

utc

TZ

Mise en place d'une communication VoIP entre un Raspberry Pi et un téléphone IP

Rapport technique

Description du sujet : L'objectif de ce projet est d'établir une communication téléphonique sur IP entre un raspberry PI et un téléphone IP (Marque Alcatel) via un réseau local. Nous utiliserons le protocole SIP pour établir cette communication téléphonique entre les deux terminaux (téléphone ip + Raspberry) à travers un serveur PABX Asterisk.

Dépôt Git : <https://gitlab.utc.fr/nibertgu/tz-voip-raspberry-pi>.

Guillaume Nibert

Encadrant : Dr. Ahmed Lounis

12/02/2021

Table des matières

Table des matières	2
Introduction	4
1. Protocole SIP et communication VoIP	6
2. Mise en place d'un serveur PABX Asterisk	8
Choix technologiques	9
Prérequis	9
Installation d'Asterisk	10
Configuration de SIP et création d'utilisateurs	20
Configuration de SIP - pjsip.conf	20
Plan de numérotation - extensions.conf	22
3. Installation et configuration d'un client SIP sur le Raspberry Pi	24
Choix technologiques	24
Prérequis	24
Installation et configuration du client SIP Linphone	25
Vérification côté client	25
Vérification côté serveur	26
4. Configuration du téléphone IP et tests de communication	27
Informations sur la configuration	29
Configuration du téléphone en mode SIP - sur l'appareil	30
Installation du serveur HTTP	32
Transfert des fichiers de configuration et des firmwares vers le téléphone Alcatel	33
Vérification côté client	37
Vérification côté serveur	37
5. Tests de communication	39
Raspberry Pi vers Alcatel IP Touch	39
Alcatel IP Touch vers Raspberry Pi	41
6. Développement d'un programme client SIP en JavaScript	43
WebRTC & WebSocket	43
Configuration du serveur Asterisk pour prendre en charge l'API WebRTC	45
Génération d'un certificat SSL/TLS auto signé	45
Activation du serveur HTTP d'Asterisk	48
Modification de pjsip.conf pour prendre en charge WebRTC	49
Modification de extensions.conf pour prendre en charge WebRTC	52
Tests de communication avec le client Web Browser Phone	53
Développement d'un client SIP JavaScript	57

Conclusion	58
Table des illustrations (hors annexes)	59
Sigles (hors annexes)	60
Références	62
Annexes	64
Annexe A1 - Installation d'une machine virtuelle sous Debian 10 Buster	64
I - Préparation de la VM	64
II - Installation et configuration de Debian 10	68
Annexe A2 - Installation et configuration de Raspberry Pi OS Buster (64 bits) pour Raspberry Pi 3B+	78

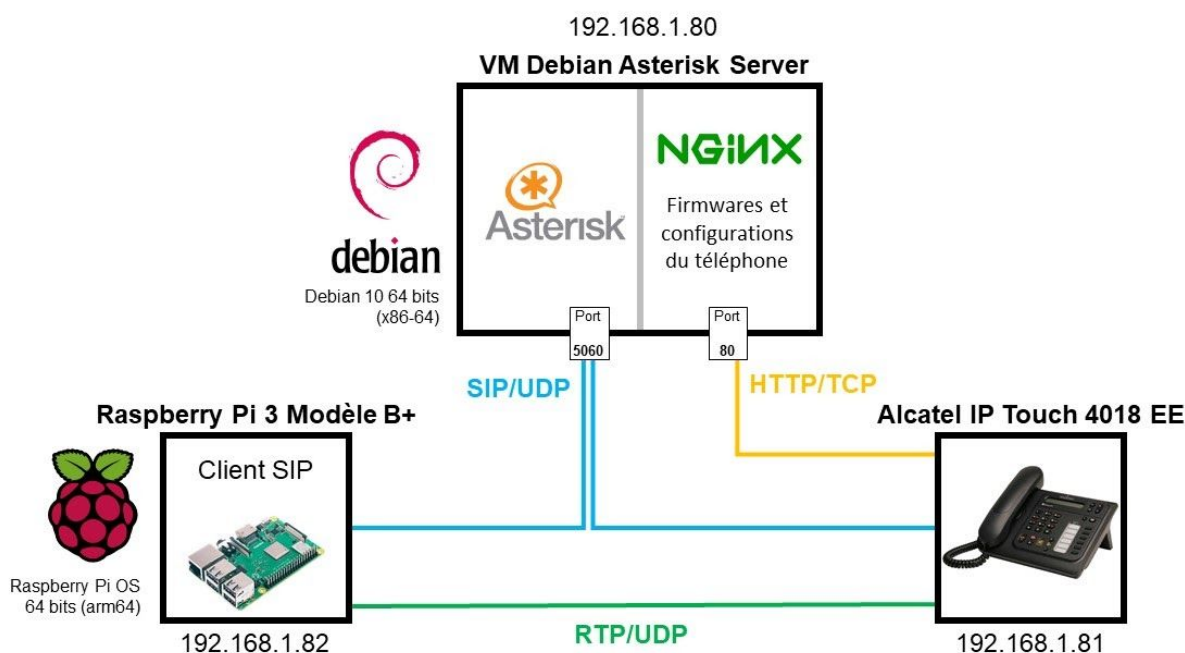
Introduction

Télégraphe, téléphone avec liaison analogique, téléphone avec liaison numérique, téléphone IP avec des modes filaires ou non, la téléphonie représente actuellement un lien mondial très utilisé et très commode, indispensable aux échanges rapides et en temps réel, tout usage confondu. Plutôt fiable, pas trop onéreux, offrant à une cadence de plus en plus rapide des possibilités de plus en plus étendues.

Le but de ce sujet est d'utiliser le protocole IP utilisé dans l'internet, pour faire communiquer des appareils entre eux (communément appelés points de terminaison). Cette TZ mettra donc en œuvre deux points de terminaison dont les caractéristiques matérielles sont les suivantes :

- point de terminaison 1 : téléphone Alcatel IP Touch 4018 EE ;
- point de terminaison 2 : Raspberry Pi 3 Modèle B+.

Ci-dessous, un schéma global de l'architecture de l'infrastructure dans le réseau **192.168.1.0** (le routeur n'est pas représenté pour des raisons de lisibilité) :



(Figure 1 - Schéma global de l'infrastructure)

Simplement, sans rentrer dans les détails, dans ce schéma, il y a deux clients : les points de terminaison que sont l'*Alcatel IP Touch 4018 EE* et le *Raspberry Pi*. Lorsqu'il y a une communication téléphonique entre ces deux appareils, le protocole utilisé pour l'initiation de la communication est SIP (*Session Initiation Protocol*) pour la couche applicative et UDP pour la couche transport. Cette initiation passe par un intermédiaire : le

serveur d'initiation SIP (Asterisk¹) dont l'accessibilité se fait sur la machine virtuelle Debian Asterisk Server au port 5060 (**en bleu**). Une fois l'initiation faite, les deux appareils se connectent entre eux directement pour laisser passer l'audio via le protocole RTP (**en vert**).

Un point important concerne la partie du téléphone Alcatel, qui a besoin à chaque démarrage de vérifier ses firmwares et ses fichiers de configuration lui permettant de se connecter au serveur *Asterisk*. Ces fichiers sont stockés dans la machine virtuelle Debian et disponibles pour l'Alcatel via le serveur HTTP au niveau du port 80 (**en orange**).

Ce rapport détaille l'architecture du système ainsi que la mise en place de toute l'infrastructure de téléphonie IP autorisant la communication entre ces deux appareils. Ainsi nous verrons en premier lieu quelques éléments théoriques sur les protocoles utilisés, notamment sur le rôle et le fonctionnement de SIP dans la VoIP et RTP. En deuxième lieu, nous évaluerons l'intérêt qu'il y a à utiliser le système Asterisk dans la conception de cette infrastructure et à sa mise en place, puis nous configurerons et testerons les deux clients (Raspberry Pi et téléphone Alcatel). Enfin nous concevrons un client SIP convivial en JavaScript pour le Raspberry Pi.

¹ En réalité *Asterisk* est un système qui est aussi capable de gérer d'autres protocoles. Il dispose d'un serveur SIP. Lorsque nous parlons du serveur *Asterisk* dans ce rapport, il faut comprendre que l'on parle du serveur SIP qu'*Asterisk* propose.

1. Protocole SIP et communication VoIP

Les premières questions à se poser avant de démarrer la mise en place de cette infrastructure sont : qu'est-ce que la VoIP ? Qu'est-ce que le protocole SIP et quel est son rapport avec la VoIP ? Quel est le lien entre le protocole SIP et le protocole RTP ?

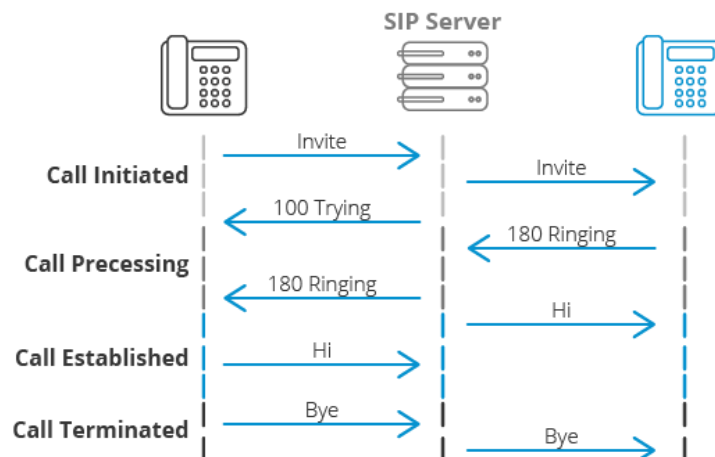
Pour répondre à la première question, il faut reprendre la pile d'Internet (suite des protocoles Internet).

Application	SIP, HTTP, SMTP, FTP, RTP...
Présentation	
Session	
Transport	UDP/TCP
Réseau	IP
Liaison	802.3 MAC, 802.11 MAC, EAP...
Physique	802.3 PHY, 802.11 PHY, cuivre, optique...

(Figure 2 - Pile protocolaire d'internet)

VoIP signifie *Voice over Internet Protocol*, la voix est concrètement un signal analogique qui peut être acquis au moyen d'un microphone puis encodé numériquement via un convertisseur analogique → numérique. Une fois encodée, il est possible de faire passer la donnée dans la pile protocolaire de la suite des protocoles Internet. Dans ce schéma, il est donc possible de transférer la voix encodée via par exemple le protocole HTTP, transportée en TCP, dans un réseau IP via une liaison Ethernet vers un autre périphérique utilisant cette même pile protocolaire. Attention, tous les protocoles applicatifs ne sont pas forcément appropriés au transfert de voix encodées par exemple. Certains sont appropriés à pour réaliser de la communication en temps réel, par exemple la transmission de voix encodées (RTP), d'autres pour l'établissement d'une communication (SIP)...

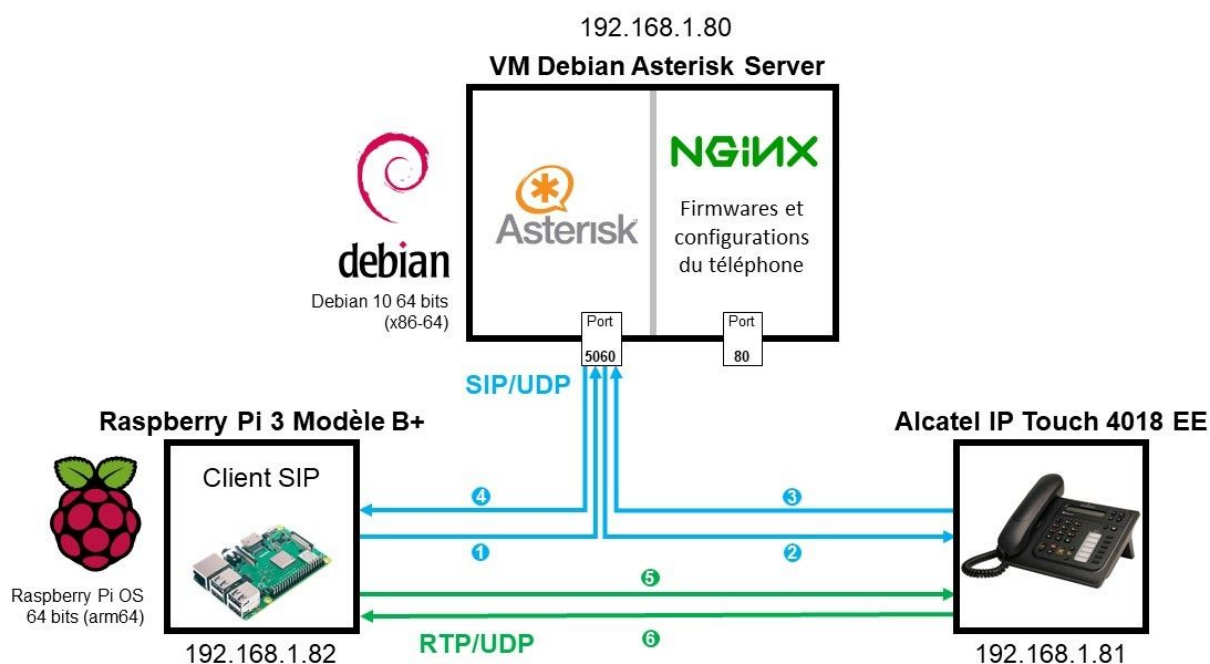
SIP est le protocole permettant d'établir la communication entre des points de terminaison. Typiquement, un point de terminaison se comporte dans un premier temps comme un client SIP pour contacter un autre point de terminaison. Il interroge d'abord un serveur SIP qui lui fournit les informations sur l'identité de cet autre point de terminaison et comment y accéder dans le réseau IP. Une fois les informations reçues, la communication s'effectue directement entre les deux points de terminaisons via le protocole RTP (*Real-time Transport Protocol*). Ce dernier protocole transmet la voix encodée. Une fois la communication terminée, SIP reprend à nouveau le relais pour fermer la connexion (session).



