX/miscapps
tar -cvzf miscapps.tar.gz miscapps
```

En **Admin** -> **Module Admin**

Admin

Module	Version	Track	Publisher	License	Status
Backup & Restore	14.0.10.3	Stable	Sangoma Technologie	GPLV3+	Enabled
Bulk Handler	13.0.14.8	Stable	Sangoma Technologie	GPLV3+	Enabled
Custom Applications	13.0.5.7	Stable	Sangoma Technologie	GPLV3+	Enabled
Feature Code Admin	13.0.6.4	Stable	Sangoma Technologie	GPLV3+	Enabled
FreePBX Framework	14.0.11	Stable	Sangoma Technologie	GPLV2+	Enabled
Process Management		Stable	Sangoma Technologie	AGPLV3+	Not Installed (Locally available: 13.0.7.1)
REST API		Stable	Sangoma Technologie	AGPLV3	Not Installed (Locally available: 13.0.21.2)
Recordings	13.0.30.13	Stable	Sangoma Technologie	GPLV3+	Enabled
Sound Languages	14.0.7	Stable	Sangoma Technologie	GPLV3+	Enabled

Applications

Module	Version	Track	Publisher	License	Status
Announcements	13.0.7.7	Stable	Sangoma Technologie	GPLV3+	Enabled
Call Recording	14.0.14	Stable	Sangoma Technologie	AGPLV3+	Enabled
Conferences	13.0.23.15	Stable	Sangoma Technologie	GPLV3+	Enabled
Core	14.0.25.4	Stable	Sangoma Technologie	GPLV3+	Enabled

Upload modules y elegir el tar.gz recién creado.

En **Module Admin**, elegir **Misc Apps** y click en **Install**.

Importación de announcements

Se crean o descargan los mensajes de voz con los que interactúa el usuario en formatos de audio y de 8bits.

Se generan audios a partir de un [conversor online de texto a voz](#)

En el menú **System recording**, click en **Announcements**, en el menú **Applications**, click en **Announcements**:

- **+Add recording**, setear un nombre y (opc) descripción del mensaje.
- Importar el archivo de sonido.
- Las opciones restantes permanecen en sus valores por defecto.

Repetir el procedimiento tantas veces como mensajes de voz requiera la aplicación.

Crear y configurar una aplicación IVR

Se crea, configura e implementa la aplicación IVR según la [wiki de FreePBX](#).

En el menú **Aplicaciones**, click en **IVR**:

IVR

[+ Add IVR](#)

Search

IVR Name



Actions

No matching records found

- [+Add IVR](#) , setear un nombre y (opc) descripción de la aplicación.
- Setear opciones adicionales de la aplicación.

Edit IVR: ID

— IVR General Options

IVR Name [?](#) ⓘ

IVR Description [?](#)

— IVR DTMF Options

Announcement [?](#) ⓘ

Enable Direct Dial [?](#) ⓘ

Timeout [?](#)

Invalid Retries [?](#)

Invalid Retry Recording [?](#) ⓘ

Append Announcement to Invalid [?](#)

Return on Invalid [?](#)

Invalid Recording [?](#) ⓘ

Invalid Destination [?](#) ⓘ

Timeout Retries [?](#)

Timeout Retry Recording [?](#) ⓘ

Append Announcement on Timeout [?](#)

Return on Timeout [?](#)

Timeout Recording [?](#) ⓘ

Timeout Destination [?](#) ⓘ

Return to IVR after VM [?](#)

— IVR Entries

Ext	Destination	Return ?	Delete
<input type="text" value="digits pressed"/>	<input type="text" value="== choose one =="/> ⓘ	<input type="button" value="Yes"/> <input checked="" type="button" value="No"/>	

+

- **IVR Name:** El sabor preferido de helado.
- **IVR Description:** Encuesta sobre preferencias en sabores de helado, para determinar el preferido de la comunidad local.
- **Announcement:** Bienvenida (*announcement creado previamente*).
- **Direct Dial:** (*Permite al usuario marcar la extensión con la que se desea comunicar directamente, dada la aplicación, se deshabilita.*) Disabled
- **Timeout:** 10
- **Invalid Retries:** 3
- **Invalid Retry Recording:**
- **Append Original Announcement:** (*Ante ingresos incorrectos del usuario, se reproduce el anuncio de bienvenida*) - ✓
- **Invalid Recording:** SorryAllDone
- **Invalid Destination:** Terminate Call:Hangup
- **Timeout Retries:** 2
- **Timeout Retry Recording:** SorryDidNotHearAnythingTryAgain
- **IVR Entries:** (*ver tabla*)

IVRs y announcements

Las aplicaciones IVR, sus entries, announcements y destinos definen la lógica de la encuesta:

1. Bienvenida

IVR	Announcement
Bienvenida sabores de helado	Bienvenida

Entry	Destino
1	IVR Género

2. Género

IVR	Announcement
Género	Género

Entry	Destino
1	Announcement Femenino
2	Announcement Masculino

3	Announcement No especifico
---	----------------------------

Announcement	Destino
Femenino	IVR Edad
Masculino	IVR Edad
No especifico	IVR Edad

3. Edad

IVR	Announcement
Edad	Edades

Entry	Destino
1	Announcement 18-21
2	Announcement 22-30
3	Announcement 30-40

Announcement	Destino
18-21	IVR Sabores de helado
22-30	IVR Sabores de helado
30-40	IVR Sabores de helado

4. Sabores

IVR	Announcement
Sabores	Sabores

Entry	Destino
1	Announcement chocolate

2	Announcement crema
3	Announcement frutilla
4	Announcement mascarpone

Announcement	Destino
chocolate	Hung up
crema	Hung up
frutilla	Hung up
mascarpone	Hung up

Misc App

En Applications -> Misc Applications -> Add

Misc Application

Enable	<input checked="" type="radio"/> Yes <input type="radio"/> No
Description: ?	encuesta
Feature Code ?	111
Destination ?	IVR Bienvenida encuesta

Adquisición de datos

Las opciones seleccionadas por los usuarios se corresponden a las señales **DTMF** (*Dual-Tone Multi-Frequency*), para registrar los mismos Settings -> Asterisk Logfile Settings y setear en On DTMF.


