

# Intégration Voix / Données

## TP 3 - Voix sur IP

### Objectifs du TP

- Comprendre le fonctionnement d'une architecture de Voix sur IP
- Implanter les solutions techniques découvertes en cours et TP (SIP, QoS)
- Observer les performances lors d'appels.
- Comprendre les paramètres influant sur la Voix et savoir les estimer.

### Configuration

#### Configuration matérielle

- 2 ou 3 stations Linux
- 1 switch
- éventuellement SIP-phones

#### Configuration logicielle

- Wireshark
- Softphone
- Asterisk
- Utilitaires réseaux habituels (ifconfig, ping...)

### Consignes et Sujet

- Consignes :  
*Au delà de la configuration de serveurs SIP, il convient aussi d'illustrer les concepts présentés en cours.  
 Pour cela, de fréquentes observations via Wireshark ou les logs des serveurs sont fondamentales.*

*Ce sujet est organisé en tutoriel et la configuration est à rendre (voir fin du sujet). Toutefois, cette organisation ne dispense pas d'observer le fonctionnement (protocoles) et les différents paramètres qualitatifs pouvant entre en jeu (codec, débit).*

### I. Rappels sur les IP-PBX et Asterisk

Quelques rappels sur les avantages des solutions de VoIP :

- Un IP-PBX possède les services d'un standard téléphonique classique (PABX);
- Permet de consolider les réseaux d'infrastructure (données et téléphone);
- Peu coûteux en terme de matériel comparativement aux solutions dédiées analogiques;
- Corolaire : La limite du nombre de communications est directement liée à la puissance de traitement des serveurs IP-PBX;
- Suivant les serveurs (et via des cartes d'extensions), il est possible d'interconnecter une structure analogique existante à une infrastructure numérique;
- Dernier avantage, actuellement, les communications véhiculées sur les réseaux IP sont soumises au même tarif que les données.

Dans le cadre de ces TP, notre choix s'est porté sur l'IP-PBX Asterisk car :

1. Il est le plus répandu;
2. Open Source;
3. Gratuit;
4. Peut fonctionner sur de nombreux matériels et OS;
5. Respecte les standards les plus répandus.
6. Facilement extensible via de nombreux modules disponibles.

### II. Téléchargement des Softphones, d'Asterisk et installation

Ce TP suppose l'utilisation de softphones (ou de SIP-phones) et de serveurs IP-PBX fonctionnant sous Asterisk.

## II.I. Softphones

Nous utiliserons les softphones Twinkle.

Leur installation sous Ubuntu en salle de TP s'effectue via `apt-get install twinkl`

Son lancement s'effectue via : `twinkl`


## II.II.Asterisk


Nous utiliserons l'IP-PBX Asterisk. Il peut être obtenu en téléchargement (version 1.6) à l'URL suivante : <http://www.asterisk.org/downloads> ou via `wget`.

L'installation d'Asterisk peut requérir sa compilation au préalable :

```
make menuselect
./configure
make
make install
```

En salle de TP, le téléchargement et l'installation peuvent s'effectuer via `apt-get install asterisk`

 **Note :** Asterisk s'installe par défaut dans `/usr/sbin/`

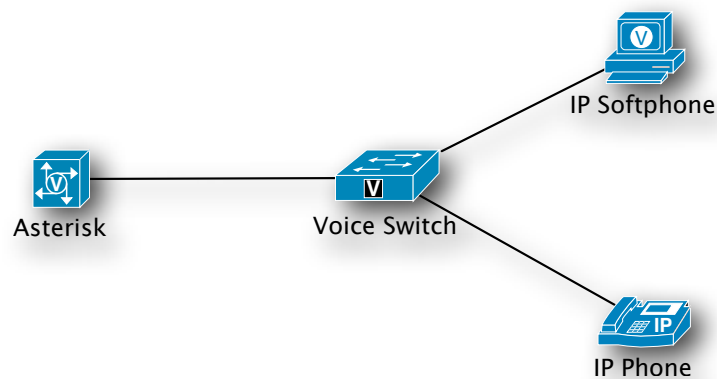
 **Attention au déploiement :**

La charge d'un système fonctionnant sous le serveur Asterisk n'est pas infinie ! Il faut prendre garde à cet aspect pour le déploiement de solutions VoIP. Par ailleurs, cette charge est plus limitée par le nombre de conversations simultanées que par le nombre d'utilisateur !

