

How-to Asterisk

Ce How-to a été réalisé avec la version 1.0.7 d'Asterisk

- Ce tutorial est a été réalisé à titre d'exemple et la configuration du serveur Asterisk peut être largement personnalisée et enrichie.

1. INSTALLATION D'ASTERISK SUR UNE DEBIAN SARGE 3.1R2

- Lancez l'installation des paquets suivants :

```
apt-get install asterisk asterisk-config asterisk-doc asterisk-sounds-main  
asterisk-prompt-fr
```

- Editez le fichier `/etc/default/asterisk`, on modifie la ligne :

```
RUNASTERISK=no
```

par

```
RUNASTERISK=yes
```

Cela permettra de lancer le service asterisk au démarrage ainsi que par la commande `/etc/init.d/asterisk`

- On lance donc asterisk

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

La ligne de commande `asterisk -r` permet d'afficher la console de commande du processus d'asterisk en cours.

Il est possible de recharger la configuration grâce à la commande `reload`.

La configuration d'Asterisk se passe dans plusieurs fichiers de configurations situées dans `/etc/asterisk/` :

Le premier fichier à modifier est `sip.conf` :

- On recherche la ligne `;language=us`, dé commentez la et placez la valeur à fr

```
language=fr
```

- On se place ensuite à la fin du fichier pour passer à la configuration des utilisateurs :

```
[1001]  
type=friend  
username=1001  
secret=1001  
host=dynamic  
callerid="Maxime"  
mailbox=1001@default
```

- On va fixer un numéro pour appeler cet utilisateur, dans le fichier `extensions.conf` :

```
exten => 1001,1,Dial(SIP/1001,20,tr)
```

On pourra désormais appeler l'utilisateur 1001 en appelant le numéro 1001, 20 est le timeout avant de raccrocher.

2. VOICEMAIL ET BOITES VOCALES

- On rajoute les lignes suivantes dans le fichier `voicemail.conf` :

```
1001 => 1001,Maxime,maxime@localhost,,|attach=no|review=yes
```

Le premier 1001 est le numéro de la boîte, le deuxième est le mot de passe de la boîte vocal, puis vient le numéro de l'utilisateur, son adresse mail, ainsi que des options.

Un message sera envoyé à l'adresse mail lorsqu'un nouveau message sera laissé sur le répondeur le fichier son ne sera pas en pièce jointe du mail grâce à l'option `attach=no`, `review=yes` permet à celui qui laisse un message de relire ou de réenregistrer son message.

Les messages vocaux sont stockés dans le répertoire `/var/spool/asterisk`

- Ajouter dans `/etc/asterisk/extension.conf` la commande pour activer le répondeur si la personne ne répond pas :

```
exten => 1001,2,Voicemail(1001)
```

- Mettre le message du mail en français en éditant le fichier `/etc/asterisk/voicemail.conf` et remplacer le message anglais par un message en français :

```
emailsubject=Nouveau message numero ${VM_MSGNUM} dans la boite ${VM_MAILBOX}
emailbody=Cher(e) ${VM_NAME},\n\n\tVous avez un nouveau message de la part de
${VM_CALLERID}.\n Ce message dure ${VM_DUR} et il a ete envoye ${VM_DATE},\nvous
pouvez consulter votre boite ${VM_MAILBOX} pour ecouter ce message.\n\n\t\t\t\t\tLe
Service de Messagerie\n
```

Recharger la configuration du serveur Asterisk via le client en ligne de commande pour valider les modifications :

```
Reload
```

3. IVR

Création des agents et des files d'attente :

- Editer le fichier `/etc/asterisk/agents.conf` et ajouter des agents qui seront chargés de répondre aux appels sur les files d'attente :

```
agent => 2001,2001,Maxime
```

Le premier 2001 correspond à l'identifiant de l'agent, le second au mot de passe, Maxime est le nom de l'agent.

On va ensuite créer une file d'attente et y associer des agents :

```
[default]
member => Agent/2001
```

On associe l'agent 2001 à la file d'attente *default*, on peut créer d'autres files d'attente en créant de nouveaux contextes.