```
[2019-06-11 20:02:24] VERBOSE[4048][C-00000001] file.c: <PJSIP/101-00000000> Playing 'custom/0-bienvenida.slin'
```

implica el establecimiento de la llamada, las siguientes:

```
[2019-06-11 20:02:29] VERBOSE[4048][C-00000001] file.c: <PJSIP/101-00000000> Playing 'custom/2-genero.slin' (language 'en')
[2019-06-11 20:02:35] VERBOSE[4048][C-00000001] file.c: <PJSIP/101-00000000> Playing 'custom/2-1-femenino.slin' (language 'en')
```

Logfile

implican la elección de género “femenino” por parte de un usuario. Por lo anterior, y conociendo los announcements, se obtiene el logfile (desde el container) como sigue:

```
docker ps # para obtener el ID del container
docker exec -it 98fe1e53368f bash # para acceder a la consola
scp /var/log/asterisk/full bibiana@192.168.0.29:freepbx
```

API

Se genera una API a partir del procesamiento del archivo “full” con `flask-api`, detectando línea por línea las palabras “playing” y las que correspondan a los announcement de la señalización enviada por el usuario, se genera un diccionario con edades, géneros y sabores elegidos, y se sirven en la api.

```
pip install flask-api
```

```
import json
from flask_api import FlaskAPI

app = FlaskAPI(__name__)

f = open('full', 'r')

playing = []
res_encuesta = {'edades': {'18-21': 0, '21-30': 0, '30-40': 0},
                'géneros': {'F': 0, 'M': 0, 'NE': 0},
```

```

        'sabores': {'chocolate': 0, 'crema': 0, 'frutilla': 0, 'mascarpone': 0}}

for linea in f:
    if 'Playing' in str(linea):
        playing.append(linea)

for elem in playing:
    if '18-21' in elem:
        res_encuesta['edades']['18-21'] += 1
    elif '21-30' in elem:
        res_encuesta['edades']['21-30'] += 1
    elif '30-40' in elem:
        res_encuesta['edades']['30-40'] += 1
    if 'femenino' in elem:
        res_encuesta['géneros']['F'] += 1
    elif 'masculino' in elem:
        res_encuesta['géneros']['M'] += 1
    elif 'no-especifico' in elem:
        res_encuesta['géneros']['NE'] += 1
    if 'chocolate' in elem:
        res_encuesta['sabores']['chocolate'] += 1
    elif 'crema' in elem:
        res_encuesta['sabores']['crema'] += 1
    elif 'mascarpone' in elem:
        res_encuesta['sabores']['mascarpone'] += 1

@app.route('/')
def resultado_encuesta():
    return res_encuesta

if __name__ == '__main__':

    # Run app
    app.run(host="localhost",
            port=8000,
            debug=True)

```

```
python api.py
```

Visualización

Se emplea la librería `dash` para visualizar los datos servidos por la api, se realiza un request a la misma, se obtiene el json de los resultados, y se plotean gráficos de barra con las keys y values en los ejes x e y correspondientemente, un dropdown permite visualizar tanto los sabores como edades y géneros.