## III. Réalisation de l'architecture d'interconnexion

Cette première architecture vise à déployer pour chaque groupe de TP un site spécifique VoIP. Cette architecture suppose que chaque groupe dispose d'un switch, de 3 stations *linux* éventuellement complétées par ses *SIP-Phones*. Dans la suite de l'exercice, seuls les réglages d'un seul groupe de TP seront décrits.

Dans ce premier TP, nous proposons la réalisation d'une architecture d'interconnexion basique dans laquelle les stations sont interconnectées via un switch. Nous supposons que ce switch forme le *backbone* de l'entreprise. Tous les autres équipements (serveur Asterisk, téléphones IP) seront reliés à ce switch.



## IV. Configuration d'Asterisk

### IV.1.Principe d'administration d'Asterisk

Le lancement du serveur Asterisk est le suivant : `asterisk`

Voici une liste d'options utilisables :

- `-c`. Mode console. Asterisk sera lancé dans un processus utilisateur et lance une IHM textuelle. Ce mode est très utile pour le dépannage ou la configuration initiale d'Asterisk.
- `-v`. Mode verbeux. Cette option est à combiner au mode client (`-c`). La verbosité d'Asterisk est réglable de `-v` à `-vvv`.
- `-r`. Mode *Remote*. Permet d'associer une IHM textuelle à un démon Asterisk fonctionnant déjà.
- `-x "<commande>"`. Exécution. Utilisé en combinaison avec l'option `-r`, cette toption permet d'exécuter une commande sur le serveur Asterisk sans devoir lancer explicitement un client et taper manuellement la commande.

- ➔ Télécharger Asterisk, et l'installer.
- ➔ Mettre en place Wireshark sur le serveur Asterisk ainsi que sur les clients SIP afin d'observer les échanges SIP (et voix).
- ➔ Lancer Asterisk en mode console très très très verbeux.

☑ **Note :** Vous trouverez à la fin de ce sujet quelques commandes utiles dans l'IHM textuelle Asterisk.

☑ **Note :** Un pack de langue française est disponible via `apt-get install asterisk-prompt-fr-proformatique`

## IV.2.Principe de configuration d'Asterisk

Hors IHM textuelle, Asterisk peut se configurer via les fichiers de configuration suivants :

- `zaptel.conf`. Ce fichier concerne les relations entre Asterisk et la configuration bas niveau des interfaces téléphoniques.
- `zapdata.conf`. Ce fichier concerne les relations entre Asterisk et la configuration des interfaces téléphoniques.
- `extensions.conf`. Ce fichier concentre les plans de numérotation pré-configurés.
- `sip.conf`. Ce fichier concerne la configuration SIP.
- `iax.conf`. Ce fichier concerne les canaux IAX (*Inter Asterisk eXchange*).

☑ **Note :** L'installation d'Asterisk par défaut crée ces fichiers et les alimente de données d'exemples. Nous supposons dans la suite que ces fichiers sont vides.

## IV.3.Le fichier sip.conf

Ce fichier contient au moins la partie *general* (`[general]`). Cette partie regroupe les options applicables à tous les clients SIP.


Pour chaque SIP-phone, il faut lui dédier une partie. Le nom de cette partie est choisi librement mais doit de préférence être unique. En effet, dans le plan de numérotation, ce client SIP sera identifié par identificateur.

Les options se notent sous la forme suivante : `nom_option=valeur`

Les options et valeurs suivantes sont utilisables :

- `type=friend | user | peer`
  - Indique si le client SIP peut émettre et recevoir des appels (*friend*) ou uniquement en recevoir (*user*) ou permettre une liaison entre deux clients seulement (*peer*).
- `context=xxx`
  - Indique le contexte d'application du client (voir partie traitant du fichier `extensions.conf`)
- `host=@IP | dynamic`
  - Indique à Asterisk à quelle adresse (IP) est joignable le client SIP (par exemple pour lui passer un appel). On peut renseigner une IP fixe ou indiquer la valeur *dynamic*. Le client SIP indiquera alors à Asterisk, lors de son inscription (*REGISTER*), où le joindre.
- `defaultuser=nom_utilisateur`
  - Permet de spécifier le *login* de l'utilisateur pour authentifier le client SIP. Si ce paramètre est omis, le nom de la partie entre `[]` sera utilisé comme login
- `secret=mon_code_secret_rien_qu_a_moi`
  - Permet de spécifier le mot de passe à employer pour authentifier le client SIP.
- `callerid=nom_entre_guillemets numero_entre_<>`
  - Indique le nom et le numéro de l'utilisateur pour la fonction de présentation du nom et numéro.
- `language=code_pays`
  - Permet de spécifier la langue parlée par le client (`fr` pour le Français). Si les fichiers sons de langue sont présents, Asterisk utilisera alors les fichiers de la langue précisée ici plutôt que les fichiers son de langue Anglaise.
- `allow=codec`
  - Permet de spécifier les codec acceptés par ce client SIP. On peut citer (non exhaustif) : `ulaw, gsm, alaw, adpcm, g723...` Chaque codec suppose une autorisation exclusive dans la spécification (1 codec autorisé = 1 ligne, 2 codec = 2 lignes...)
- `disallow=codec`
  - Permet de spécifier les codec interdits par ce client SIP. L'étendue des codec est identique à la propriété précédente.
- `nat=yes | no`
  - Permet de spécifier si le client est séparé du serveur Asterisk par un équipement déployant les fonctions de NAT.