(Figure 3 - Établissement et fin de la communication - crédits : 3cx.fr)

Pour revenir à notre schéma, pour l'établissement d'une communication (par exemple Raspberry Pi appelle Alcatel) :

- RPi envoie une demande de contact à Alcatel (*Invite*) en passant par Asterisk. Alcatel reçoit la réponse *Invite*, sonne et indique qu'il sonne à RPi. Quelqu'un décroche l'Alcatel, le contact est établi, donc...**
- ...RTP prend le relais et transfère les voix entre les deux appareils en connexion directe (sans passer par Asterisk).**



(Figure 4 - Établissement d'une communication téléphonique entre le Raspberry Pi et le téléphone Alcatel IP Touch 4018 EE)

Une fois la communication terminée (une personne raccroche), SIP se charge de fermer la session entre les deux participants.

2. Mise en place d'un serveur PABX Asterisk

Dans les grandes structures (entreprises, services publics...), on retrouve principalement deux types de communications téléphoniques : la communication interne et la communication externe. Les agents se retrouvent donc avec un téléphone fixe interne capable d'émettre des appels vers un autre téléphone fixe interne ou un téléphone externe (public). Pour que cela fonctionne, il faut un PABX², aussi appelé autocommutateur téléphonique privé.



Le PABX (ci-gauche), présente de nombreux avantages, notamment :

- financièrement, la facture s'allège, les appels internes ne passent pas par le réseau public (Orange, anciennement France Télécom) ;
- on peut attribuer davantage de numéros interne sans difficulté ;
- il est possible de proposer des services tels que la conférence téléphonique, les renvois, le transfert d'appel ;
- et, bien entendu, relier une ligne interne à une ligne externe.

Vous trouverez une liste exhaustive des fonctions d'un PABX traditionnel sur Wikipédia **(1)**.

(Figure 5 - PABX Matra série MC6500)

Le fonctionnement global de ce système étant présenté, comment peut-on donc réaliser une communication utilisant le protocole SIP ?

Historiquement, il y a eu trois grandes phases dans l'évolution des communications téléphoniques :

1. Le réseau téléphonique commuté (RTC), analogique. Dans ce réseau, impossible d'utiliser le protocole SIP puisque ce dernier est numérique.
2. Le réseau numérique à intégration de services (RNIS ou ISDN dans sa version anglaise), numérique, dont la base de la pile réseau est conforme au modèle OSI. À partir de la couche 3 (transport), on retrouve les protocoles suivants : Q.931 **(2)**, X.25 couche 3 **(3)**. Ici, c'est la couche 3 qui pose problème. SIP s'appuie sur IP pour fonctionner.
3. La voix sur IP, sur le réseau Internet, est numérique. SIP pourra fonctionner dans ce cas. Le PABX associé est un PABX IP ou encore appelé IPBX (Internet Protocol Branch eXchange).

² PABX, de l'anglais : Private Automatic Branch eXchange

Il existe de nombreux IPBX dans le monde. Asterisk a la particularité d'être le numéro 1 mondial en termes d'utilisation. Il comporte une version libre et une version propriétaire. Et propose une interopérabilité avec les anciens réseaux (RTC et RNIS) au moyen de cartes matérielles et de modules logiciels. Ce dernier point est probablement un grand facteur de son succès : les entreprises qui utilisent encore du vieux matériel, et qui, dans une optique de modernisation migrent vers du matériel récent, utilisent probablement plusieurs systèmes (RTC, RNIS ou encore VoIP), Asterisk va permettre de gérer simultanément ces trois systèmes.

Procédons donc à la mise en place d'Asterisk.

Choix technologiques

- OS : Debian 10, il est libre et est principalement utilisé en tant que serveur dans le monde informatique. C'est un système proposant des paquets régulièrement mis à jour en termes de stabilité et de sécurité.
- Asterisk : version 18 LTS³ compilée à partir de ses sources. La version dans les dépôts de Debian est ancienne (16 pour Debian 10) et bien qu'elle soit aussi une version LTS, le fait qu'elle soit déjà compilée offre moins de flexibilité en termes d'ajout de modules. Par ailleurs, le module assurant la gestion du protocole SIP⁴ dans la version 16 est dépréciée depuis la version 17 **(4)** au profit d'un nouveau module (**PJSIP**) capable de gérer le protocole SIP ainsi que les fonctions de traversée de NAT avec SIP⁵ **(5)**.

Prérequis

Avoir une machine Debian 10 (sans interface graphique) et à jour⁶ connectée au réseau local et à internet, disposant également d'un accès SSH (cf. [annexe A1](#) pour la mise en place détaillée de ce prérequis).

Considérons dans cette partie les informations suivantes de cette machine :

Adresse IP	Utilisateur	Mot de passe
192.168.1.80	asterisktz	voiputc
	root	voiputc

³ LTS : Long Term Support, version maintenue pendant une longue durée. Ce sont les versions à privilégier pour une mise en production.

⁴ *chan_sip.so*, le module de gestion SIP d'Asterisk 16 et antérieur n'est officiellement plus maintenu.

⁵ La problématique de traversée NAT avec SIP n'est pas vue dans ce rapport.

⁶ `sudo apt update && sudo apt upgrade -y && sudo apt dist-upgrade -y && sudo apt autoremove -y && sudo reboot`

Installation d'Asterisk

Remarque : nous n'installerons pas les librairies permettant la gestion des cartes d'interface téléphoniques de Digium⁷ (société maintenant Asterisk) et la gestion des protocoles utilisés dans les réseaux RNIS⁸.

1. Se connecter en SSH à la machine `asterisktz`.

```
# ssh login@adresse_ip_vm -p 22  
ssh asterisktz@192.168.1.80 -p 22
```

2. Installer les paquets permettant la compilation d'Asterisk et des prérequis.

```
asterisktz@asterisktz:~$ Installation de la chaîne de compilation et prérequis  
  
sudo apt update && sudo apt install linux-headers-$(uname -r)  
build-essential autoconf libglib2.0-dev libtool net-tools
```

3. Redémarrer la machine Debian et se connecter à nouveau en SSH.

```
asterisktz@asterisktz:~$ Redémarrage de la machine Debian  
  
sudo reboot
```

```
# Reconnexion en SSH  
ssh asterisktz@192.168.1.80 -p 22
```

4. Se placer dans le répertoire `/usr/src` et télécharger Asterisk 18 à cette adresse :
<https://downloads.asterisk.org/pub/telephony/asterisk/asterisk-18-current.tar.gz>.

```
asterisktz@asterisktz:~$ Changement de répertoire vers /usr/src  
  
cd /usr/src
```

```
asterisktz@asterisktz:/usr/src$ Téléchargement d'Asterisk 18 LTS  
  
sudo wget https://downloads.asterisk.org/pub/telephony/asterisk/asterisk-18-current.tar.gz
```

5. Décompresser l'archive.

```
asterisktz@asterisktz:/usr/src$ Décompression de l'archive  
  
sudo tar -zxvf asterisk-18-current.tar.gz
```

⁷ Librairie DAHDI (Digium/Asterisk Hardware Device Interface) :
<https://wiki.asterisk.org/wiki/pages/viewpage.action?pageId=4817506>.

⁸ Librairie libpri : <https://wiki.asterisk.org/wiki/pages/viewpage.action?pageId=4817506>.

6. Exécuter le script `install_prereq` installant les prérequis.

Remarque : pour connaître le numéro de version mineure, taper `ls`. Dans ce rapport, il s'agit d'Asterisk 18.2.

```
asterisktz@asterisktz: /usr/src$
```

Changement de répertoire

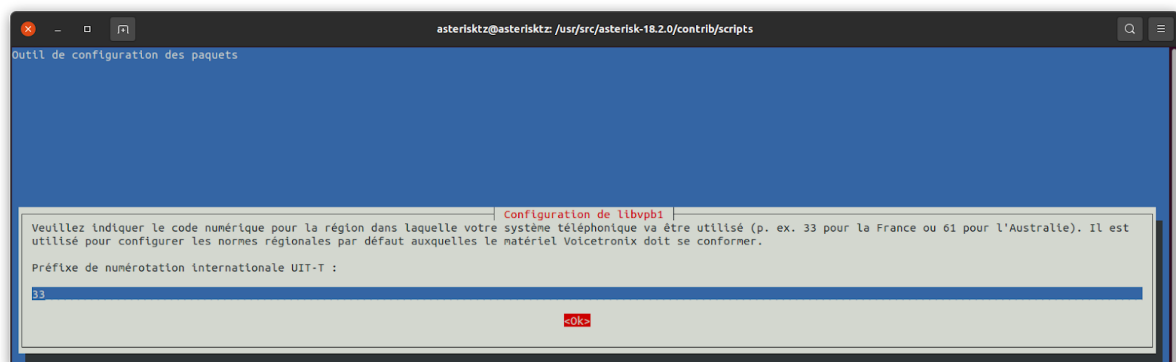
```
cd asterisk-18.2.0/contrib/scripts/
```

```
asterisktz@asterisktz: /usr/src/asterisk-18.2.0/contrib/scripts$
```

Prérequis

```
sudo ./install_prereq install
```

Pendant l'installation des prérequis une fenêtre s'affiche demandant de sélectionner l'indicatif des numéros de téléphones. Sélectionner **33** pour la France et valider par **<Ok>**.



(Figure 6 - Paramétrage indicatif téléphonique)

7. Vérification des dépendances requises.

```
asterisktz@asterisktz: /usr/src/asterisk-18.2.0/contrib/scripts$
```

Chgt de répertoire

```
cd ../..
```

```
asterisktz@asterisktz: /usr/src/asterisk-18.2.0$
```

Exécution du script `configure`

```
sudo ./configure
```

Si l'opération se déroule bien, on obtient un message similaire à celui-ci avec le logo d'Asterisk :

```

      .$$$$$$$$$$$$$$$$$=..
      .7$7$7..           .7$7$7:..
      .7$7$7..           .7$7$7:..
      .$$:..             ,,$7.7
      .7$.              7$$$$$      .$$77
      ..$$$.            $$$$$$      .$$$7
      ..7$ .?.          $$$$$$ .?.   7$$$$.
      $.$. .$$$7. $$$$7 .7$$$$.      .$$$$.
      .777. .$$$$$$$77$$$$77$$$$$7.  $$$$,
      $$$~ .7$$$$$$$$$$$$$$$$$7.     .$$$$.
      .$$7 .7$$$$$$$$$7:              ?$$$$.
      $$$  ?7$$$$$$$$$$$$$I           .$$$7
      $$$  .7$$$$$$$$$$$$$$$$$$$      :$$$$.
      $$$  $$$$$$7$$$$$$$$$$$$$$$     .$$$$.
      $$$  $$$  7$$$7 .$$$ .$$$$.     .$$$$.
      $$$  $$$$$  $$$$$7 .$$$$.      .$$$$.
      7$$$7 7$$$7 7$$$7 7$$$7
      $$$$$$ $$$$ $$$$
      $$$7.  $$$ (TM)
      $$$$$$. .7$$$$$ $
      $$$$$$$$$$7$$$$$$$$$. $$$$$$
      $$$$$$$$$$$$$$.