```
pip install dash
```

```
# coding: utf-8

import dash
import dash_core_components as dcc
import dash_html_components as html

import requests

#external_stylesheets = ['https://codepen.io/chriddyp/pen/bWLwgP.css']
dash_app = dash.Dash(__name__)#, external_stylesheets=external_stylesheets)
dash_app.config['suppress_callback_exceptions']=True

url_resultados = 'http://localhost:8000/'
req_resultados = requests.get(url_resultados)
resultados = req_resultados.json()

opciones = []
for key in resultados.keys():
    opciones.append({'label': key, 'value': key})

dash_app.layout = html.Div([
    dcc.Dropdown(
        id='my-dropdown',
        options=opciones,
        placeholder='Seleccione una opción'
    ),
```

```

dcc.Graph(id='my-graph'),
dcc.Interval(
    id='interval-component',
    interval=1000, # in milliseconds
    n_intervals=0
)
#   html.Div(id='output-container')
])

@dash_app.callback(
    dash.dependencies.Output('my-graph', 'figure'),
    [dash.dependencies.Input('my-dropdown', 'value')])

def update_graph(selected_dropdown_value):
    res = resultados[selected_dropdown_value]
    x = list(res.keys())
    y = list(res.values())

    return {
        'data': [
            {
                'x': x,
                'y': y,
                'name': selected_dropdown_value,
                'marker': {'size': 1},
                'showlegend': True,
                'type': 'bar'
            }
        ]
    }

@dash_app.callback(
    dash.dependencies.Output('output-container', 'children'),
    [dash.dependencies.Input('my-dropdown', 'value')])
def update_output(value):
    return 'Usuario: "{}".format(value)

```