- `canreinvite=yes` | `no`
  - Permet à Asterisk de récupérer plus d'informations sur l'appelant. Doit être positionné à `no` en cas de NAT.

 **Note :** Attention aux mots de passe : Si l'utilisateur doit s'authentifier via un code PIN, un mot de passe alphanumérique risque d'être difficile à saisir...

→ Editer le fichier `sip.conf` pour ajouter la description de vos deux SIP-phones afin de leur permettre d'émettre et de recevoir des appels.

#### IV.4. Configuration des softphones Twinkle

La configuration d'un client SIP demande les informations suivantes : nom du compte SIP, mot de passe associé, localisation du serveur IP-PBX.

Pour utiliser Twinkle, il faut créer un profil utilisateur et vérifier que les entrées/sorties son sont correctement configurées.

#### IV.5. Le fichier `extensions.conf`

Ce fichier concerne le plan de numérotation (*dialplan*). Il regroupe au moins les parties suivantes : *globals*, *general*, *default*. La configuration de plan de numérotation est de fonctionnement analogue à celle utilisée dans le monde PABX.

Les sections *general* et *global* servent à la configuration globale du serveur indépendamment des contextes définis par la suite. La section *general* s'intéresse à la configuration tandis que la section *global* vise les définitions de variables globales. Certaines options sont souhaitables dans la section *general* dont :


`autofallthrough=yes` pour indiquer au serveur de raccrocher une fois les différents traitements effectués.


Le *dialplan* Asterisk s'architecture autour de 4 activités : les contextes, les extensions, les applications et les priorités.

- **contextes** : Les plan de numérotation étant souvent très complexes, ils sont séparés en plusieurs sections, nommées contextes dans Asterisk. Ce système permet une isolation (sauf collaboration explicite) entre les contexte et évite les interférences de configuration par exemple. Les contextes sont notés de la manière suivante : `[nom_contexte]`
- **extensions** : Une extension définit une série d'étape dans le rendu d'un service lors d'un appel. Par exemple, une extension que tel numéro renvoie (*extension*) vers : une horloge parlante (*application*) suivie de la messagerie vocale (*application*).  
Une extension se décrit par : `exten => numero_extension, priorité, application`
  - `numero_extension`. Ce numéro représente la destination de l'appel. Par exemple, pour un contexte donné, le numéro 123 représente un «appel» vers «123». Certaines valeurs particulières sont à prendre en compte :
    - `s` (*start*) : indique un traitement indifférent du numéro composé.
    - `t` (*timeout*) : indique une temporisation écoulée. Par exemple lorsque le serveur attend qu'un client indique un numéro
    - `i` (*invalid*) : indique une entrée non référencée dans le plan de numérotation.

 **Note :** `s` et `i` supposent que l'utilisateur n'a pas entré de numéro d'extension. Cas de filtrage par exemple.

- **applications** : Il s'agit du coeur du plan de numérotation. Une application reflète le comportement d'Asterisk en réponse à un contexte. Différentes applications existent et peuvent être ajoutées. Les applications les plus usitées sont l'appel `Dial()`, décrocher `Answer()` et raccrocher `Hangup()`.

 **Note :** Les différents paramètres d'une application sont séparés par une virgule.

 **Note :** Si une application emploie plusieurs paramètres et que l'on souhaite en omettre un ou plusieurs, les virgules séparant les paramètres sont obligatoires. Par exemple : `App(p1,p2,p3)` ou l'on souhaite omettre `p1` et `p2` donne `App(, ,p3)`.

