ASTERISK

Configuration

Variable

${EXTEN} récupère le numérop de l'extension actuel
${STRFTIME(${EPOCH},,%Y:%m:%d_%Hh%Mm%Ss)} récupère la date et l'heure actuel.

Voicemail

/etc/asterisk/**extention.conf :**

```
exten => 3000,1,NoOp()
same => n,Dial(SIP/${EXTEN},30)
same => n,VoiceMail(${EXTEN}@default)
same => n,Hangup

exten => 456,1,VoiceMailMain()
same => n,HangUp()
```

/etc/asterisk/**voicemail.conf :**

```
[general]
format=wav
maxmsg=100
maxsec=60
minsec=3
skipms=3000
maxsilence=10
silencethreshold=128
maxlogin=3

[default]
3000 => 123
```

Enregistrement d'appel

/etc/asterisk/**extensions.conf :**

```
exten => 1000,1,NoOp()
same => n,Set(CurrentDateTime=${STRFTIME(${EPOCH},,%Y:%m:%d_%Hh%Mm%Ss)})
same => n,MixMonitor(${EXTEN}-${CurrentDateTime}.wav)
```

Fichier audio enregistrer en .wav dans le dossier “/var/spool/asterisk/monitor/”
si nous somme le 10 juin 2023 à 10H30 et 20 sec au moment de l’appel le nom du fichier sera : 1000-2023:06:10_10h30m20s

File d’attente & Pré-décrochage

/etc/asterisk/**queues.conf** :

```
[file-attente]
musicclass=classic
strategy=ringall
timeout=20
member => SIP/x
```

/etc/asterisk/**extensions.conf** :

```
exten => x,1,Answer()
same => n,Queue(file-attente)
same => n,Hangup
```

+ Musique personnalisé :

créer un dossier attente dans le répertoire /var/lib/asterisk/moh et mettre la musique de son choix dans celui-ci.

/etc/asterisk/**musiconhold.conf :**

```
[musique]
mode=files
directory=moh/attente
```

/etc/asterisk/**extensions.conf :**

```
exten => x,1,NoOp()
same => n,Answer()
same => n,Set(CHANNEL(musicclass)=musique)
same => n,Queue(file-attente)
same => n,Hangup
```

Salle de conférence

/etc/asterisk/**confbridge.conf :**

```
[user]
type=user
admin=no
music_on_hold_when_empty=yes
announce_user_count=yes
```

```
[conf]
type=bridge
max_members=5
```

/etc/asterisk/**extensions.conf :**

```
exten => 999,1,NoOp()
same => n,Answer()
same => n,ConfBridge(Salle_1,conf,user)
same => n,Hangup()
```

Serveur Vocal Intéractif (SVI)

/etc/asterisk/**extensions.conf** :

```
exten => 900,1,Goto(svi_1,s,1)

[svi_1]
exten => s,1,Answer()
exten => s,2,Set(TIMEOUT(response)=10)
exten => s,3,Playback(en/service1)
exten => s,4,Playback(en/service2)
exten => s,5,WaitExten()

exten => 1,1,Dial(SIP/y)
exten => 2,1,Dial(SIP/z)
exten => _[04-9*#],1,Goto(svi_1,s,1)

exten => t,1,Goto(svi_1,s,3)
```

Pour faire fonctionner les fichiers audios, il faudrait qu'ils soient en 4 formats : ulaw, alaw, gsm, wav.
Pour convertir pour pouvez utiliser sox :

```
apt install sox
sox -V service2.wav -r 8000 -c 1 -e ul service1.ulaw
```

Ecoute/Chuchotement

/etc/asterisk/**extensions.conf** :

```
exten => 52,1,Answer()
exten => 52,n,ChanSpy(SIP/1000,q)
```

"q" est pour écouter sans être entendu

Pour écouter et être entendu par les deux utilisateurs changer "q" par "B"

Voici une conf qui permet de choisir directement l'utilisateur que vous voulez écouter :
/etc/asterisk/**extensions.conf** :

```
exten => 52,1,Goto(spy,s,1)

[spy]
exten => s,1,Answer()
same => n,WaitExten()

exten => _[01-9*#],1,NoOp()
same => n,Chanspy(SIP/${EXTEN},q)
```

Vous appelez le numéro 52, puis vous devez saisir le numéro que vous souhaitez écouter.