```
if __name__ == '__main__':  
    dash_app.run_server(  
        host="localhost",  
        port=8001,  
        debug=True)
```

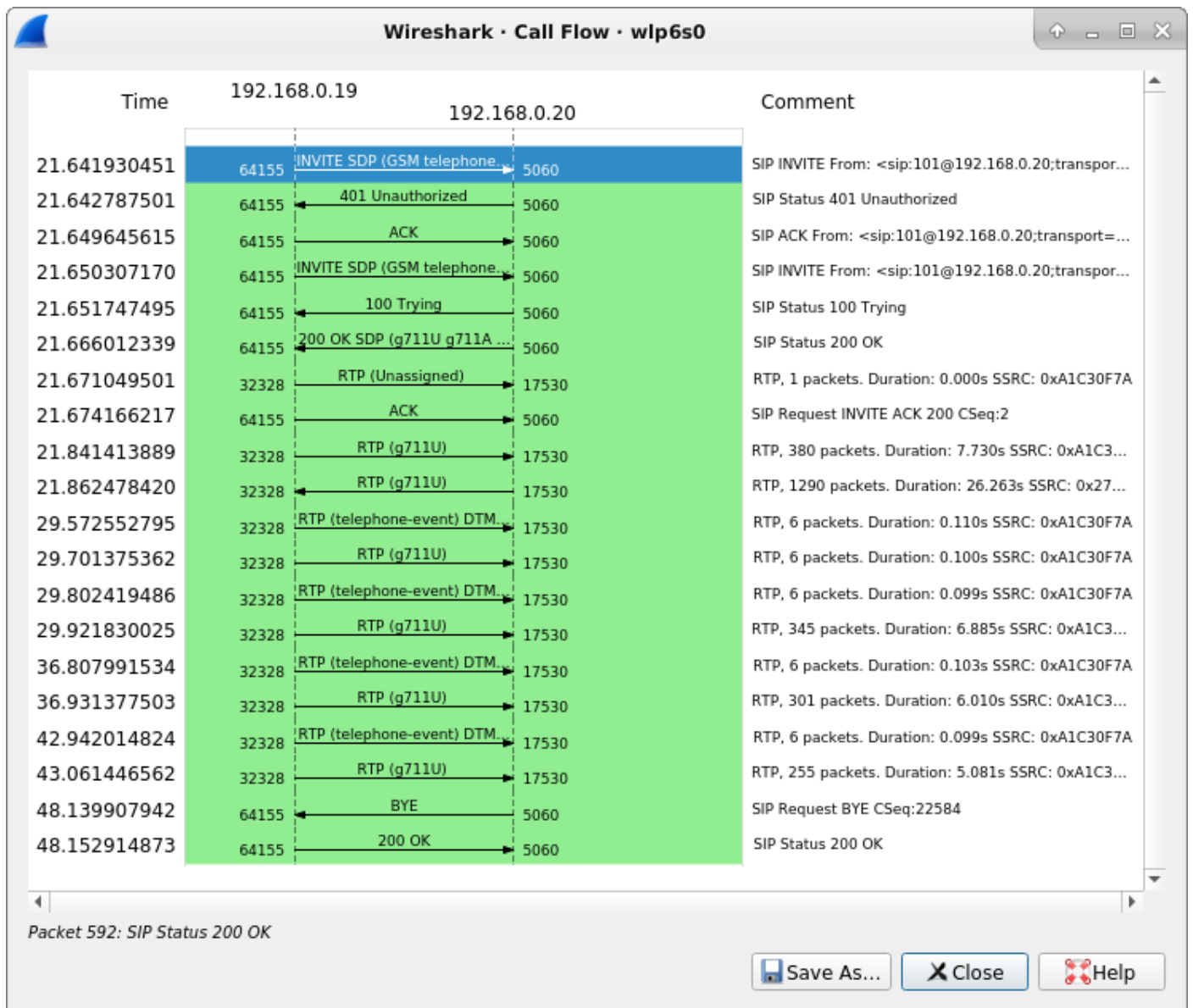
```
python dash_app_encuesta.py
```

Se accede a la API desde <http://localhost:8000> y <http://localhost:8001>.

Resultados

Se realizan llamadas desde clientes a la central, marcando la extensión correspondiente a la encuesta, se registran los resultados en `Wireshark`, empleando la opción `Telephony -> VoIP calls -> Flow Sequence`

```
sudo apt install wireshark  
sudo wireshark
```



Captura llamada de VoIP en Wireshark

Se pueden observar los paquetes correspondientes al protocolo **SIP** (Session Initiation Protocol), -capas 5, 6 y 7- de señalización, empleado para iniciar, mantener y terminar sesiones en tiempo real, en este caso de voz.

Capturando todos los paquetes entre las IP correspondientes al servidor de VoIP y el cliente:

***wlp6s0**

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

ip.dst == 192.168.0.19 or ip.src == 192.168.0.19

No.	Time	Source	Destination	Protocol	Length	Info
13	4.231127005	192.168.0.19	192.168.0.20	UDP	46	64155 → 5060 Len=4
71	11.540958891	192.168.0.19	224.0.0.251	MDNS	123	Standard query 0x0000 ANY Android.local, "QU" ques
73	11.745779127	192.168.0.19	224.0.0.251	MDNS	123	Standard query 0x0000 ANY Android.local, "QM" ques
75	12.053017907	192.168.0.19	224.0.0.251	MDNS	123	Standard query 0x0000 ANY Android.local, "QM" ques
77	12.259100888	192.168.0.19	224.0.0.251	MDNS	288	Standard query response 0x0000 PTR, cache flush Ar
81	13.283189129	192.168.0.19	224.0.0.251	MDNS	288	Standard query response 0x0000 PTR, cache flush Ar
90	15.331236808	192.168.0.19	224.0.0.251	MDNS	288	Standard query response 0x0000 PTR, cache flush Ar
125	19.324851457	192.168.0.19	224.0.0.251	MDNS	288	Standard query response 0x0000 PTR, cache flush Ar
587	21.641930451	192.168.0.19	192.168.0.20	SIP/SDP	918	Request: INVITE sip:111@192.168.0.20;transport=UDP
588	21.642787501	192.168.0.20	192.168.0.19	SIP	542	Status: 401 Unauthorized
589	21.649645615	192.168.0.19	192.168.0.20	SIP	388	Request: ACK sip:111@192.168.0.20;transport=UDP
590	21.650307170	192.168.0.19	192.168.0.20	SIP/SDP	1213	Request: INVITE sip:111@192.168.0.20;transport=UDP
591	21.651747495	192.168.0.20	192.168.0.19	SIP	350	Status: 100 Trying
592	21.666012339	192.168.0.20	192.168.0.19	SIP/SDP	887	Status: 200 OK
593	21.671049501	192.168.0.19	192.168.0.20	RTP	55	PT=Unassigned, SSRC=0xA1C30F7A, Seq=5359, Time=326
594	21.674166217	192.168.0.19	192.168.0.20	SIP	443	Request: ACK sip:192.168.0.20:5060
595	21.841413889	192.168.0.19	192.168.0.20	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xA1C30F7A, Seq=5360, Ti
596	21.861281420	192.168.0.19	192.168.0.20	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xA1C30F7A, Seq=5361, Ti
597	21.862478420	192.168.0.20	192.168.0.19	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x275F4E4E, Seq=16464, T
598	21.882499512	192.168.0.20	192.168.0.19	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x275F4E4E, Seq=16465, T
599	21.882753944	192.168.0.19	192.168.0.20	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xA1C30F7A, Seq=5362, Ti
600	21.901337239	192.168.0.19	192.168.0.20	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xA1C30F7A, Seq=5363, Ti

Frame 1406: 58 bytes on wire (464 bits), 58 bytes captured (464 bits) on interface 0

Ethernet II, Src: LgElectr_0c:53:80 (5c:af:06:0c:53:80), Dst: HonHaiPr_d4:07:95 (9c:2a:70:d4:07:95)

Internet Protocol Version 4, Src: 192.168.0.19, Dst: 192.168.0.20

User Datagram Protocol, Src Port: 32328, Dst Port: 17530

Real-Time Transport Protocol