```

```

configure: Package configured for:
configure: OS type : linux-gnu
configure: Host CPU : x86_64
configure: build-cpu:vendor:os: x86_64 : pc : linux-gnu :
configure: host-cpu:vendor:os: x86_64 : pc : linux-gnu :

```

Si cela s'est mal déroulé, voici un lien vers la documentation :

<https://wiki.asterisk.org/wiki/display/AST/Checking+Asterisk+Requirements>

8. Sélection des options à installer pour Asterisk. Veiller à ce que le terminal ait une taille minimale de **80 x 27**.

```
asterisktz@asterisktz: /usr/src/asterisk-18.2.0$
```

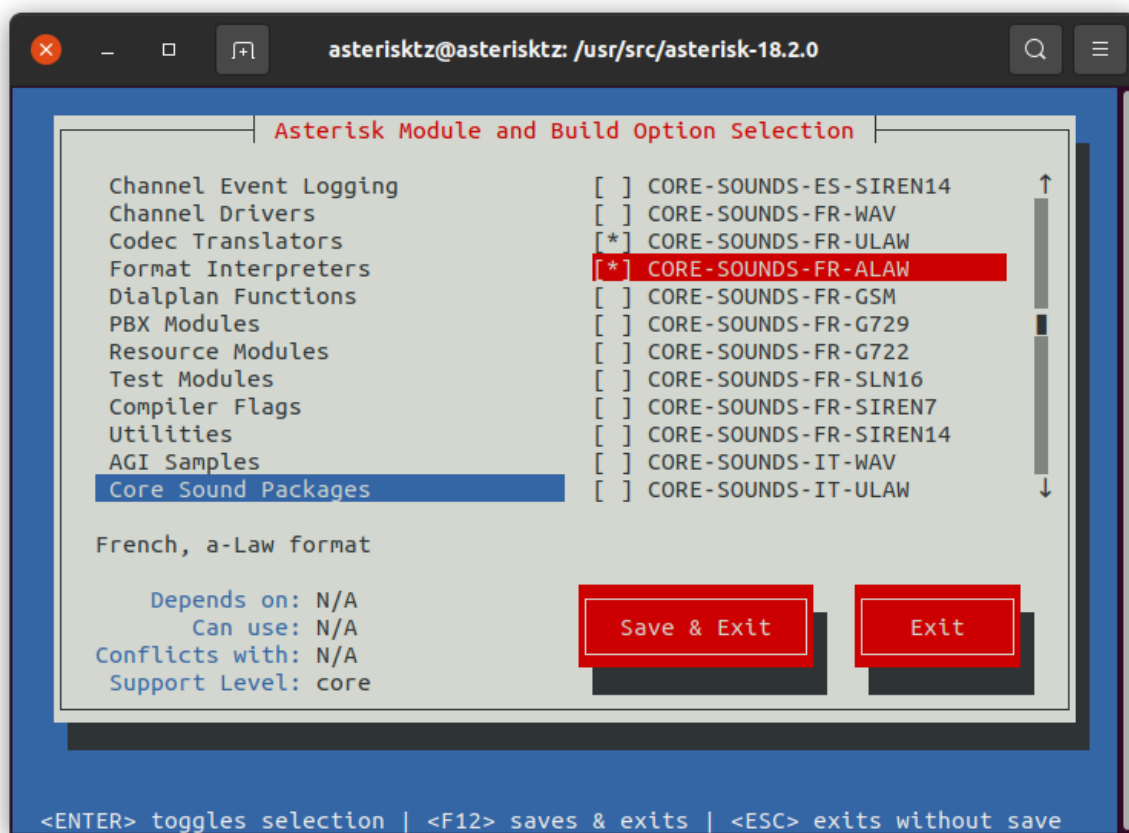
Choix des options pour Asterisk

```
sudo make menuselect
```

Une fenêtre s'affiche permettant de choisir les options.

Aller dans la section Core Sound Package pour installer les sons français (messages vocaux) :

- Désélectionner le package **CORE-SOUNDS-EN-GSM**. Il s'agit des sons anglais avec le codec audio GSM.
- Sélectionner les packages **CORE-SOUNDS-FR-ULAW** et **CORE-SOUNDS-FR-ALAW**. Il s'agit des sons français avec les codecs ULAW et ALAW⁹ (pris en charge par l'Alcatel IP Touch 4018EE (6)).



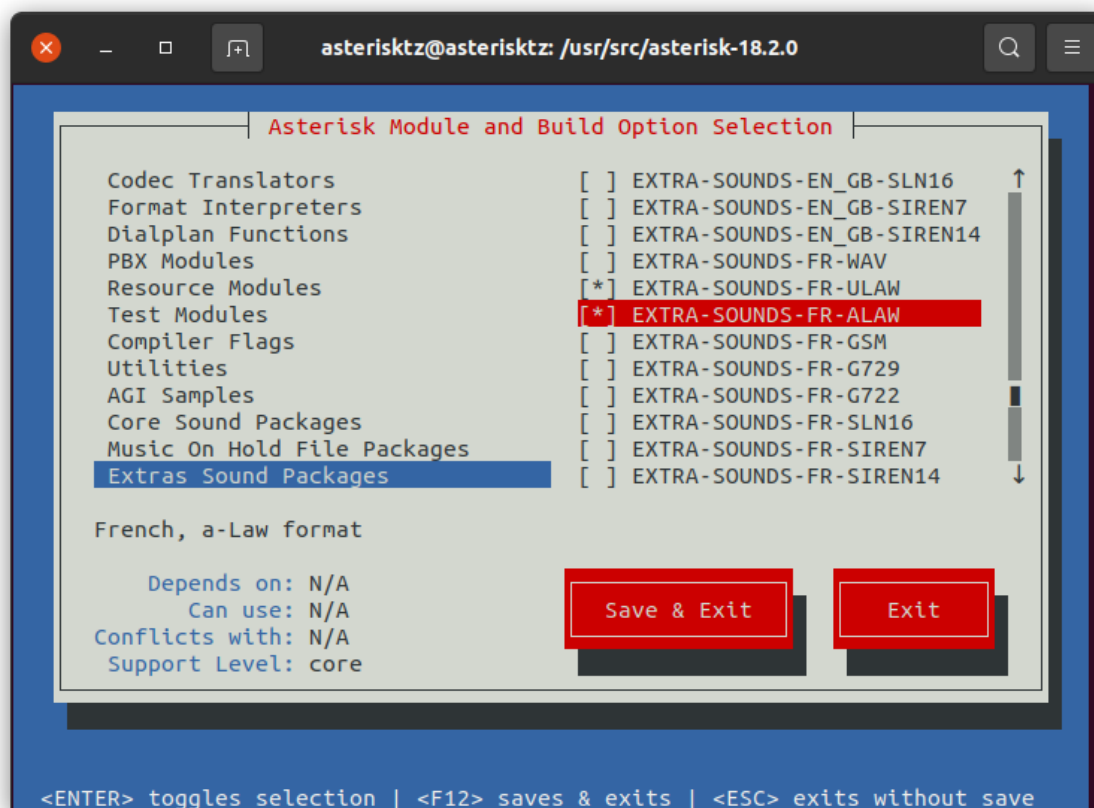
(Figure 7 - Sélection des modules de sons d'Asterisk)

⁹ Ces codecs sont définis par la norme G.711 de l'UIT :

<https://www.itu.int/rec/T-REC-G.711-198811-I/fr>

Le codec ALAW est utilisé en Europe et en Afrique. Le codec ULAW en Amérique du Nord et au Japon (7).

Puis dans la section Extra Sound Packages, sélectionner les packages
EXTRA-SOUNDS-FR-ULAW et **EXTRA-SOUNDS-FR-ALAW**.



(Figure 8 - Sélection des modules de sons extra d'Asterisk)

Valider en tapant la touche **F12**.

9. Compiler le programme avec les options précédemment choisies puis installer Asterisk.

```
asterisktz@asterisktz: /usr/src/asterisk-18.2.0$
```

Compilation d'Asterisk

```
sudo make
```

La compilation prend du temps (en général ~15 min) selon la puissance de la machine. Une fois réussie...

```
+----- Asterisk Build Complete -----+
+ Asterisk has successfully been built, and +
+ can be installed by running:             +
+                                           +
+               make install               +
+----- Asterisk Build Complete -----+
```

...on installe Asterisk.

```
asterisktz@asterisktz: /usr/src/asterisk-18.2.0$
```

Installation d'Asterisk

```
sudo make install
```

L'installation se termine sur ce message :

```
+----- Asterisk Installation Complete -----+
+
+   YOU MUST READ THE SECURITY DOCUMENT
+
+ Asterisk has successfully been installed.
+ If you would like to install the sample
+ configuration files (overwriting any
+ existing config files), run:
+
+ For generic reference documentation:
+   make samples
+
+ For a sample basic PBX:
+   make basic-pbx
+
+----- or -----+
+
+ You can go ahead and install the asterisk
+ program documentation now or later run:
+
+           make progdocs
+
+ **Note** This requires that you have
+ doxygen installed on your local system
+-----+
```

10. Créer les fichiers d'exemples de configuration dans le dossier `/etc/asterisk`.

```
asterisktz@asterisktz: /usr/src/asterisk-18.2.0$
```

Création fichiers d'exemple de config.

```
sudo make samples
```

11. Installer les scripts de démarrage.

```
asterisktz@asterisktz: /usr/src/asterisk-18.2.0$
```

Installation des scripts de démarrage

```
sudo make config
sudo ldconfig
```

12. Démarrage automatique du service Asterisk au lancement de la machine.

asterisktz@asterisktz: /usr/src/asterisk-18.2.0\$

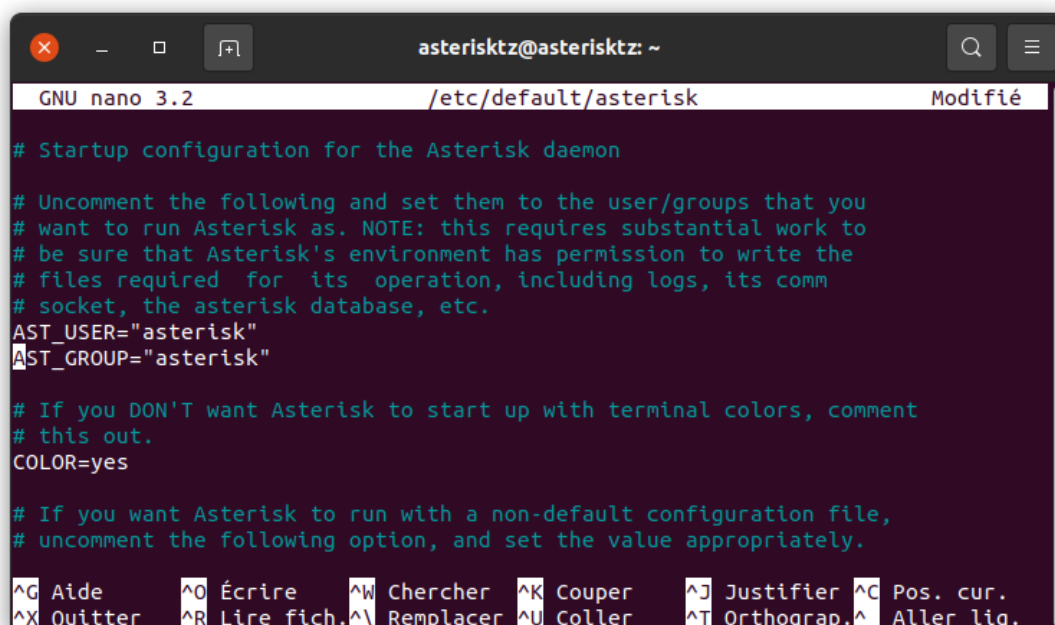
Création d'un utilisateur asterisk

```
cd
sudo groupadd asterisk
sudo useradd -r -d /var/lib/asterisk -g asterisk asterisk
sudo usermod -aG audio,dialout asterisk
sudo chown -R asterisk.asterisk /etc/asterisk
sudo chown -R asterisk.asterisk /var/{lib,log,spool}/asterisk
sudo chown -R asterisk.asterisk /usr/lib/asterisk
```

asterisktz@asterisktz: ~\$ Ajouter asterisk en tant qu'utilisateur par défaut du service **Asterisk** - 1

```
sudo nano /etc/default/asterisk
```

Décommenter les lignes `AST_USER="asterisk"` et `AST_GROUP="asterisk"` (enlever le croisillon # devant chaque ligne). Enregistrer avec **Ctrl + O**. Quitter avec **Ctrl + X**.