Mode absence & Emplois du temps

Temps :

de 17h à minuit et de minuit à 9h les appel sont renvoyer vers l'extension "close" en initialisant la variable "GLOBAL(MY_EXTEN)" avec le numéro d'extension appelé
ce qui permet de laisser un message sur la bonne boite vocal

Mode Absence sur demande :

Quand l'extension appel le *77 le variable GLOBAL(ABSENCE_MODE${EXTEN} passe à 1 ce qui active le mode absence.
Quand l'extension appel le *88 cela désactive le mode absence.
Quand la variable "GLOBAL(ABSENCE_MODE${EXTEN})" est sur 1 l'appel est redirigé vers l'extension "close" toujours avec l'option "GLOBAL(MY_EXTEN)" initialisé.

/etc/asterisk/**extensions.conf** :

```
exten => 1000,1,NoOp()
same => n,Set(GLOBAL(MY_EXTEN)=${EXTEN})
same => n,GotoIfTime(17:00-23:59,mon-fri,*,*?close,1)
same => n,GotoIfTime(00:00-8:59,mon-fri,*,*?close,1)
same => n,GotoIf($"[${GLOBAL(ABSENCE_MODE${EXTEN})}" = 1]?close,1)
same => n,Dial(SIP/${EXTEN},15)
same => n,Hangup


exten => close,1,NoOp()
same => n,Playback(closed)
same => n,VoiceMail(${GLOBAL(MY_EXTEN)}@default)
same => n,Hangup()
```

```
exten => *77,1,NoOp()
same => n,Set(GLOBAL(ABSENCE_MODE${CALLERID(num)})= 1)
same => n,Playback(goodbye)
same => n,Hangup()

exten => *88,1,NoOp()
same => n,Set(GLOBAL(ABSENCE_MODE${CALLERID(num)})= 0)
same => n,Playback(hello)
same => n,Hangup()
```

Ligne de commande

Entrée CLI :
asterisk -r

Ecriture commandes direct :
asterisk -x “ma commande”

Création d'un appel

asterisk -r
channel originate SIP/2000 extension 1000@internal

Rediriger un channel

core show channels
channel redirect SIP/2000-00000010 internal,1000,1

L’utilisateur 2000 est redirigé vers le 1000

Mise en écoute

ChanSpy(SIP/x)

Récupération dans la liste d'attente

Liste des appels en cours

asterisk -r
core show channels concise

Liste des appels en liste d'attente

asterisk -r
queue show

Liste des appels enregistrés

ls -lt /var/spool/asterisk/monitor/

Liste des 'personnes' dans une salle de conférence

asterisk -r
confbridge list nom_conference

Récapitulatif

[internal]

; Utilisateur
exten => 1000,1,NoOp()
→ No Operation, rien ne ce passe

same => n,System(mmctl post create ${IDMattermost}:${IDConvGustave} --message "Message en provenance de ${CALLERID(num)} pour ${EXTEN}")
→ Envoi de message a Mattermost (voire après)

same => n,Set(GLOBAL(MY_EXTEN)=${EXTEN})
→ Création de la variable de type Global MY_EXTEN qui prend pour valeur le num de l'extension qui est appelé

same => n,Set(CurrentDateTime=${STRFTIME(${EPOCH},,%Y:%m:%d_%Hh%Mm%Ss)})
→Création de la variable de type privée CurrentDateTime qui prend pour valeur la date et l'heur actuel

same => n,GotoIfTime(17:00-23:59,mon-fri,*,*?close,1)
same => n,GotoIfTime(00:00-8:59,mon-fri,*,*?close,1)
→ Redirige Vers l'extension “close ” SI il est entre 17h et 9h

same => n,GotoIf($["${GLOBAL(ABSENCE_MODE${EXTEN})}"= 1]?close,1)

→ Redirige Vers l'extension “close ” SI le mode absence est activé

same => n,MixMonitor(${EXTEN}-${CurrentDateTime}.wav)
→ enregistre la conversation sous le nom de l'extension actuel - la date actuelle

same => n,Dial(SIP/${EXTEN},15)
→ fait sonner le téléphone pendant 15 sec

same => n,VoiceMail(${EXTEN}@default)
→ permet à l'appelant de laisser un msg vocal sur la boit vocal de l'extension qui a appeler

same => n,Hangup
→ raccroche après avoir laisser un msg

Mattermost

Mattermost est la solution d'envoi de message que j'ai sélectionné. Pour installé et configuré Mattermost et PostgreSQL : https://docs.mattermost.com/install/installing-ubuntu-2004-LTS.html

Une fois configurée il faut installer mmctl pour effectuer des commandes pour mattermost : https://docs.mattermost.com/manage/mmctl-command-line-tool.html

Après cela se connecter a Mattermost sur navigateur ou avec l'app desktop
et créé son utilisateur.