```

0000  9c 2a 70 d4 07 95 5c af 06 0c 53 80 08 00 45 00  *p... \...S...E
0010  00 2c 7d bd 40 00 40 11 3b 8c c0 a8 00 13 c0 a8  ,}..@..;.....
0020  00 14 7e 48 44 7a 00 18 f3 e1 80 e5 16 6c c2 d3  ...HDz...1...
0030  ba 94 a1 c3 0f 7a 01 0a 00 a0                    ....Z...

```

wireshark_wlp6s0_20190611202758_810jRt.pcapng Packets: 3930 · Displayed: 2650 (67.4%) · Dropped: 0 (0.0%) Profile: Default

Captura con filtro Ip scr e IP dst en Wireshark

Se observa el protocolo **RTP** (*Real Time Protocol*) involucrado -capa 7-, así como el protocolo de transporte **UDP**, típicamente implementado en comunicaciones multimedia en tiempo real, sin corrección de errores ni retransmisión de paquetes.

API

localhost:8000

90%

...

Search

Flask API v1.1

Resultado encuesta

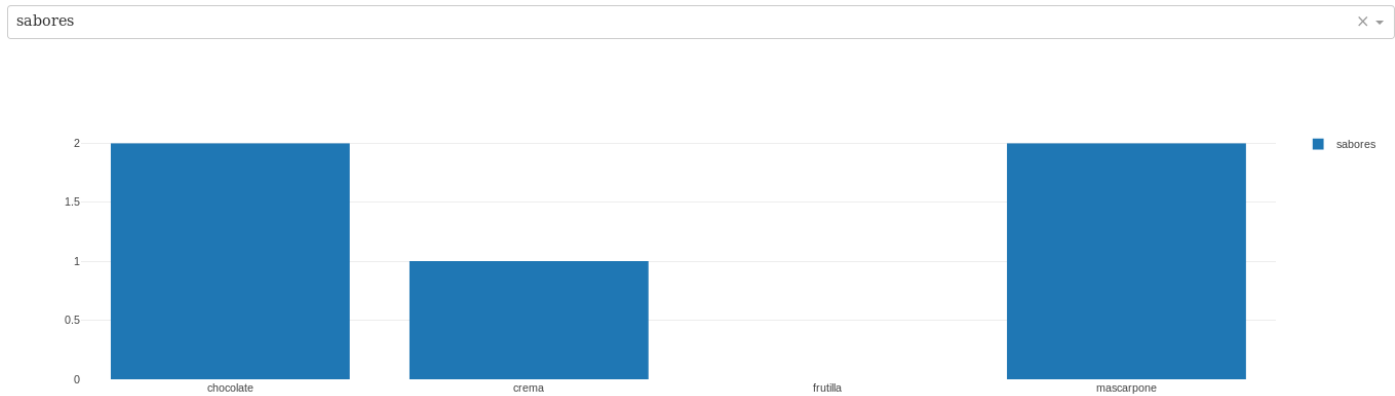
GET http://localhost:8000/

HTTP 200 OK
Content-Type: application/json