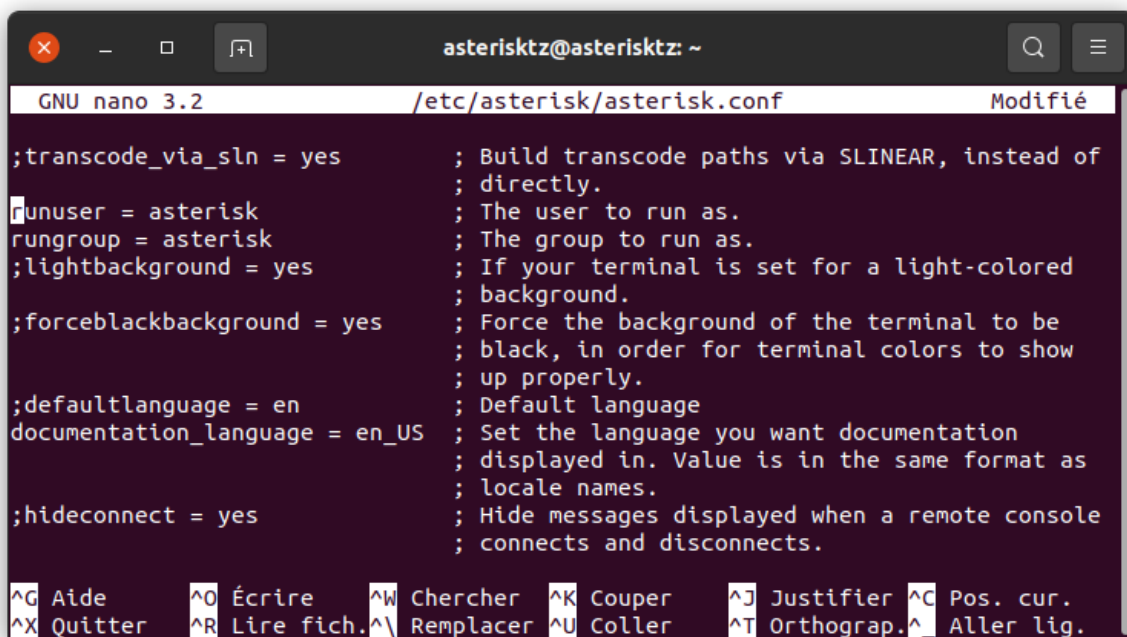
(Figure 9 - Configuration d'asterisk en temps qu'utilisateur par défaut du service Asterisk)

asterisktz@asterisktz: ~\$ Ajouter asterisk en tant qu'utilisateur par défaut du service **Asterisk** - 2

```
sudo nano /etc/asterisk/asterisk.conf
```

Décommenter les lignes (enlever le point-virgule ; devant chaque ligne). Enregistrer avec **Ctrl + O**. Quitter avec **Ctrl + X**.

```
runuser = asterisk ; The user to run as.
rungroup = asterisk ; The group to run as.
```

```
GNU nano 3.2 /etc/asterisk/asterisk.conf Modifié

;transcode_via_sln = yes      ; Build transcode paths via SLINEAR, instead of
                                ; directly.
runuser = asterisk            ; The user to run as.
rungroup = asterisk           ; The group to run as.
;lightbackground = yes        ; If your terminal is set for a light-colored
                                ; background.
;forceblackbackground = yes   ; Force the background of the terminal to be
                                ; black, in order for terminal colors to show
                                ; up properly.
;defaultlanguage = en         ; Default language
documentation_language = en_US ; Set the language you want documentation
                                ; displayed in. Value is in the same format as
                                ; locale names.
;hideconnect = yes            ; Hide messages displayed when a remote console
                                ; connects and disconnects.

^G Aide      ^O Écrire   ^W Chercher  ^K Couper    ^J Justifier ^C Pos. cur.
^X Quitter   ^R Lire fich.^_ Remplacer  ^U Coller    ^T Orthograp.^_ Aller lig.
```

(Figure 10 - Configuration d'asterisk en temps qu'utilisateur par défaut du service Asterisk)

13. Démarrer le service Asterisk

```
asterisktz@asterisktz:~$
```

Démarrage d'Asterisk

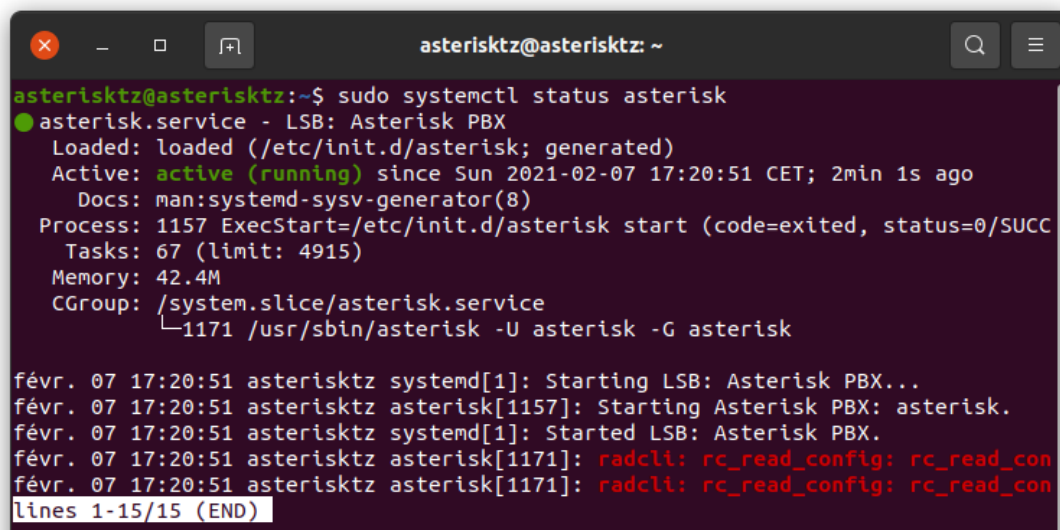
```
sudo systemctl start asterisk
```

Vous pouvez maintenant vérifier son statut actuel via la commande suivante :

```
asterisktz@asterisktz:~$
```

Démarrage d'Asterisk

```
sudo systemctl status asterisk
```



```
asterisktz@asterisktz: ~  
asterisktz@asterisktz:~$ sudo systemctl status asterisk  
● asterisk.service - LSB: Asterisk PBX  
   Loaded: loaded (/etc/init.d/asterisk; generated)  
   Active: active (running) since Sun 2021-02-07 17:20:51 CET; 2min 1s ago  
     Docs: man:systemd-sysv-generator(8)  
  Process: 1157 ExecStart=/etc/init.d/asterisk start (code=exited, status=0/SUCCESS)  
    Tasks: 67 (limit: 4915)  
   Memory: 42.4M  
    CGroup: /system.slice/asterisk.service  
            └─1171 /usr/sbin/asterisk -U asterisk -G asterisk  
  
févr. 07 17:20:51 asterisktz systemd[1]: Starting LSB: Asterisk PBX...  
févr. 07 17:20:51 asterisktz asterisk[1157]: Starting Asterisk PBX: asterisk.  
févr. 07 17:20:51 asterisktz systemd[1]: Started LSB: Asterisk PBX.  
févr. 07 17:20:51 asterisktz asterisk[1171]: radcli: rc_read_config: rc_read_con  
févr. 07 17:20:51 asterisktz asterisk[1171]: radcli: rc_read_config: rc_read_con  
lines 1-15/15 (END)
```

(Figure 11 - Lancement du service Asterisk avec erreurs)

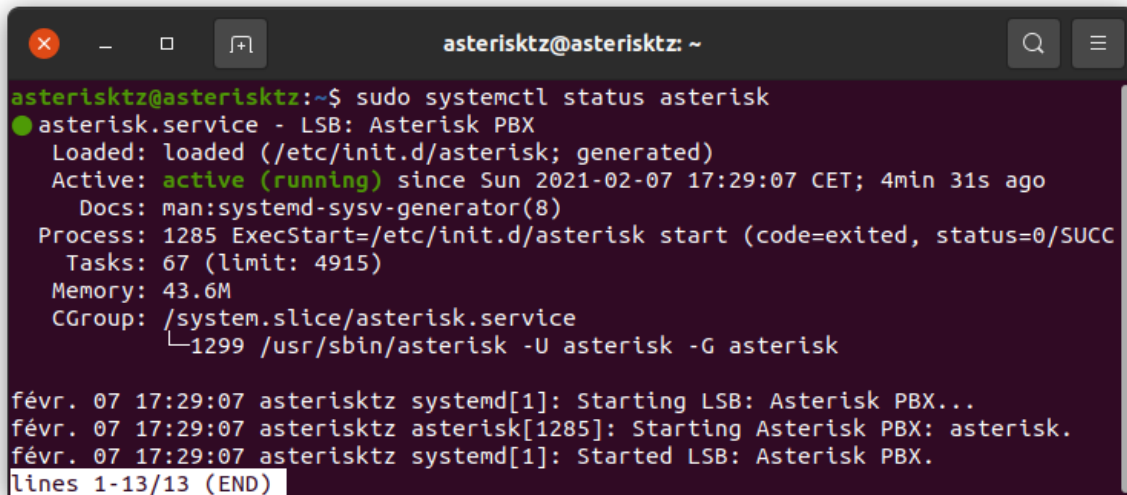
À ce stade, s'il y a ces deux lignes rouges, on peut les corriger de cette manière¹⁰ :

```
asterisktz@asterisktz:~$
```

Fix

```
sudo systemctl stop asterisk  
  
sudo sed -i 's";\[radius\]"\[radius\]"g' /etc/asterisk/cdr.conf  
  
sudo sed -i 's";radiuscfg =>  
/usr/local/etc/radiusclient-ng/radiusclient.conf"radiuscfg =>  
/etc/radcli/radiusclient.conf"g' /etc/asterisk/cdr.conf  
  
sudo sed -i 's";radiuscfg =>  
/usr/local/etc/radiusclient-ng/radiusclient.conf"radiuscfg =>  
/etc/radcli/radiusclient.conf"g' /etc/asterisk/cel.conf  
  
sudo systemctl start asterisk
```

¹⁰ Fix : <https://clearhat.org/blog/a-fix-for-apt-install-asterisk-on-ubuntu-18-04>.

A terminal window titled 'asterisktz@asterisktz: ~' showing the output of the command 'sudo systemctl status asterisk'. The output indicates that the 'asterisk.service' is loaded and active (running) since Sun 2021-02-07 17:29:07 CET. It shows the process ID 1285, the start command, and the tasks and memory usage. The bottom of the terminal shows the logs for the service starting successfully.

```
asterisktz@asterisktz:~$ sudo systemctl status asterisk
● asterisk.service - LSB: Asterisk PBX
   Loaded: loaded (/etc/init.d/asterisk; generated)
   Active: active (running) since Sun 2021-02-07 17:29:07 CET; 4min 31s ago
     Docs: man:systemd-sysv-generator(8)
  Process: 1285 ExecStart=/etc/init.d/asterisk start (code=exited, status=0/SUCC
    Tasks: 67 (limit: 4915)
   Memory: 43.6M
    CGroup: /system.slice/asterisk.service
            └─1299 /usr/sbin/asterisk -U asterisk -G asterisk

févr. 07 17:29:07 asterisktz systemd[1]: Starting LSB: Asterisk PBX...
févr. 07 17:29:07 asterisktz asterisk[1285]: Starting Asterisk PBX: asterisk.
févr. 07 17:29:07 asterisktz systemd[1]: Started LSB: Asterisk PBX.
lines 1-13/13 (END)
```

(Figure 12 - Lancement du service Asterisk sans erreur)

14. Démarrage automatique du service Asterisk.

asterisktz@asterisktz:~\$

Démarrage automatique d'Asterisk

```
sudo /lib/systemd/systemd-sysv-install enable asterisk
```

Asterisk est opérationnel. Passons donc à la création d'utilisateurs et la configuration SIP !

Configuration de SIP et création d'utilisateurs

Nous avons donc un serveur opérationnel, il faut donc maintenant configurer SIP et créer des utilisateurs.

Nous allons créer 3 utilisateurs dont les caractéristiques sont les suivantes :

	Téléphone Alcatel	Raspberry Pi	Test
Usage	Compte dédié pour le téléphone Alcatel IP Touch 4018 EE	Compte dédié pour le Raspberry Pi	Compte dédié pour les tests
Nom d'affichage	Alcatel IP Touch	Raspberry Pi	Guillaume Nibert
Numéro de téléphone	5001	5002	5003
Identifiant	alcatel	rpi	guillaume
Mot de passe	11111111	22222222	33333333

Il y a deux fichiers de configuration à éditer : *pjsip.conf* et *extensions.conf*. Le premier sert à créer les comptes, configurer le fonctionnement de SIP (UDP/TCP), les systèmes d'authentification... et le deuxième à définir le comportement du système, plus précisément le plan de numérotation (similaire au routage si l'on parlait de paquets IP). Ce "routage" se fait en fonction des numéros de téléphone (identifiants).

Ces fichiers sont présents dans */etc/asterisk* et dans le dépôt GitLab, répertoire *asterisk_sip*.

Configuration de SIP - pjsip.conf

1. Renommer le fichier de configuration *pjsip.conf* par *pjsip_original.conf*.

```
asterisktz@asterisktz:~$
```

Sauvegarde du fichier de configuration *pjsip.conf* initial

```
sudo mv /etc/asterisk/pjsip.conf /etc/asterisk/pjsip_original.conf
```

2. Créer un fichier *pjsip.conf*...

```
asterisktz@asterisktz:~$
```

Paramétrage des comptes et de SIP

```
sudo nano /etc/asterisk/pjsip.conf
```

...et placer le contenu suivant :

/etc/asterisk/pjsip.conf

```
[transport-udp]
type=transport
protocol=udp
bind=0.0.0.0

;Modèles de base, ils seront recopiés pour chaque utilisateur

[endpoint_basic] (!)
type=endpoint          ; point de terminaison (téléphone/raspberry/pc...)
context=plan-num       ; se sert du plan de numérotation défini dans extensions.conf
disallow=all           ; désactivation de tous les codecs audio
allow=ulaw              ; sauf le codec ULAW
allow=alaw              ; et le codec ALAW
language=fr

[authentication] (!)
type=auth              ; type de la section : authentification
auth_type=userpass     ; authentification par mot de passe

[aor_template] (!)
type=aor               ; savoir où le point de terminaison peut être contacté
max_contacts=1

;Définitions des comptes utilisateurs associés aux matériels

[alcatel] (endpoint_basic)
auth=alcatel
aors=alcatel
callerid="Alcatel IP Touch" <5001> ; pour avoir le nom de l'appelant qui s'affiche
[alcatel] (authentication)
password=11111111
username=alcatel
[alcatel] (aor_template)

[rpi] (endpoint_basic)
auth=rpi
aors=rpi
callerid="Raspberry Pi" <5002>
[rpi] (authentication)
password=22222222
username=rpi
[rpi] (aor_template)

[guillaume] (endpoint_basic)
auth=guillaume
aors=guillaume
callerid="Guillaume Nibert" <5003>
[guillaume] (authentication)
password=33333333
username=guillaume
[guillaume] (aor_template)
```

La création des comptes et la configuration de SIP est terminée. Passons au plan de numérotation.