- **priorités** : Lorsque plusieurs traitements sont à effectuer pour une même extension, ils doivent être ordonnées par des priorités. 3 types de priorités sont utilisables :
  - Les priorités explicites. Il s'agit de numéroté séquentiellement les extensions. Exemple : 1, 2, 3...
  - Les priorités implicites. Ce type de priorité est utile lorsqu'une extension possède nombre d'étapes. Si l'on souhaite rajouter une étape, il faut alors modifier toutes les lignes suivantes pour rétablir l'ordre. Les priorités implicites utilisent l'ordre de lecture. Elles se traduisent par `n`. Il est à noter que les priorités implicites ne peuvent être utilisées que dans le cas où une extension possède plusieurs traitements ET que le premier traitement aura une priorité explicite égale à 1.
  - Les priorités nommées. Asterisk autorise l'utilisation de labels adossés aux priorités. Ces priorités sont représentées sous la forme `X(label)` où `X` est une priorité explicite ou implicite et `label` le nom de la priorité choisi.


 **Note :** Un erreur fréquente consiste à «oublier» certains numéros. Dans ce cas, le serveur Asterisk ne peut continuer le traitement. Ex: 1, 3.

## IV.6. Les services essentiels

### a) Décrocher et Raccrocher

Ces deux applications de base ne requièrent aucun argument. Décrocher s'effectue via l'appel à `Answer ()` et raccrocher via `Hangup ()`

 **Note :** Toutes les applications d'Asterisk ne décrochent pas forcément, Il faut alors faire appel à `Answer ()`.

 **Note :** Selon la configuration par défaut d'Asterisk (`autofallthrough`), il faut prendre garde à raccrocher...

### b) Répéter

Cette fonction permet à Asterisk de lire un fichier son préalablement enregistré. Deux applications permettent un tel traitement :

- `Playback(chemin/ressource)` où chemin peut être absolu (format UNIX) ou relatif au répertoire de sons d'Asterisk. La lecture ne peut être interrompue par la pression d'une touche.
- `Background(chemin/ressource)` où chemin peut être absolu (format UNIX) ou relatif au répertoire de sons d'Asterisk. La lecture peut être interrompue par la pression d'une touche. Plus de détails au §IV.6.g).iv).

### c) Appeler

L'appel se fait au travers de l'application `Dial ()` qui s'organise de la manière suivante :

```
Dial (technologie/utilisateur[:mdp]@hôte_distant[:port] [/remote_extension], time_out, option )
```

où : Technologie indique à Asterisk quel moyen technique utiliser pour joindre le récepteur. Par exemple : SIP

extension\_distante représente la ressource à joindre. Par exemple, un utilisateur.

utilisateur[:mdp]@hôte\_distant[:port] dénote un autre serveur Asterisk.

time\_out indique la durée en secondes avant que Asterisk passe à la priorité suivante en cas de non réponse.

option peut être T, t, k, K et/ou m. t indique un transfert d'appel possible de l'appelé; T un transfert d'appel possible de l'appelant; k une mise en attente possible de l'appelé, K une mise en attente possible de l'appelant et m une musique d'attente.

➔ Editer le fichier `extensions.conf` pour :

1. Décrocher, répéter «1» et raccrocher lorsque le premier SIP-phone contacte Asterisk quelle que soit sa demande.
2. Permettre une communication téléphonique entre vos deux SIP-phones.

➔ Configurer les SIP-phones et tester !

➔ Observer les échanges réseau réalisés pour chaque client et pour le serveur Asterisk. Observer les échanges protocolaires SIP.

## IV.7. Déploiement de services IP-PBX

De nombreux services (et combinaison de services) sont disponibles dans Asterisk soit directement intégrés, soit des plugins à ajouter (IHM graphique, contrôle à distance, nouvelles fonctionnalités...). Dans le cadre des TP, nous nous limiterons aux principales.

### a) Messagerie vocale (*voicemail*)

La configuration des boîtes vocales s'effectue dans le fichier `voicemail.conf`. Ce fichier s'organise en contextes comme la plupart des fichiers de configuration d'Asterisk. La configuration de la messagerie vocale suppose deux étapes : La création de boîtes vocales et l'intégration de la messagerie dans le plan de numérotation.