```
{
  "edades": {
    "18-21": 4,
    "21-30": 4,
    "30-40": 2
  },
  "géneros": {
    "F": 2,
    "M": 2,
    "NE": 1
  },
  "sabores": {
    "chocolate": 2,
    "crema": 1,
    "frutilla": 0,
    "mascarpone": 2
  }
}
```

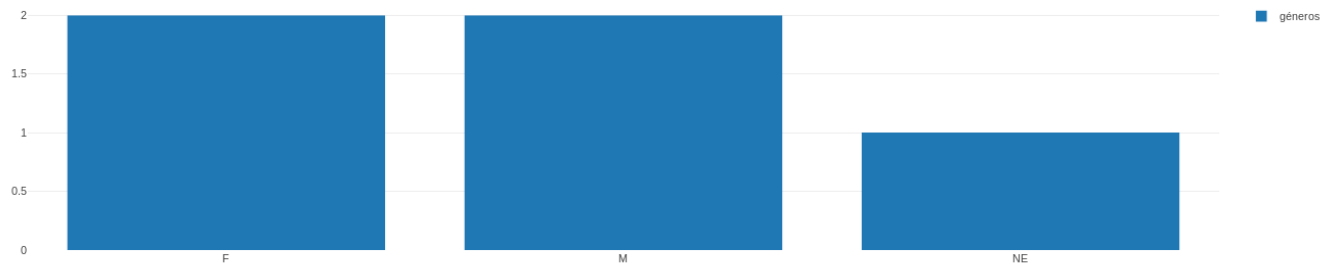
http://localhost:8000

Visualización



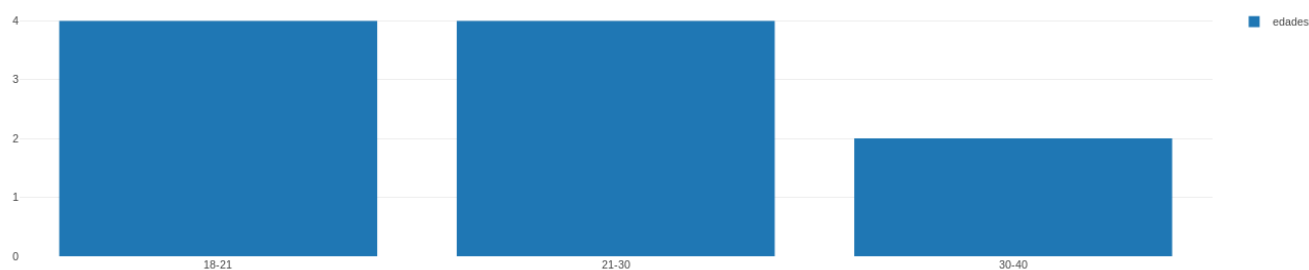
géneros

X



edades

X



<http://localhost:8001>

Conclusión

La elección y combinación de herramientas open source o libres forman satisfactoriamente un sistema de encuestas, empleando la implementación y configuración de aplicaciones IVR pensadas para menú, con el correspondiente procesamiento de datos. Se logra entonces la puesta en marcha de un servidor de voz sobre IP, módulos de interacción por voz, adquisición y visualización de datos de la encuesta.

Proyección

Siendo el sistema un prototipo, se consideran las siguientes mejoras o agregados:

- Obtener un número público para trascender la red LAN.
- Considerar encuestas trucas en el procesamiento de datos.
- Correlacionar géneros, edades y sabores para un mejor análisis de los resultados.

Referencias

- [Documentación Docker](#)
- [Imagen docker con Asterisk + FreePBX + IVR](#)
- [Documentación Asterisk](#)
- [Wiki FreePBX](#)
- [Flask API](#)
- [Documentación Dash](#)