Plan de numérotation - extensions.conf

1. Renommer le fichier de configuration `extensions.conf` par `extensions_original.conf`.

asterisktz@asterisktz:~\$

Sauvegarde du fichier de configuration `extensions.conf` initial

```
sudo mv /etc/asterisk/extensions.conf  
/etc/asterisk/extensions_original.conf
```

2. Créer un fichier `extensions.conf`...

asterisktz@asterisktz:~\$

Définition d'un plan de numérotation

```
sudo nano /etc/asterisk/extensions.conf
```

...et placer le contenu suivant :

`/etc/asterisk/extensions.conf`

```
[plan-num]  
exten => 5001,1,Answer(500)  
exten => 5001,2,Dial(PJSIP/alcatel,25)  
exten => 5001,3,Hangup()  
  
exten => 5002,1,Answer(500)  
exten => 5002,2,Dial(PJSIP/rpi,25)  
exten => 5002,3,Hangup()  
  
exten => 5003,1,Answer(500)  
exten => 5003,2,Dial(PJSIP/guillaume,25)  
exten => 5003,3,Hangup()  
  
; les 1, 2 et 3 correspondent aux priorités des appels des  
; applications Answer(), Dial() et Hangup(). 1 étant la priorité la  
; plus forte. On peut également écrire 1,n,n où le premier n  
; correspond à 2 et le deuxième à 3.
```

“L'application **Answer()** prend un délai (en millisecondes) comme premier paramètre. L'ajout d'un court délai est souvent utile pour s'assurer que le point de terminaison ait le temps de commencer à traiter l'audio avant de commencer la communication via l'application **Dial()**. Sinon, vous risquez de ne pas entendre le tout début” (8). **Hangup()** comme son nom l'indique raccroche l'appel en cours.

Le plan de numérotation est terminé. Les comptes SIP ont été créés. On peut donc passer à la configuration d'un client SIP sur le Raspberry Pi.

3. Installation et configuration d'un client SIP sur le Raspberry Pi

L'installation et la configuration d'un client SIP sur le Raspberry Pi est nécessaire pour communiquer en VoIP. Ce client va se connecter au serveur Asterisk et en fonction du numéro que le client appelle, le serveur se sert du plan de numérotation défini dans `extensions.conf` pour contacter le bon point de terminaison (téléphone Alcatel par exemple).

Choix technologiques

- OS : Raspberry Pi OS Buster (arm64), la version 64 bits n'est que très récente mais semble être prometteuse. En effet, elle est plus performante que la version 32 bits (armhf) **(9)**. C'est un avantage important non négligeable surtout sur une Raspberry Pi 3B+ qui va être utilisée dans un mode Desktop. Toute amélioration des performances est intéressante. Par ailleurs, Raspberry Pi OS est un système officiellement maintenu par la fondation Raspberry Pi.
- Client SIP : *linphonec*, la version en ligne de commande de *Linphone*¹¹. Elle est stable et fonctionne parfaitement sur Raspberry Pi OS. Elle est open source. Malheureusement, le client populaire *Jami*¹² (anciennement *GNU Ring*), développé par *Savoir-faire Linux* semble ne pas fonctionner correctement avec Raspberry Pi OS.

Prérequis

Disposer d'un *Raspberry Pi 3B+* sous *Raspberry Pi OS Buster (64 bits)* à jour¹³ connecté au réseau local et à internet, disposant également d'un accès SSH (cf. [annexe A2](#) pour la mise en place détaillée de ce prérequis). Le serveur Asterisk doit être en fonctionnement.

Considérons dans cette partie les informations suivantes de cette machine :

Adresse IP	Utilisateur	Mot de passe
Eth : 192.168.1.82	pi	voippiutc
Wi-Fi : 192.168.1.92		

¹¹ Site officiel de Linphone : <https://www.linphone.org/>.

¹² Site officiel de Jami : <https://jami.net/>.

¹³ `sudo apt update && sudo apt upgrade -y && sudo apt dist-upgrade -y && sudo apt autoremove -y && sudo reboot`

Installation et configuration du client SIP Linphone

1. Démarrer le Raspberry Pi puis lancer un terminal et installer Linphone.

```
pi@raspberrypi:~$
```

Installation de Linphone

```
sudo apt install linphone -y
```

2. Une fois l'installation terminée, enregistrer l'utilisateur *rpi* associé sur le serveur.

```
pi@raspberrypi:~$
```

Exécution de *linphonec*

```
linphonec
```

```
linphonec>
```

Enregistrement du client *rpi* sur le serveur Asterisk

```
#register sip:identifiant@domaine domaine <mot_de_passe>
```

```
register sip:rpi@192.168.1.80 192.168.1.80 22222222
```

Le client *rpi* est bien connecté au serveur Asterisk. Il peut donc contacter le téléphone Alcatel et réciproquement.

Vérification côté client

Si l'enregistrement sur le serveur a réussi, la commande exécutée précédemment renvoie ceci :

```
linphonec> Refreshing on sip:rpi@192.168.1.80...  
linphonec> Registration on sip:192.168.1.80 successful.  
linphonec>
```

Vérification côté serveur

Sur une console serveur, taper la commande `sudo asterisk -rvvv`

Une fois entrée, taper la commande `pjsip show endpoints`. Si le client SIP du Raspberry Pi est bien connecté alors la console renvoie ceci :

Output

```
asterisk@tz:~$ sudo asterisk -rvvv
asterisk@tz:~$ pjsip show endpoints

Endpoint: <Endpoint/CID.....> <State.....>
<Channels.>
  I/OAuth:
<AuthId/UserName.....>
  Aor: <Aor.....> <MaxContact>
  Contact: <Aor/ContactUri.....> <Hash.....> <Status>
<RTT(ms).....>
  Transport: <TransportId.....> <Type> <cos> <tos>
<BindAddress.....>
  Identify:
<Identify/Endpoint.....>
  Match: <criteria.....>
  Channel: <ChannelId.....> <State.....>
<Time.....>
  Exten: <DialedExten.....> CLCID: <ConnectedLineCID.....>
=====
==

Endpoint:   alcatel/5001                               Unavailable    0 of inf
InAuth:    alcatel/alcatel
Aor:       alcatel                                     1

Endpoint:   guillaume/5003                             Unavailable    0 of inf
InAuth:    guillaume/guillaume
Aor:       guillaume                                  1

Endpoint:   rpi/5002                                     Not in use     0 of inf
InAuth:    rpi/rpi
Aor:       rpi                                         1
Contact:   rpi/sip:rpi@192.168.1.82;transport=udp    cec2f9dd2f NonQual  nan

Objects found: 3
```

On remarque bien que le client a pour adresse IP **192.168.1.82** et qu'il est bien connecté. Le **Not in use** indique qu'il n'y a pas d'appel en cours.

Le téléphone Alcatel n'est en revanche toujours pas connecté. Nous allons donc le faire en partie suivante.

4. Configuration du téléphone IP et tests de communication

Le téléphone Alcatel IP Touch 4018 EE est un téléphone possédant deux modes de fonctionnement.

1. Le mode NOE (*New Office Environment*) : protocole de communication propriétaire développé par Alcatel/Lucent ;
2. Le mode SIP.

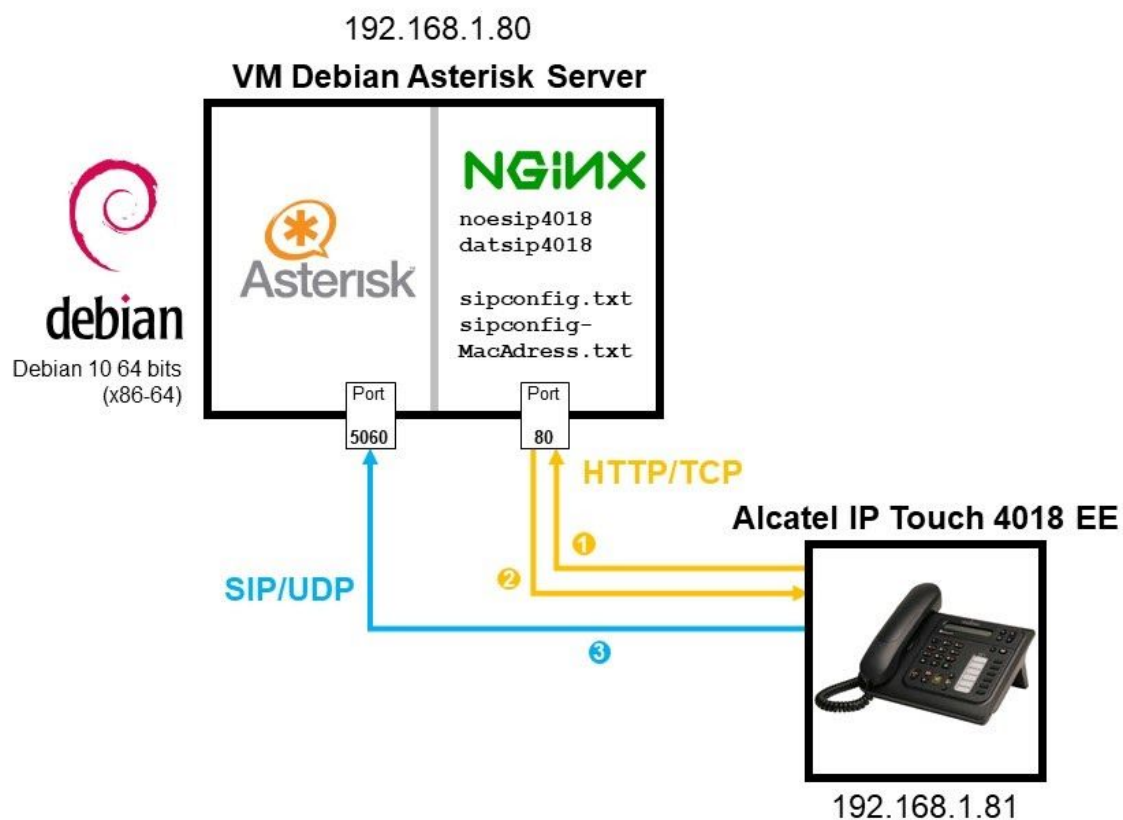
Il faut en premier lieu, le configurer en mode SIP. Ensuite, il a la particularité de se configurer automatiquement en téléchargeant automatiquement ses fichiers de configuration et ses fichiers de firmwares sur un serveur *TFTP*, *HTTP* ou *HTTPS* au moment du démarrage.

Parmi les trois protocoles, le plus sécurisé est l'*HTTPS*. Nous avons essayé de le mettre en place avec des certificats RSA 2048 bits et des *cipher suites* compatibles avec des équipements anciens. Malheureusement, étant en réseau local, le téléphone Alcatel semble ne pas accepter les certificats auto signés¹⁴. Compte-tenu de cette problématique, il ne reste que deux choix : *TFTP* ou *HTTP*. Ces protocoles applicatifs ne sont pas très sécurisés : pas d'authentification, transfert des données en clair sur le réseau... Le choix se fait donc au niveau des protocoles de transport : *TFTP* utilise nécessairement *UDP*, mode non connecté alors que *HTTP* peut être configuré pour utiliser *TCP*, mode connecté, qui intègre une détection et correction des erreurs. Les programmes transmis sont des firmwares, si des données sont corrompues à cause d'une erreur, cela pourrait bricker le téléphone. Privilégions donc le protocole *HTTP* associé au protocole *TCP*. Nous allons donc mettre en place un serveur *HTTP NGINX*¹⁵ car léger, dont le seul but est de fournir les firmwares et les fichiers de configuration au téléphone Alcatel IP Touch. Il sera accessible à l'adresse **192.168.1.80** au niveau du port **80**.

Bien évidemment, si cela est possible, pour une mise en production, nous recommandons d'utiliser *HTTPS*.

¹⁴ Selon ce forum, il semble que les téléphones Alcatel IP Touch ne fassent confiance qu'à l'autorité de certification délivrée par l'entreprise Alcatel-Lucent. Le certificat racine issu d'Alcatel-Lucent semble se trouver au sein du téléphone : <https://www.alcatelunleashed.com/viewtopic.php?t=28822>. Nous n'avons pas les moyens d'obtenir un certificat à partir de l'autorité de certification d'Alcatel-Lucent.

¹⁵ Serveur HTTP open source NGINX : <https://www.nginx.com/>.