#### (i) Création des boîtes vocales

La création de boîtes vocales obéit à la configuration suivante :

```
nom_boite => mdp,nom_titulaire[,email[,email_mobile[,options]]]
```

où : nom\_boite désigne le nom de la boîte vocale.

mdp et nom\_titulaire sont respectivement le mot de passe et le nom du titulaire de la boîte vocale.

email et email\_mobile font référence à l'email de l'utilisateur éventuellement secondé par une seconde adresse. Ces adresse peuvent être utilisées afin qu'Asterisk envoie par mail les messages de la boîte.

options concerne une liste d'options à choisir parmi les suivantes. Chaque option s'ajoute selon le format option=valeur et sont séparés par le caractère |

- attach=yes | no : indique que les messages devront être relayés par email avec le fichier son en attachement. Par défaut fixé à no.
- review=yes | no : indique que l'appelant peut laisser des messages vocaux déjà écoutés dans sa boîte. Par défaut fixé à no

➔ Editer le fichier voicemail.conf pour ajouter à chaque utilisateur sa propre boîte vocale

### (ii) Intégration dans le plan de numérotation de la messagerie

La messagerie vocale est une application nommée VoiceMail et s'utilise de la manière suivante :

VoiceMail (utilisateur@contexte\_voicemail,type\_accueil)

où : utilisateur est le nom de l'utilisateur propriétaire de la boîte vocale

contexte est le contexte où se situe la boîte vocale

type\_accueil est une valeur au choix entre u et b. u correspond à un message en cas d'absence et b en cas d'occupation (déjà en ligne par exemple).

➔ Modifier le fichier extensions.conf pour prendre en compte la messagerie vocale dans le plan de numérotation.

### (iii) Intégration dans le plan de numérotation de l'interrogation de la messagerie

L'accès à la messagerie s'effectue via l'application VoiceMailMain (utilisateur@contexte\_voicemail).

Cette application met en place un menu dont les possibilités sont les suivantes (la configuration par défaut d'Asterisk étant en langue anglaise, le menu suivant est présenté dans cette langue) :

```
1 Old Messages
  3 Advanced options
    1 Send reply
    2 Call back
    3 Envelope
    4 Outgoing call
    5 Leave message
    * Return to main menu
  4 Play previous message
  5 Repeat current message
  6 Play next message
  7 Delete current message
  8 Forward message to another mailbox
  9 Save message in a folder
  * Help; during msg playback: Rewind
  # Exit; during msg playback: Fastforward
2 Change folders
3 Advanced options
0 Mailbox options
  1 Record your unavailable message
  2 Record your busy message
  3 Record your name
  4 Change your password
  * Return to the main menu
* Help
# Exit
```

➔ Modifier le fichier extensions.conf afin que chaque utilisateur puisse interroger sa boîte vocale.

**(iv) Liaison SIP - messagerie vocale**

Afin que les utilisateurs soient prévenus de l'arrivée d'un nouveau message, il est possible de lier un compte SIP avec une boîte vocale. Pour cela, il suffit de rajouter dans le contexte de l'utilisateur choisi (dans le fichier `sip.conf`) l'identifiant de sa boîte vocale. Cette déclaration sera de la forme : `mailbox=nom_boite@contexte`

➔ Modifier le fichier `sip.conf` pour finir de prendre en compte la messagerie vocale pour chaque utilisateur

**b) Horloge parlante**

L'application concernée est `SayUnixTime(time_unix, fuseau_horaire, parametres)`

où `parametres` sont construits à partir de (liste non exhaustive) :

A ou a : jour de la semaine  
B, b ou h : nom du mois  
d ou e : jour du mois  
Y : année  
k : heures au format 24h  
M : minutes

➔ Modifier le fichier `extensions.conf` pour ajouter ce service

**c) Suivi d'appels (facturation)**

Si la facturation n'est pas intégrée telle que dans Asterisk, un module de log d'appel est néanmoins présent. Nous allons étudier comment fonctionne ce module. La configuration du module CDR pour *Call Detail Recording*, s'effectue via l'intermédiaire du fichier `cdr.conf`

Comme les autres fichiers de configuration, ce fichier dispose d'une section `general`. La seule entrée qui nous intéresse est la suivante :

`enable = yes|no` qui permet d'activer ou non la fonction de log. (Yes par défaut)

Une fois la fonction activée, Asterisk remplit un fichier CSV placée généralement dans le dossier suivant `/var/log/asterisk/cdr-csv/`

Chaque entrée est séparée par une virgule. Parmi les entrées *logguées* on peut trouver :

<code>accountcode</code> : compte (si positionné dans les fichiers <code>.conf</code> comme <code>SIP.conf</code> via l'option <code>accountcode=valeur</code> )	<code>dstchannel</code> : canal de destination employé,
<code>src</code> : ID du destinataire,	<code>lastapp</code> : dernière application appelée,
<code>dst</code> : extension destination,	<code>start</code> : date et heure de l'appel,
<code>dstcontext</code> : contexte de destination,	<code>answer</code> : date et heure de la réponse,
<code>clid</code> : ID de l'appelant,	<code>end</code> : date et heure de fin de l'appel,
<code>channel</code> : canal utilisé,	<code>duration</code> : durée de l'appel en secondes,
	<code>billsec</code> : durée de l'appel en secondes,
	<code>disposition</code> : <i>flags</i> de l'appel (ANSWERED, BUSY, NO ANSWERED).