L'option `AgentLogin()` permettra à l'agent de s'authentifier et de recevoir des appels depuis la file d'attente, dans notre cas en appelant le numéro de 800 :

```
exten => 800,1,AgentLogin()
```

Configuration pour la musique d'attente

- Afin que les utilisateurs qui se retrouvent dans la file d'attente puissent avoir le droit à de la musique d'attente, on modifie le fichier `/etc/asterisk/musiconhold.conf`.

```
default => quietmp3:/var/asterisk/sounds/mp3
```

Le dossier de musique d'attente est donc maintenant configuré, il n'y a plus qu'à ajouter les différentes pistes à jouer dans ce dossier.

Attention : il semblerait qu'il y ait quelques problèmes avec la version 1.0.7 d'Asterisk rendant parfois la musique d'attente inactive. Ce problème ne semble pas toucher les versions plus récentes (1.2.7) qui ne sont cependant pas disponible avec la distribution stable de GNU/Linux Debian.

Modification du fichier `extensions.conf` :

- Editer le fichier `/etc/asterisk/extensions.conf` dans le contexte `[default]` pour commencer par y définir le numéro auquel le serveur vocal sera joint.

```
exten => 100,1, Ringing()
exten => 100,2, Wait(4)
exten => 100,3, Goto(accueil,s,1)
```

La première ligne va permettre d'obtenir une tonalité et la deuxième un temps d'attente de 4 secondes.

Enfin, la troisième ligne renverra la communication vers un autre contexte nommé `accueil` qui servira de menu d'accueil pour l'utilisateur. La fonction `Goto()` redirige vers une priorité d'extension d'un contexte précis, ici on va rejoindre la priorité `1` de l'extension `s` présente dans le contexte `accueil`.

Remarque : les deux premières lignes (`Ringing` et `Wait`) sont facultatives. Néanmoins elles apportent un confort supplémentaire à l'utilisateur.

Contexte d'accueil :

- *Source*

```
[accueil] ; définition d'un contexte pour l'accueil
exten => s,1,SetGlobalVar(sounds_path=/var/asterisk/sounds/)
exten => s,2,Background(${sounds_path}welcome)
exten => #,1,Goto(menu,s,1)
exten => i,1,Playback(${sounds_path}erreur-saisie)
exten => t,1,Goto(accueil,s,1)
```

- *Détails*

Extensions prédéfinies :

- **i** lancée lorsqu'une extension est invalide ;
- **s** lancée au lancement du contexte ;
- **h** lancée lors que l'appel se termine ;
- **t** lancée au bout du timeout si l'appel n'est pas prit ;
- **T** lancée au bout de la durée maximale de communication ;
- **o** lancée lorsque l'opérateur pressera la touche 0 dans le VoiceMail.

```
exten => s,1,SetGlobalVar(sounds_path=/var/asterisk/sounds/)
```

L'utilisation de variables globales peut simplifier la configuration, elles peuvent être définies afin d'être réutilisable dans la configuration. On définit alors la variable contenant le chemin d'accès aux fichiers sons sur le serveur.

```
exten => s,2,Background(${sounds_path}welcome)
```

La fonction *Background()* va permettre de jouer en tâche de fond le message de bienvenue *welcome.gsm* présent dans le répertoire des sons. On notera qu'il **ne faut pas** spécifier l'extension du fichier.

```
exten => #,1,Goto(menu,s,1)
```

L'extension # va se déclencher lorsque l'utilisateur pressera la touche #, il pourra être invité à cela dans le message de bienvenue précédent. Cette action redirigera l'appel vers le menu principal (*contexte : menu*).

```
exten => i,1,Playback(${sounds_path}erreur-saisie)
```

A l'extension i pour l'extension invalide (cf. extensions prédéfinies), on attribue la fonction *Playback()* qui va jouer elle aussi un son pour avertir l'utilisateur qu'il c'est trompé lors de la saisie.

Remarque : contrairement à la fonction *Background()*, la fonction *Playback()* ne rendra pas la main à l'utilisateur avant la fin de la lecture.

```
exten => t,1,Goto(accueil,s,1)
```

Pour terminer on va rediriger à la fin du timeout vers le même contexte, ainsi l'utilisateur réentendra le menu des options jusqu'à ce qu'il entre une option valide.