(Figure 13 - Récupération des fichiers de configuration du téléphone Alcatel via HTTP)

Informations sur la configuration

Adresse IP du téléphone Alcatel. Ceci est à configurer sur le routeur (cf. annexe A1.15).	192.168.1.81
Adresse IP et port du serveur HTTP	192.168.1.80:80
Alimentation PoE reliée au routeur	TP-Link TL-POE150S ¹⁶

¹⁶ TP-Link TL-POE150S : <https://www.tp-link.com/fr/business-networking/accessory/tl-poe150s/>.

Configuration du téléphone en mode SIP - sur l'appareil

Brancher le téléphone à l'injecteur PoE puis suivre les instructions du manuel d'installation ci-dessous (captures d'écrans de la procédure).

3. SIP stand-alone mode

3.1. Firmware version identification

In addition to existing call processing support for the Alcatel-Lucent OmniPCX™ Communication Servers (NOE signaling protocol), the Alcatel-Lucent IP Touch 40x8EE phone can also be configured as Session Initiation Protocol (SIP) endpoints and, therefore, operate in a standard SIP environment. The Alcatel-Lucent IP Touch 40x8 EE phone may also switch to SIP survival mode in SIP standalone mode.

Firmware version identification - phone out of the box.

In this context, the phone is natively configured in NOE mode

- 1- Power on the phone.
- 2- At the phase 2/5 network setup, press the "i" key and just after press the "#" key.
- 3- '0 - MAC address' menu appears.
- 4- Scroll down to '2 - Soft infos'.
- 5- Press the OK key.
- 6- The version is displayed : 'Version NOE 4.xx.xx'. If the version is at least 4.xx.xx, SIP boot mode is possible.

Firmware version identification - phone already installed.

- 1- Power on the phone.
- 2- At the phase 2/5 network setup, press the "i" key and just after the "#" key.
- 3- '0 - MAC address' menu appears.
- 4- Scroll down to '2 - Soft infos'.
- 5- Press the OK key.
- 6- Scroll down to 'Run mode':
 - if NOE is displayed, it means that the NOE signaling protocol is running,
 - if SIP is displayed, it means that the SIP signaling protocol mode is running.

3.2. Switching From NOE To SIP Mode

There are two methods.

Method 1 - Manual

- 1- Power on the phone.
- 2- At the phase 2/5 network setup, press the "i" key and just after the "#" key.
- 3- '0 - MAC address' menu appears.
- 4- Scroll down to '2 - Soft infos'.
- 5- Press OK.
- 6- 'Version NOE x.xx.xx' is displayed.
- 7- Scroll down to 'Run mode', 'Set Mode: NOE' is displayed.
- 8- Press OK to switch from NOE to SIP mode.
- 9- Press '#' to save the new configuration.
- 10- Press the Hang Up key to reset the device to take into account the new configuration.

For checking SIP status, follow 'Firmware version identification already installed' section.

(Figure 14 - Alcatel-Lucent, 3. SIP stand-alone mode In : IP Touch 4008/4018 Extended Edition - SIP Phone
Installation Guide - 8AL90824AAAA ed02, p.4-5, Août 2010, Disponible sur :

<https://www.cluster2.hostgator.co.in/files/writeable/uploads/hostgator136107/file/iptouchsipphoneinstallationguide-ed02.pdf>)

Une fois passé en mode SIP, il faut configurer l'adresse IP du téléphone afin qu'il soit en accord avec celui défini dans le routeur (**192.168.1.81**), il faut également lui fournir l'adresse IP du serveur *HTTP* et son port.

1. Brancher le téléphone à l'injecteur PoE.
2. À la phase **2/5 network setup**, appuyer sur **"i"** puis sur **"#"** jusqu'à l'apparition du menu **"MAC address"**.
3. S'il y a un mot de passe, entrer **000000**.
4. Descendre jusqu'à **IP Parameters**, puis cliquer sur **OK**.
5. Descendre, puis sélectionner **IP mode: Static**.
6. Descendre, puis entrer l'adresse IP : **IP@: 192.168.1.81**
7. Descendre, puis entrer le masque de sous réseau : **Subnet: 255.255.255.0**
8. Descendre, puis entrer l'adresse IP du routeur : **Router: 192.168.1.254**
9. Descendre, puis sélectionner dans DL Scheme: **HTTP**.
10. Descendre, puis sélectionner **Use Defaultport**.
11. Descendre, puis sélectionner l'adresse du serveur *HTTP* : **DL Addr: 192.168.1.80**
12. Descendre, puis sélectionner le port du serveur *HTTP* : **DL Port: 80**
13. Descendre, ne pas sélectionner de VLAN.
14. Descendre jusqu'à sauvegarder **save**, puis cliquer sur **OK**.

La configuration sur l'appareil est terminée, cependant pour l'instant il ne va cesser de redémarrer en boucle puisque le serveur *HTTP* n'existe pas encore, ni les fichiers de configuration associés et les firmwares du téléphone.

Installation du serveur HTTP

On va donc revenir sur notre machine virtuelle Debian et installer un serveur *HTTP*.

1. Se connecter en SSH à la machine `asterisktz`.

```
ssh asterisktz@192.168.1.80 -p 22
```

2. Installer le serveur *HTTP* **nginx**.

```
asterisktz@asterisktz:~$
```

Installation du serveur **nginx**

```
sudo apt install nginx
```

```
asterisktz@asterisktz:~$
```

Vérification de son état

```
sudo systemctl status nginx
```

S'il est **active**, c'est qu'il fonctionne sinon, il faut le lancer (`sudo systemctl start nginx`). Normalement il est configuré pour se lancer automatiquement au démarrage, si tel n'est pas le cas alors exécuter la commande suivante :

```
sudo /lib/systemd/systemd-sysv-install enable nginx
```

Le serveur *HTTP* est prêt, il ne reste plus qu'à transférer les firmwares, les informations sur le compte SIP dédié au téléphone Alcatel et l'adresse IP du serveur Asterisk.

Transfert des fichiers de configuration et des firmwares vers le téléphone Alcatel

Concrètement il y a 4 fichiers à transférer sur le serveur HTTP (10) :

- ***sipconfig.txt*** : fichier global de configuration, contient les paramètres à appliquer à tous les téléphones Alcatel IP Touch 4018EE connectés au réseau.
- ***sipconfig-MacAddress.txt*** : fichier de configuration spécifique à un seul téléphone Alcatel IP Touch 4018EE. Il s'agit de l'adresse MAC du téléphone en question qui doit être écrite en minuscules.
- ***noesip4018*** : firmware propriétaire contenant l'application gérant le protocole SIP pour l'Alcatel IP Touch 4018EE.
- ***datsip4018*** : ressources contenant les sonneries et différentes mélodies.

Le site du fabricant Alcatel-Lucent ne proposant pas forcément une documentation appropriée pour connaître la structure des fichiers ***sipconfig.txt***, nous nous sommes basés sur un fichier d'exemple de configuration réalisé par Florian Duraffourg, diplômé de Télécom SudParis :

<https://github.com/fduraffourg/utls/blob/master/iptouch/sipconfig-reynoud.txt>

Concernant les firmwares, également difficiles à trouver sur le site officiel d'Alcatel-Lucent, nous les avons trouvés sur le forum Alcatel Unleashed¹⁷ par le biais du post de *fbird* le 21 mars 2016 :

<https://www.alcatelunleashed.com/viewtopic.php?p=95015#p95015>

Le fichier le plus important est ***sipconfig-MacAddress.txt*** dont le contenu important en gras et en vert est le suivant (nous avons volontairement enlevé les parties non essentielles, vous retrouverez le fichier complet dans le dépôt du projet disponible au paragraphe suivant) :

Remarque : nous avons ajouté des commentaires dans ce rapport, pour éviter les erreurs prendre celui du dépôt.

sipconfig-MacAddress.txt (commenté)

```
[...]  
  
[dns]  
  
#####  
## The primary DNS IP address HAS TO BE FILLED  
## If no DNS, use the SIP proxy address instead  
#####  
  
    dns_addr=192.168.1.254  # on prend le DNS de la Freebox  
    dns2_addr=  
    hostname=192.168.1.254  
  
[sip]
```

¹⁷ The #1 Worldwide board for technical support on Alcatel-Lucent Voice & Data gear.

```
#####
## Domain name : IP address, FQDN or domain name (see the SIP proxy config)
#####

    domain_name=192.168.1.80 # domaine du serveur Asterisk

#####
## Primary SIP proxy and SIP registrar settings
##
## Proxy address : IP address, FQDN or domain name
## Registrar address : IP address, FQDN or domain name (usually, the proxy)
## SIP proxy UDP port : usually 5060
## SIP registrar UDP port : by default 5060
#####

    proxy_addr=192.168.1.80
    proxy_port=5060
    registrar_addr=192.168.1.80
    registrar_port=5060
    outbound_proxy_addr=
    outbound_proxy_port=

#####
## Redundancy settings
##
## Proxy address : IP address, FQDN or domain name
## Registrar address : IP address, FQDN or domain name (usually, the proxy)
## SIP proxy UDP port : usually 5060
## SIP registrar UDP port : by default 5060
## sip_transport_mode_survi : Transport mode in PCS mode
##         0 = UDP or TCP
##         1 = UDP
##         2 = TCP
#####

    proxy2_addr=192.168.1.80
    proxy2_port=5060
    registrar2_addr=192.168.1.80
    registrar2_port=5060
    outbound_proxy2_addr=
    outbound_proxy2_port=
    pcs_addr=192.168.1.80
    pcs_port=5060
    sip_transport_mode_survi=0 # dans notre cas, ce sera de l'UDP
    option_timer=120

#####
## Global SIP parameters
## Transport mode : 0 = UDP or TCP
##         1 = UDP
##         2 = TCP
## local_rtp_port : RFC3605 is not supported in this release, so
##         only default value can be used
## PRACK type : 0 = PRACK supported
##         1 = PRACK required
##         2 = PRACK disabled
## Codec settings : 0 = G711 (PCMU)
##         4 = G723.1
##         8 = G711 (PCMA)
##         18 = G729A
#####

    register_expire=3600
```

```
register_retry=300
local_sip_port=
sip_transport_mode=0 # dans notre cas, ce sera de l'UDP
local_rtp_port=42000
local_rtcp_port=42001
prack_type=0
preferred_vocoder=8,0,4,18

#####
## SIP authentication.
##
## Realm : If no authentication, leave empty
## Authentication name : HAS TO BE FILLED
##                               If no authentication, PUT A VALUE LIKE none
## Authentication password : If no authentication, leave empty
#####

authentication_realm=192.168.1.80 # l'authentification se fait sur Asterisk
authentication_name=alcatel      # nom d'utilisateur puis mot de passe
authentication_password=11111111
user_name=alcatel
display_name=Alcatel IP Touch   # nom d'affichage quand on appelle

[...]

[sntp]

#####
## SNTP server settings (can be OXE or an external server)
##
## Timezone construction : UT::60:032802:103103 (Paris - 2021)
##       GMT delta : 60 = + sixty minutes from GMT time
##       Daylight saving start (mmddhh) : 032902 = 28 March 2am
##       Daylight saving end (mmddhh) : 103103 = 31 October 3am
## The daylight saving settings HAVE to be changed each year.
#####

sntp_addr=192.168.1.254          # Pour synchroniser l'heure du téléphone avec
timezone=UT::60:032802:103103   # le serveur NTP intégré de la Freebox
                                # La timezone est à changer chaque année, elle
                                # est réglée en fonction de la zone GMT et gère l'heure d'été et l'heure d'hiver.

[...]

[init]

#####
## For IP Touch with SIP binary in 1.xx, 2.00.10 and 2.00.20, equal or greater
## than 2.00.81
##       mode 0 = SIP
##       mode 1 = NOE
##
## For IP Touch with SIP binary 2.00.30 to 2.00.80
##       mode 0 = NOE
##       mode 1 = SIP
#####

application_mode=0              # Mode SIP (en fonction des versions des firmwares)

[audio]

#####
## Tone country : 0 = English
```

```
##          1 = French
##          2 = German
##          3 = Italian
##          4 = Spanish
##          5 = Dutch
##          6 = Portuguese
## DTMF type : 0 = RFC2833
##          1 = In-band
##          2 = SIP INFO
## DTMF level / RLR handset / SLR handset / Sidetone handset :
## 0 = 0db, 1 = +3db, 2 = +6db, 3 = -3db, 4 = -6db
## VAD / DTMF feedback / Hearing Aid :
## 0 = VAD not used
## 1 = VAD used
#####

tone_country=1    # à régler en fonction des normes utilisés en France
dtmf_type=1
dtmf_level=0
dtmf_avt_payload_type=96
vad=0
dtmf_feedback_enable=0
rlr_handset=0
slr_handset=0
sidetone_handset=2
hearing_aid_enable=0

[appl]

#####
## Password to access the administrator menu on the phone (digits only)
## Power priority : 1 = critical
##          2 = high
##          3 = low
## Time format : 0 = 24 hours format
##          1 = AM / PM
## Speed dial numbers (first and last name, URI)
#####

admin_password=000000    # mot de passe admin lorsque on appuie sur i puis #
bluetooth_parameters=blue
supported_language=0
remote_forward_code=
remote_forward_deactive_code=
power_priority=
asset_id=
time_format=0
speed_dial_1_first_name=
speed_dial_1_last_name=
speed_dial_1_uri=
speed_dial_2_first_name=
speed_dial_2_last_name=
speed_dial_2_uri=
speed_dial_3_first_name=
speed_dial_3_last_name=
speed_dial_3_uri=
speed_dial_4_first_name=
speed_dial_4_last_name=
speed_dial_4_uri=

[...]
```

1. Télécharger l'ensemble de ces 2 fichiers de firmware (*noesip4018* et *dat sip4018*, trouvés sur le forum Alcatel Unleashed) et les 2 fichiers de configuration (*sipconfig.txt* et *sipconfig-MacAddress.txt*), disponibles dans le dépôt du projet :
<https://gitlab.utc.fr/nibertgu/tz-voip-raspberry-pi/-/tree/master/iptouch4018ee>
2. Les placer dans le dossier `/var/www/html` de la machine virtuelle Debian.
3. Brancher le téléphone à l'injecteur PoE pour l'allumer. L'Alcatel IP Touch va chercher les derniers firmwares sur le serveur *HTTP NGINX*, les configurations et sera connecté au serveur *Asterisk*.