☒ **Note :** `duration` et `billsec` sont différents ! `billsec` mesure le temps de connexion établie tandis que `duration` mesure le temps d'utilisation de la ligne de l'appelant (durée pour « composer le numéro » compris).

➔ Observer le fonctionnement des logs Asterisk et déduire le nombre d'appels effectués pour un utilisateur particulier.

**d) Conférences téléphoniques**

Asterisk permet l'utilisation de salles de conférences téléphoniques. Pour fonctionner, le fichier `meetme.conf` doit être configuré. Dans celui-ci il faut indiquer les numéros des salles de conférence virtuelles en respectant le format suivant : `conf => numero_salle,mdp,mdp_admin`

où : `numero_salle` est le numéro de la salle de conférence virtuelle ;  
`mdp` est le mot de passe pour accéder à la salle de conférence ;  
`mdp_admin` est le mot de passe de l'administrateur de la salle.

L'accès aux salles de conférence s'effectue au travers de l'application `ConfBridge`.

`ConfBridge(num_conf, options, mdp)` où `num_conf` est le numéro de la salle de conférence et `mdp` est le mot de passe pour accéder à la salle.

Des options disponibles sont :

- a : Mode administrateur
- c : Énonce le nombre d'utilisateurs présents dans la salle lorsqu'un utilisateur entre dans la salle.
- s : Énonce le menu utilisateur ou administrateur lorsque la touche \* est pressée
- w : Patienté jusqu'à ce qu'un utilisateur particulier entre dans la salle.

➔ Modifier les fichiers `meetme.conf` et `extensions.conf` pour ajouter ce service

## e) Mise en attente et transfert d'appels

Les transferts d'appels s'effectuent par appel à l'application `Transfer(destination)`. Cette application s'emploie comme une version simplifiée de l'application `Dial`. Il suffit en effet de préciser l'identifiant de l'utilisateur à qui transmettre l'appel (complété de la technologie à employer en cas de changement).

La mise en attente (fonction de parage) permet de place un appel en attente dans un « emplacement de parking » afin de pouvoir le récupérer d'une autre extension. Cette fonctionnalité nécessite la constitution d'un plan de parage et se contrôle via le fichier `features.conf`

La section `general` du fichier `features.conf` contient les informations suivantes :

- `parkext`. Il s'agit du numéro de « l'emplacement de parking ». Tout appel en attente doit être transféré vers ce numéro d'extension (700 par défaut), Asterisk se chargeant alors d'indiquer quel est le numéro d'emplacement effectif.
- `parkpos`. Cette option définit le nombre d'appels pouvant être mis en attente (« nombre de places de parking »). Par exemple : 701-740 pour définir 40 positions.
- `context`. Nom du contexte du parking dans le plan de numérotation.
- `parkingtime`. Indique la durée maximale (en secondes) des mises en attente.

La configuration du fichier `features.conf` demande l'introduction du plan de parage dans le plan de numérotation :

```
[contexte_d_entree] include => parkedcalls
```

L'utilisation de la mise en attente s'effectue de la manière suivante. Lors d'un appel, il suffit de transférer l'appel vers le numéro 700 (par défaut). Lors du transfert, Asterisk annonce le numéro de parage. Le correspondant est alors mis en attente. Il est possible alors de reprendre l'appel mis en attente par simple appel au numéro de parage communiqué par le serveur.

☒ **Note :** Asterisk ne lit le fichier `features.conf` qu'une seule fois au démarrage. Toute modification de ce fichier impose donc de redémarrer le service Asterisk.

☒ **Note :** Il convient de porter attention à la configuration des mises en attente (musiques d'attente) dans le fichier `musiconhold.conf`. De plus, si les fichiers de musique d'attente ne sont pas présents, il faudra en télécharger.

☒ **Attention :** le parage d'appel impose des transferts d'appels. De fait, les options `t` et/ou `T` doivent être présentes lors de l'appel à l'application `Dial` afin de les autoriser.

➔ Modifier le fichier `extensions.conf` pour ajouter ces deux services.