Envoi de massage pour chaque appel

Pour autoriser Asterisk a utilisée mmctl il faut configuré l'utilisateur asterisk

$ mmctl user create --username=asterisk --email=asterisk@gmail.com --password=MyPassWord
$ sudo su asterisk
$ mmctl auth login http://url_du_serveur --name community --username asterisk --password MyPassWord

Dans Mattermost ajouter l'utilisateur asterisk qu'on vient de créer
Regarder l'id de conv en message direct entre votre user et l'user asterisk

IDMattermost corespond au nom du serveur mattermost et IDConvGustave

Dans **extensions.conf :**

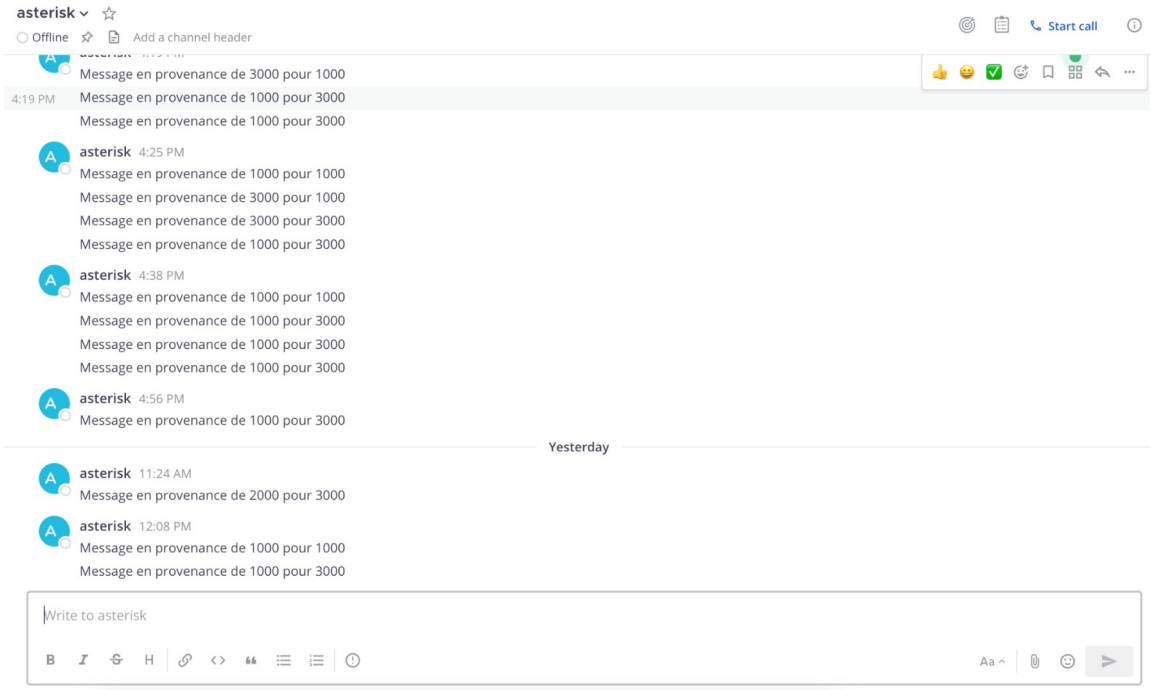
creation de variable global:

```
[globals]
IDMattermost = test
IDConvGustave = y6qktfy6ht8e9gf6i3rxmst7no
```

dans chaque extension mettre

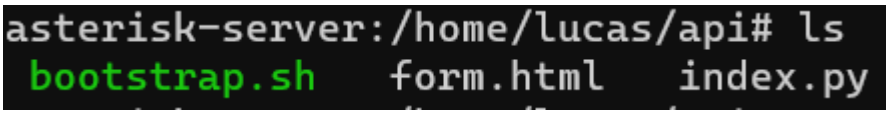
```
same => n,System(mmctl post create ${IDMattermost}:${IDConvGustave} --message "Message en provenance de ${CALLERID(num)} pour ${EXTEN}")
```

exemple de message envoyer



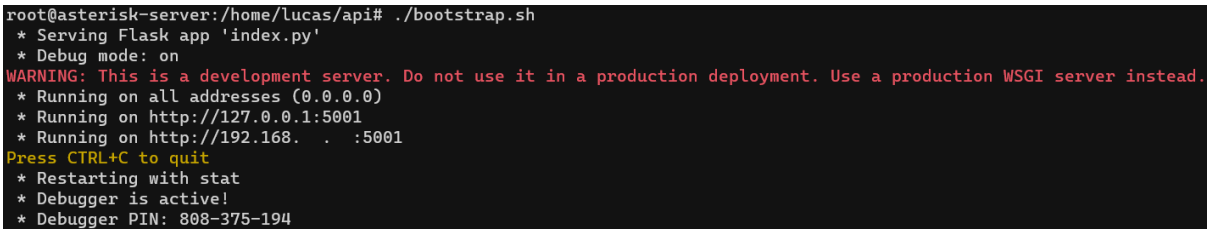
API Flask Python

Les fichiers de l'API sont situés dans /home/lucas/api. Le dossier est constitué de 3 fichiers :