Vérification côté client

S'il affiche la date et l'heure et qu'il ne redémarre pas en boucle c'est qu'il est bien connecté au serveur *Asterisk*.

Vérification côté serveur

Sur une console serveur, taper la commande `sudo asterisk -rvvv`

Une fois entrée, taper la commande `pjsip show endpoints`. Si le client SIP du Raspberry Pi est bien connecté alors la console renvoie ceci :

Output

```
asterisktz*CLI> pjsip show endpoints

Endpoint: <Endpoint/CID.....> <State.....>
<Channels.>
  I/OAuth:
<AuthId/UserName.....>
  Aor: <Aor.....> <MaxContact>
  Contact: <Aor/ContactUri.....> <Hash.....> <Status>
<RTT(ms) ..>
  Transport: <TransportId.....> <Type> <cos> <tos>
<BindAddress.....>
  Identify:
<Identify/Endpoint.....>
  Match: <criteria.....>
  Channel: <ChannelId.....> <State.....>
<Time.....>
  Exten: <DialedExten.....> CLCID: <ConnectedLineCID.....>
=====
==

Endpoint:  alcatel/5001                                Not in use      0 of inf
InAuth:    alcatel/alcatel
Aor:       alcatel                                     1
Contact:   alcatel/sip:alcatel@192.168.1.81           8c02c0558c NonQual  nan

Endpoint:  guillaume/5003                                Unavailable     0 of inf
InAuth:    guillaume/guillaume
Aor:       guillaume                                  1
```

```
Endpoint:  rpi/5002                                Not in use      0 of inf
InAuth:    rpi/rpi
Aor:       rpi                                      1
Contact:   rpi/sip:rpi@192.168.1.82;transport=udp  cec2f9dd2f NonQual  nan
Objects found: 3
```

On remarque bien que le client a pour adresse IP **192.168.1.81** et qu'il est bien connecté. Le **Not in use** indique qu'il n'y a pas d'appel en cours. L'objectif de la partie suivante est de réaliser une communication entre les deux appareils.

5. Tests de communication

Tous les points de terminaison sont connectés au serveur Asterisk. Il est donc possible d'appeler du Raspberry Pi vers l'Alcatel ou de l'Alcatel vers le Raspberry Pi.

Afin de réaliser la communication, il faut préalablement avoir lancé le serveur Asterisk, le serveur TFTP, allumé tous les périphériques et branché une sortie audio sur le Raspberry Pi (jack, Bluetooth ou HDMI) pour pouvoir écouter le flux audio.

Raspberry Pi vers Alcatel IP Touch

1. Lancer un terminal puis exécuter *linphonec*.

```
pi@raspberrypi:~$
```

Exécution de *linphonec*

```
linphonec
```

2. Appeler le téléphone Alcatel IP Touch, il a pour numéro 5001 (cf. **partie 2 - Configuration de SIP et création d'utilisateurs**).

```
linphonec> call 5001
```

Ouput (commenté)

```
# Message d'erreur non important, la vidéo est bien désactivée on ne fait que de la VoIP.
2021-02-09 13:30:36:367 ortp-error-LinphoneCore has video disabled for both capture and display, but video policy is to start the call with video. This is a possible mis-use of the API. In this case, video is disabled in default LinphoneCallParams

# Établissement de lien vers l'Alcatel
Establishing call id to sip:5001@192.168.1.80, assigned id 1
# L'Alcatel a été trouvé, il est contacté, ça sonne du côté de l'Alcatel.
Contacting sip:5001@192.168.1.80
linphonec> Call 1 to sip:5001@192.168.1.80 in progress.
linphonec> Call 1 with sip:5001@192.168.1.80 connected.
# Nous avons décroché le téléphone Alcatel.
Call answered by sip:5001@192.168.1.80
# La communication est en cours, l'audio passe, les paramètres sont ajustés.
linphonec> 2021-02-09 13:30:36:563 ortp-error-no such method on filter MSPulseWrite, fid=16394 method index=2
Media streams established with sip:5001@192.168.1.80 for call 1 (audio).
Call is updated by remote.
linphonec> 2021-02-09 13:30:40:761 ortp-error-no such method on filter MSPulseWrite, fid=16394 method index=2
Call parameters were successfully modified.
linphonec> Media streams established with sip:5001@192.168.1.80 for call 1 (audio).
Call is updated by remote.
linphonec> Call parameters were successfully modified.
```

```
linphonec> Media streams established with sip:5001@192.168.1.80 for call 1  
(audio).  
# La communication vient de se terminer, quelqu'un a raccroché un des appareils.  
Call terminated.  
linphonec> Call 1 with sip:5001@192.168.1.80 ended (No error).
```

Au moment où l'appel est lancé, voici ce que l'écran du téléphone Alcatel affiche :



(Figure 15 - Appel de Raspberry Pi vers Alcatel IP Touch 4018 EE)

La communication fonctionne donc dans un sens. Voyons ce que cela donne si c'est l'Alcatel qui appelle le Raspberry Pi.

Alcatel IP Touch vers Raspberry Pi

1. Depuis le téléphone, appeler le 5002, numéro du Raspberry Pi (cf. **partie 2 - Configuration de SIP et création d'utilisateurs**).



(Figure 16 - Appel de Alcatel IP Touch 4018 EE vers Raspberry Pi)

2. Depuis le terminal Raspberry Pi, veiller à ce que **linphonec** soit actif. Au moment où l'Alcatel lance son appel, on le reçoit dans le terminal :

Output

```
linphonec> Receiving new incoming call from "Alcatel IP Touch"  
<sip:5001@192.168.1.80>, assigned id 3
```

Pour y répondre, il suffit de taper **answer** et l'**id** de l'appel :

```
answer 3
```

La communication est lancée et fonctionne de la même manière.

Output

```
linphonec> Receiving new incoming call from "Alcaltel IP Touch"
<sip:5001@192.168.1.80>, assigned id 3
answer 3
Connected.
linphonec> Call 3 with "Alcaltel IP Touch" <sip:5001@192.168.1.80> connected.
2021-02-09 13:46:28:345 ortp-error-no such method on filter MSPulseWrite,
fid=16394 method index=2
Media streams established with "Alcaltel IP Touch" <sip:5001@192.168.1.80> for
call 3 (audio).
linphonec> Call is updated by remote.
linphonec> 2021-02-09 13:46:28:424 ortp-error-no such method on filter
MSPulseWrite, fid=16394 method index=2
Call parameters were successfully modified.
linphonec> Media streams established with "Alcaltel IP Touch"
<sip:5001@192.168.1.80> for call 3 (audio).
Call is updated by remote.
linphonec> Call parameters were successfully modified.
linphonec> Media streams established with "Alcaltel IP Touch"
<sip:5001@192.168.1.80> for call 3 (audio).
Call terminated.
linphonec> Call 3 with "Alcaltel IP Touch" <sip:5001@192.168.1.80> ended (No
error).
```

Les communications fonctionnent donc dans les deux sens. L'objectif de la partie suivante est donc de réaliser une interface graphique en JavaScript côté Raspberry Pi, plus conviviale que le client *linphonec* en ligne de commande.

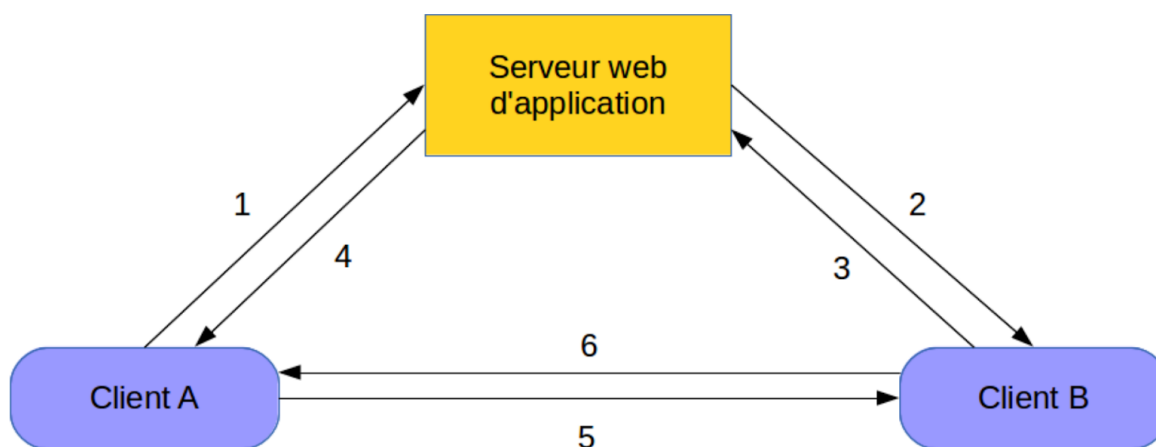
6. Développement d'un programme client SIP en JavaScript

L'établissement de la communication (SIP) ainsi que la communication en elle-même (RTP) fonctionnent correctement. Le développement d'un programme client SIP en JavaScript va nécessiter des modifications de notre environnement. En effet, nous allons donc passer un appel depuis un navigateur web supportant JavaScript vers un autre appareil (ayant un client SIP JavaScript ou non). Avant de commencer la mise en place. Il est nécessaire de comprendre ce que sont les API WebRTC, WebSocket.

WebRTC & WebSocket

L'API WebRTC (*Web Real-Time Communication*) est une interface logicielle dont le but est de mettre en relation deux périphériques afin qu'ils puissent communiquer directement. Cette mise en relation nécessite d'ouvrir un canal de communication entre un client et un serveur : la technologie qui permet de répondre à cela est l'API WebSocket.

Concrètement, l'établissement d'une connexion fonctionne similairement à SIP. Ci-dessous, voici un schéma explicatif issu de Wikipédia.



(Figure 17 - Établissement d'une connexion entre deux clients utilisant WebRTC)

- “1 : A demande au serveur une connexion avec B.
- 2 : Le serveur relaie la demande de A à B.
- 3 : Si B accepte, il envoie une demande de connexion à A.
- 4 : Le serveur relaie la demande à A.
- 5 et 6 : Les PeerConnection bidirectionnelles sont établies.” - [Wikipédia](#).

Les PeerConnection correspondent dans notre cas au flux RTP entre les deux clients. Une fois la communication établie, comme pour SIP la communication entre les deux clients est directe et les flux médias ne passent pas par le serveur web d'application.

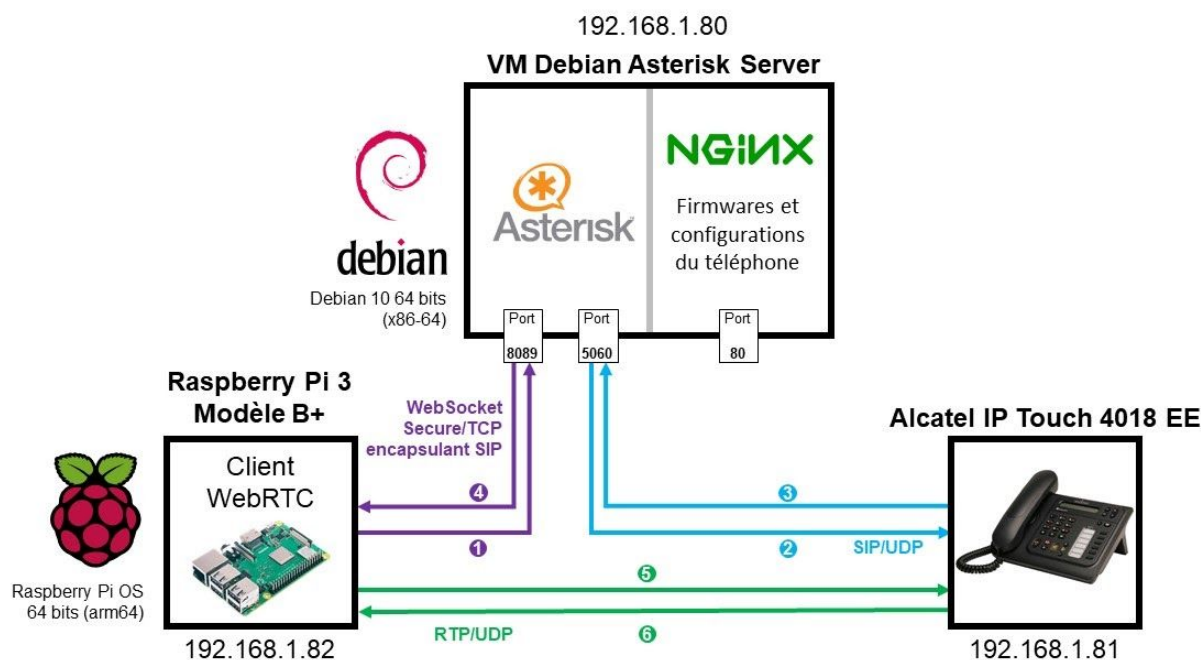
Imaginons maintenant :

- client A : Raspberry Pi, disposant du navigateur web Mozilla Firefox prenant en charge WebRTC.
- client B : Alcatel IP Touch 4018 EE, ne prenant pas en charge WebRTC.