## f) Fonctionnalités avancées

### (i) Filtrage dans les extensions

Le filtrage consiste à utiliser des jokers (comme \* ou ? pour UNIX) afin de remplacer certains caractères. Toute extension filtrée commence par le caractère `_` (*underscore*).

Les filtres utilisables sont :

- `X` : permet de remplacer un chiffre dans l'extension
- `Z` : idem `X` mais ne remplace pas le 0 non plus
- `N` : idem `Z` mais ne remplace pas le 1 non plus
- `.` : permet de remplacer n'importe quel caractère et par extension n'importe quelle série de caractères.

Asterisk permet le filtrage d'extension par l'identifiant de l'appelant (*Caller-Id*). Pour cela, il faut ajouter le caractère `/` suivi du numéro (éventuellement filtré aussi) de l'appelant. Exemple : `exten => 123/999,1,application()`

Ici, lors de l'appel à l'extension 123, Asterisk sélectionnera l'extension uniquement si l'identifiant de l'appelant est 999.



➔ Modifier le fichier `extensions.conf` pour faire en sorte que tout appel dont le numéro commence par 123 permette d'accéder à l'horloge parlante.


## (ii) Utilisation de variables

La section `global` du fichier `extensions.conf` est à utiliser. Une variable se positionne de la manière suivante : `nom_variable => valeur`

Leur utilisation se fait sous la forme `${nom_variable}`

Les valeurs suivantes sont préconfigurées (liste non exhaustive) :

- `CALLERID(id)` où `id` peut être remplacé par `all`, `name` ou `num`. Cette variable indique respectivement le nom et le numéro de l'appelant, uniquement son nom, uniquement son numéro.
- `EXTEN` représente l'identifiant de l'extension courante.

 **Attention :** le parage d'appel impose des transferts d'appels. De fait, les options `t` et/ou `T` doivent être présentes lors de l'appel à l'application `Dial` afin de les autoriser.

## (iii) Redirection d'extension


Asterisk permet l'utilisation de primitive de logique de programmation dans la gestion des extensions. Ainsi les applications `Goto()` et `GotoIf()` permettent-elles d'effectuer des branchements conditionnels ou non entre extensions.


`Goto(id)`

où : `id` représente un contexte + une extension + une priorité. Ces éléments sont séparés par une virgule. Si le contexte et l'extension sont omis, Asterisk considère le contexte et l'extension courante.

`GotoIf(condition ? id_si_vrai : id_si_faux)`

où : Lorsque la condition est évaluée, Asterisk branche vers `id_si_vrai` si la condition est respectée et vers `id_si_faux` sinon. Les `id` peuvent être de la forme : `label, priorité, extension-virgule-priorité` ou `contexte-virgule-extension-virgule-priorité`  
`condition` se représente sous la forme suivante : `[$valeur_x = valeur_y]` où `valeur_x` et `valeur_y` peuvent être des variables.

 **Attention :** les comparaisons s'effectuent sur une base de chaînes de caractères. Les valeurs et variables doivent donc être encadrées de `"` et `"`.

 **Exemple :** `GotoIf("${variable1}" = "valeur")? id_si_vrai : id_si_faux)`

## (iv) Saisie d'informations

La saisie d'informations est essentielle pour un fonctionnement interactif d'un PBX. Asterisk propose aux administrateurs plusieurs moyens de récupérer les informations saisies par un utilisateur.

Afin de saisir un numéro d'extension, il est possible d'employer les applications `Background` et `Waittexten`.

`Background(chemin/ressource)` permet la saisie de numéros d'extensions pendant qu'un message d'accueil est diffusé.

`Waittexten(nb)`. Cette application demande à Asterisk d'attendre au plus `nb` secondes avant de considérer la saisie de l'utilisateur comme complète. `Waittexten` est souvent employée en complément de l'application `Background`.

Quelle que soit la solution choisie, le branchement vers l'extension est automatique.

 **Note :** Il convient de porter attention au fait que plusieurs extensions commençant par le même chiffre peuvent être confondues si l'utilisateur ne les saisit pas assez rapidement.

Si le contenu d'une variable doit être récupéré alors l'application `Read` peut s'en charger. L'appel à l'application s'effectue via :

`Read(nom_variable, message_annonce, limite_saisie)`

où : `nom_variable` est le nom de la variable qui va stocker l'information numérique,  
`message_annonce` est le message qui sera diffusé avant la saisie,  
`limite_saisie` représente le nombre de caractères à prendre en compte. La touche `#` validant la saisie.