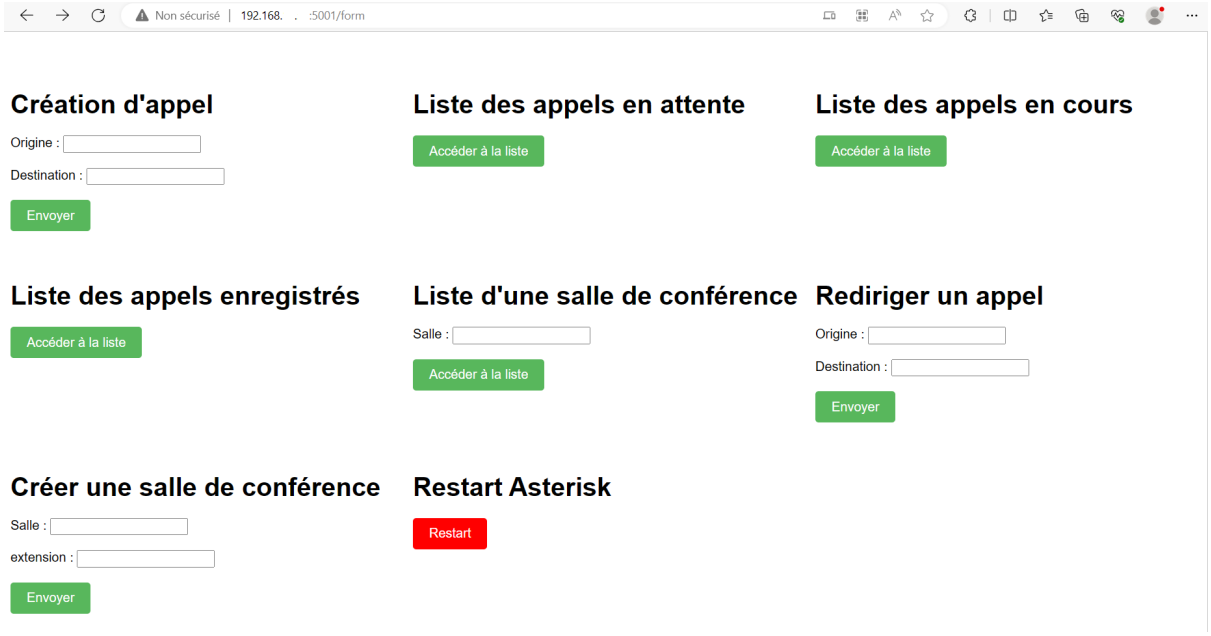
bootstrap.sh

Le fichier est constitué de commandes bash qui permet de lancer l'API Flask Python. Il contient la localisation du fichier python, le port sur lequel se lance l'API (ici le port est réglé sur 5001 de base il s'agit de 5000) et les adresses IP qui peuvent y accéder. Pour l'exécuter il suffit de réaliser la commande :
cd /home/lucas/api
./bootstrap.sh



form.html

Le fichier contient du code html simple permettant d'avoir un formulaire des commandes Asterisk. Pour y accéder : http://@IP:5001/form



C'est d'ici que vous exécuterez les commandes.

De plus, le formulaire HTML facilite l'interaction entre les utilisateurs et l'application en permettant la collecte et le traitement des données. Cela offre aux utilisateurs une interface conviviale et simple d’utilisation pour interagir avec l'API et permet à l'application de recevoir des informations pour son fonctionnement.

index.py

Ce fichier contient le code nécessaire pour créer et configurer l'application Flask, ainsi que pour définir les routes et les fonctionnalités de l'API. Le fichier index.py est l'endroit où ces configurations sont définies à l'aide de variables. Les routes définissent les URL auxquelles l'API Flask répondra et les fonctions qui seront exécutées pour chaque route. Pour définir les routes il faut utiliser @app.route('/exemple'). C’est donc dans ce fichier que contient toute la configuration des différents commandes Asterisk, la création d’appel, la redirection d’un appel, la création d’une salle de conférence, redémarrer le service Asterisk et les listes des appels en cours, des appels en attente, des appels enregistrés et des utilisateurs dans une salle de conférence.

Configuration

Chaque def effectue une commande et les commandes qui retourne des informations importantes sont parser et retourner en XML.