Contexte de menu :

- *Source*

```
[menu] ; définition d'un contexte pour le menu
exten => s,1,Background(${sounds_path}menu)
exten => 0,1,Goto(menu,s,1)
exten => 1,1,Goto(appel,s,1)
```

```
exten => 2,1,Goto(message,s,1)
exten => 3,1,Goto(support,s,1)
exten => i,1,Playback(${sounds_path}erreur-saisie)
exten => i,2,Goto(menu,s,1)
exten => t,1,Goto(menu,s,1)
```

- *Détails*

```
exten => s,1,Background(${sounds_path}menu)
```

Comme pour le message de bienvenue, il sera joué à l'utilisateur le fichier son du menu pour l'inviter à sélectionner un service.

```
exten => 0,1,Goto(menu,s,1)
exten => 1,1,Goto(appel,s,1)
exten => 2,1,Goto(message,s,1)
exten => 3,1,Goto(support,s,1)
```

Il faut ensuite définir les différentes extensions de redirection vers les autres services. Une seule priorité est nécessaire pour ces actions car elles ne s'occuperont que de rediriger vers d'autres contextes.

Contexte d'appel :

- *Source*

```
[appel] ; définition d'un contexte pour le menu d'appel
exten => s,1,Background(${sounds_path}appel)
exten => 0,1,Goto(menu,s,1)
exten => _1XXX,1,Dial(SIP/${EXTEN},20,tr)
exten => i,1,Playback(${sounds_path}erreur-saisie)
exten => t,1,Goto(appel,s,1)
```

- *Détails*

Mis à part les éléments récurrents tels que les messages de menu, d'erreurs de saisie et de redirection. Le service fourni par ce contexte est un service de redirection d'appel. On invite donc l'utilisateur à composer le numéro du correspondant à joindre.

```
exten => _1XXX,1,Dial(SIP/${EXTEN},20,tr)
```

Le numéro d'extension `_1XXX` va cibler toutes les extensions commençant par 1 et comportant 4 chiffres. Une fois le numéro entrée, via la fonction `Dial()` le correspondant sera joint.

Le protocole utilisé sera le protocole SIP et la cible la variable globale `${EXTEN}` contenant le numéro composé par l'utilisateur. Le second paramètre sera le timeout, temps au bout duquel la tentative de connexion sera annulée si l'appel n'a pas été prit.

Options :

- **t** autoriser l'appelé à transférer l'appel ;
- **r** générer une tonalité pour l'appelant.

Contexte de messagerie :

- *Source*

```
[message] ; définition d'un contexte pour la messagerie
exten => s,1,VoiceMailMain(${CALLERIDNUM})
exten => t,1,Hangup()
```

- *Détails*

```
exten => s,1,VoiceMailMain(${CALLERIDNUM})
```

L'utilisateur ici sera redirigé vers le module de serveur de messagerie vocale `VoiceMailMain`. Il pourra consulter et gérer ses messages. Le paramètre `${CALLERIDNUM}` va servir à pointer au lancement du module sur la messagerie de l'appelant sans qu'il ait à entrer son identifiant.

```
exten => t,1,Hangup()
```

Cette ligne va être utile au cas où il ne serait pas possible de joindre le `VoiceMail`. Arrivé au bout du timeout, la communication sera terminée par la fonction `Hangup()`.

Contexte de support :

- *Source*

```
[support] ; définition d'un contexte pour le support
exten => s,1,GoToIfTime(09:00-17:00|mon-fri|*|*?s,4)
exten => s,2,Playback(${sounds_path}no-relation-support)
exten => s,3,Goto(menu,s,1)
exten => s,4,Playback(${sounds_path}relation-support)
exten => s,5,Queue(default)
exten => t,1,Hangup()
```

- *Détails*

```
exten => s,1,GoToIfTime(09:00-17:00|mon-fri|*|*?s,4)
```

La première priorité du contexte utilise une fonction `GoToIfTime()` similaire à la fonction `Goto()` à la différence que l'on peut spécifier une plage horaire. Si la condition est remplie le `Goto()` est exécuté, sinon on passe à la priorité suivante.

- **09:00-17:00** de 09h00 à 17h00 ;
- **mon-fri** du lundi au vendredi ;
- ***** tous les jours du mois ;
- ***** tous les mois ;
- **s,4** extension **s** priorité **4** du contexte en cours.

Dans le cas présent si la condition est validée, le message de mise en relation avec le support technique sera diffusé.

```
exten => s,5,Queue(default)
```

La fonction `Queue()` permet de diriger l'appel vers une file d'attente, ici vers la file d'attente nommé `default`.

4. LIENS UTILES

<http://www.asterisk.org/> (site officiel)

<http://www.asteriskguru.com/>

<http://voip-info.org/>

<http://www.cis-consultants.com/>