➔ Réaliser un standard téléphonique interactif permettant à un utilisateur de choisir les services précédemment déployés.

➔ Modifier ce standard pour faire en sorte que le menu soit différent suivant deux utilisateurs différents prédéfinis.

## V. Exercice

Réaliser un standard téléphonique possédant trois utilisateurs. Les fonctionnalités disponibles doivent être :

- Les appels entre utilisateurs,
- les transferts d'appel et mises en attente,
- les conférences téléphoniques,
- la messagerie vocale,

Par simple appel au 222, l'utilisateur doit pouvoir accéder à un menu lui permettant de choisir les actions disponibles.

De plus, tout appel qui ne peut aboutir (en cas d'indisponibilité ou d'occupation) doit être pris par la messagerie vocale.

Le numéro 333 doit permettre de faire sonner tous les téléphones (autre que l'appelant).

Tout appel vers l'international doit être interdit.

## VI. Commandes CLI utiles :

### VI.1. Commandes Asterisk générales

<code>!</code>	Lancer une commande shell
<code>cdr status</code>	Affiche le statut du module CDR
<code>feature show</code>	liste les fonctionnalités configurées
<code>help</code>	Affiche l'aide générale ou l'aide spécifique à une commande
<code>originate</code>	Débuter un appel
<code>reload</code>	Recharge le fichiers de configuration d'Asterisk
<code>restart now</code>	Redémarre Asterisk <u>immédiatement</u>
<code>restart when convenient</code>	Redémarre Asterisk <u>lorsque le serveur sera inactif</u>
<code>stop now</code>	Stoppe Asterisk <u>immédiatement</u>
<code>stop when convenient</code>	Stoppe Asterisk <u>lorsque le serveur sera inactif</u>

### VI.2. Commandes spécifiques au plan de numérotation

<code>dialplan add extension</code>	Ajoute une nouvelle extension au contexte
<code>dialplan add include</code>	Inclut un contexte dans un autre contexte
<code>dialplan reload</code>	Demande à Asterisk de recharger les extensions -Utile en cas de modif. du fichier extensions.conf lorsque le serveur fonctionne
<code>dialplan remove extension</code>	Retire une extension spécifique
<code>dialplan save</code>	Sauvegarder les modifications du plan de numérotation
<code>dialplan show</code>	Afficher le plan de numérotation

### VI.3. Commandes spécifiques à SIP

<code>sip history</code>	Active l'historique SIP
<code>sip history off</code>	Désactive l'historique SIP
<code>sip notify</code>	Envoi d'une notification (paquet <i>notify</i> ) à un pair SIP
<code>sip reload</code>	Demande à Asterisk de recharger les configurations SIP - Utile en cas de modifications du fichier sip.conf lorsque le serveur fonctionne
<code>sip show channels</code>	Liste les canaux SIP actifs
<code>sip show channel x</code>	Affiche les informations détaillées d'un canal SIP particulier
<code>sip show domains</code>	Liste les domaines SIP locaux
<code>sip show history</code>	Affiche l'historique SIP
<code>sip show peers</code>	Liste les pairs SIP
<code>sip show peer x</code>	Affiche les informations détaillées d'un pair SIP particulier
<code>sip show settings</code>	Affiche les informations générales SIP
<code>sip show users</code>	Liste les utilisateurs SIP
<code>sip show user x</code>	Affiche les informations détaillées d'un utilisateur SIP particulier

#### VI.4. Commandes spécifiques à la messagerie vocale

```
voicemail show users
```

Liste les boîtes vocales définies

```
voicemail show users for x
```

Liste les boîtes vocales définies pour un contexte particulier

#### VI.5. Autres commandes

```
database put blacklist nom_ou_numero
```

Sauvegarde le nom ou le numéro dans la liste noire

```
database del blacklist nom_ou_numero
```

Supprime le nom ou le numéro de la liste noire

### VII. Documentation

Les documents suivants complètent les informations de ce sujet :

- Cours VoIP et SIP d'André Aoun
- Téléphonie sur IP : Laurent Ouakil et Guy Pujolle
- Asterisk : <http://www.asterisk.org/>
- Asterisk: The Future of Telephony par Jim van Meggelen, Jared Smith, et Leif Madsen. Disponible sous licence *Creative Commons*
- Asterisk Guru : <http://www.asteriskguru.com/>
- VoIP Info : <http://www.voip-info.org/>

---

*Vos notes :*



Les fichiers de configuration d'Asterisk ET les traces de communication (§IV.6.c) sont à rendre par courrier électronique à la fin des TP.