Comment faire communiquer les deux points de terminaison ?

On ne peut pas faire en sorte que le téléphone Alcatel puisse prendre en charge WebRTC, c'est du matériel propriétaire, le code source est fermé, et même si cela était possible, cela prendrait nécessiterait une durée de travail non négligeable afin de comprendre comment le logiciel fonctionne, programmer de nouveaux firmwares et configurer le système.

En revanche, côté Raspberry Pi, il est possible d'utiliser WebRTC et SIP en encapsulant le protocole SIP dans une WebSocket. Cela est défini dans la RFC 7118¹⁸. Il faut également un serveur capable de gérer WebRTC/SIP pour le client A et seulement SIP pour le client B. Le serveur Asterisk prend en charge WebRTC avec SIP. Il faut donc apporter des modifications pour que le serveur puisse prendre en charge WebRTC.



(Figure 18 - Raspberry Pi appelle Alcatel IP Touch depuis un client WebRTC)

¹⁸ RFC 7118 : *The WebSocket Protocol as a Transport for the Session Initiation Protocol (SIP)*, Disponible sur : <https://tools.ietf.org/html/rfc7118>.

Au niveau du client Raspberry Pi, les navigateurs web tels que Mozilla Firefox, Safari ou encore ceux basés sur Chromium implémentent nativement l'API WebRTC. Nous utiliserons Mozilla Firefox comme client utilisant WebRTC.

Configuration du serveur Asterisk pour prendre en charge l'API WebRTC

Afin d'améliorer la sécurité entre le client WebRTC et le serveur Asterisk, nous allons mettre en place une web socket sécurisée (WSS) via TLS. Nous allons donc dans un premier temps générer un certificat auto signé.

Génération d'un certificat SSL/TLS auto signé

Dans un but d'amélioration de la sécurité et de modernisation, nous adoptons un certificat généré avec des algorithmes ECDSA, bien plus performants que les algorithmes RSA classiques **(11)**. Nous allons utiliser l'algorithme ECDSA P-521, recommandé par l'ANSSI **(12)** et compatible avec Mozilla Firefox¹⁹.

Pour réaliser cette opération, il faut avoir au préalable démarré la machine virtuelle Debian et l'outil OpenSSL.

¹⁹ Mozilla Firefox utilise NSS (Network Security Services), bibliothèque prenant en charge cet algorithme.

1. Se connecter en SSH à la machine **asterisktz**.

```
# ssh login@adresse_ip_vm -p 22  
ssh asterisktz@192.168.1.80 -p 22
```

2. Créer les dossiers dans lesquels seront stockés le certificat de l'Autorité de Certification, appelé certificat racine (**ca**), le certificat associé à l'IP 192.168.1.80 (**certs**) et le fichier de demande de signature de certificat à l'autorité (**csr**).

```
asterisktz@asterisktz:~$
```

Création des dossiers

```
mkdir ca && mkdir certs && mkdir csr
```

3. Création des certificats.

```
asterisktz@asterisktz:~$
```

Création de la clef privée du certificat racine (autorité de certification)

```
openssl ecparam -genkey -name secp521r1 -out ca/TZVoIP-Root-CA.key
```

```
asterisktz@asterisktz:~$
```

Génération du certificat racine à partir de sa clef privée

```
openssl req -x509 -new -nodes -key ca/TZVoIP-Root-CA.key -sha384 -days  
3650 -utf8 -out ca/TZVoIP-Root-CA.crt
```

Informations à renseigner

```
You are about to be asked to enter information that will be incorporated  
into your certificate request.  
What you are about to enter is what is called a Distinguished Name or a DN.  
There are quite a few fields but you can leave some blank  
For some fields there will be a default value,  
If you enter '.', the field will be left blank.  
-----
```

```
Country Name (2 letter code) [AU]:FR  
State or Province Name (full name) [Some-State]:Hauts-de-France  
Locality Name (eg, city) []:Compiègne  
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Université de  
Technologie de Compiègne  
Organizational Unit Name (eg, section) []:TZ VoIP  
Common Name (e.g. server FQDN or YOUR name) []:TZ VoIP Root  
Email Address []:guillaume.nibert@etu.utc.fr
```

```
asterisktz@asterisktz:~$
```

Génération de la clé privée du certificat de l'IP et de son fichier de
demande de signature

```
openssl req -new -sha384 -nodes -utf8 -out csr/asterisktz.csr -newkey  
ec:<(openssl ecparam -name secp521r1) -keyout certs/asterisktz.key
```

Informations à renseigner

```
Generating an EC private key
writing new private key to 'certs/asterisktz.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:FR
State or Province Name (full name) [Some-State]:Hauts-de-France
Locality Name (eg, city) []:Compiègne
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Université de
Technologie de Compiègne
Organizational Unit Name (eg, section) []:TZ VoIP
Common Name (e.g. server FQDN or YOUR name) []:192.168.1.80
Email Address []:guillaume.nibert@etu.utc.fr

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

asterisktz@asterisktz:~\$ Création du fichier contenant les paramètres du certificat à créer

```
nano csr/openssl-v3.cnf
```

csr/openssl-v3.cnf

```
authorityKeyIdentifier=keyid,issuer
basicConstraints=CA:FALSE
keyUsage = digitalSignature, nonRepudiation, keyEncipherment,
dataEncipherment
subjectAltName = @alt_names

[alt_names]
IP.1 = 192.168.1.80
```

asterisktz@asterisktz:~\$ Génération du certificat et signature auprès de l'autorité de certification

```
openssl x509 -req -in csr/asterisktz.csr -CA ca/TZVoIP-Root-CA.crt -CAkey
ca/TZVoIP-Root-CA.key -CAcreateserial -out certs/asterisktz.crt -days 365
-sha384 -extfile csr/openssl-v3.cnf
```

asterisktz@asterisktz:~\$ Fabrication du certificat full-chain

```
cat certs/asterisktz.crt certs/asterisktz.key > certs/asterisktz.pem
```

asterisktz@asterisktz:~\$ Restriction à des droits de lecture uniquement sur ce certificat pour

empêcher toute modification

```
chmod a+r certs/asterisktz.pem
```

Le certificat auto signé a été créé. Pour plus de détails concernant le processus de certification dans le cas d'un certificat non autosigné, consulter : <https://letsencrypt.org/fr/how-it-works/>. Passons à l'activation du serveur HTTP intégré à Asterisk.

Activation du serveur HTTP d'Asterisk

Sur la même machine :

1. Éditer le fichier de configuration du serveur HTTP intégré à Asterisk.

asterisktz@asterisktz:~\$ Édition du fichier de configuration du serveur HTTP intégré à Asterisk

```
sudo nano /etc/asterisk/http.conf
```

2. Remplacer le contenu par :

/etc/asterisk/http.conf

```
[general]
enabled=no
tlsenable=yes
tlsbindaddr=0.0.0.0:8089
tlscertfile=/home/asterisktz/certs/asterisktz.crt
tlsprivatekey=/home/asterisktz/certs/asterisktz.key
enablestatic=no
sessionlimit=1000
```

Ici on n'autorise que des connexions chiffrées (***tlsenable=yes***) sur le port 8089. Les connexions non chiffrées seraient activables en passant ***enabled*** à ***yes*** sur le port 8088.

3. Redémarrer le service Asterisk pour activer le serveur HTTP intégré.

asterisktz@asterisktz:~\$

Application des changements

```
sudo systemctl restart asterisk
```

Pour vérifier l'activation du serveur HTTP, il suffit de taper la commande ***sudo asterisk -rvvv***, puis la commande ***http show status***. Elle devrait renvoyer ceci :


```
asterisktz*CLI> http show status
HTTP Server Status:
Prefix:
Server: Asterisk/18.2.0
Server Disabled

Enabled URI's:
/httpstatus => Asterisk HTTP General Status
/phoneprov/... => Asterisk HTTP Phone Provisioning Tool
/metrics/... => Prometheus Metrics URI
/ari/... => Asterisk RESTful API
/ws => Asterisk HTTP WebSocket

asterisktz*CLI>
```

En vert, c'est l'élément qui nous intéresse : l'utilisation du WebSocket pour SIP. Passons maintenant à la configuration de *pjsip.conf* et de *extensions.conf* pour prendre en compte à la fois WebRTC et SIP. Nous nous sommes basés sur le projet *BrowserPhone*²⁰ et avons adapté les fichiers de configurations en question. Ces fichiers sont disponibles dans le répertoire *asterisk_webrtc* du dépôt GitLab.

Modification de *pjsip.conf* pour prendre en charge WebRTC

1. Éditer le fichier de configuration *pjsip.conf*.

```
asterisktz@asterisktz:~$
```

Édition du fichier de configuration *pjsip.conf*

```
sudo nano /etc/asterisk/pjsip.conf
```

2. Remplacer le contenu par :

/etc/asterisk/pjsip.conf

```
[global]
max_forwards=70
user_agent=AsteriskTZ
default_realm=192.168.1.80
keep_alive_interval=300

; == Transports

[udp_transport]
type=transport
protocol=udp
bind=0.0.0.0
tos=af42
cos=3
```

²⁰ BrowserPhone : <https://github.com/InnovateAsterisk/Browser-Phone>.

```
[wss_transport]
type=transport
protocol=wss
bind=0.0.0.0

[tcp_transport]
type=transport
protocol=tcp
bind=0.0.0.0

[tls_transport]
type=transport
protocol=tls
bind=0.0.0.0
cert_file=/home/asterisk/certs/asterisktz.crt
priv_key_file=/home/asterisk/certs/asterisktz.key
cipher=ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA
384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:D
HE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384;
method=tlsv1_2

; == ACL

[ac1] ; Les communications sont uniquement autorisées dans réseaux locaux de classes A,
type=ac1 ; B et C.
deny=0.0.0.0/0.0.0.0
permit=10.0.0.0/255.0.0.0
permit=172.16.0.0/255.240.0.0
permit=192.168.0.0/255.255.0.0

; Modèles

[single_aor](!)
max_contacts=1
qualify_frequency=120
remove_existing=yes

[userpass_auth](!)
auth_type=userpass

[basic_endpoint](!)
moh_suggest=default
context=from-extensions
inband_progress=no
rtptimeout=120
message_context=textmessages
allow_subscribe=yes
subscribe_context=subscriptions
direct_media=yes
dtmf_mode=rfc4733
```

```
device_state_busy_at=1
disallow=all

[phone_endpoint] (!)
allow=ulaw,alaw

[webRTC_endpoint] (!)
transport=wss_transport
allow=ulaw,alaw
dtls_auto_generate_cert=yes
webRTC=yes

; Utilisateurs

[alcatel] (basic_endpoint,phone_endpoint)
type=endpoint
callerid="Alcatel IP Touch" <5001>
auth=alcatel
aors=alcatel
[alcatel] (single_aor)
type=aor
[alcatel] (userpass_auth)
type=auth
username=alcatel
password=11111111

[rpi] (basic_endpoint,webRTC_endpoint)
type=endpoint
callerid="Raspberry Pi" <5002>
auth=rpi
aors=rpi
[rpi] (single_aor)
type=aor
[rpi] (userpass_auth)
type=auth
username=rpi
password=22222222

[guillaume] (basic_endpoint,webRTC_endpoint)
type=endpoint
callerid="Guillaume Nibert" <5003>
auth=guillaume
aors=guillaume
[guillaume] (single_aor)
type=aor
mailboxes=guillaume@default
[guillaume] (userpass_auth)
type=auth
username=guillaume
password=33333333
```

Pour davantage de détails, se référer à la documentation de PJSIP :

<https://wiki.asterisk.org/wiki/display/AST/PJSIP+Configuration+Sections+and+Relationships>.

Modification de *extensions.conf* pour prendre en charge WebRTC

3. Éditer le fichier de configuration *extensions.conf*.

```
asterisktz@asterisktz:~$
```

Édition du fichier de configuration *extensions.conf*

```
sudo nano /etc/asterisk/extensions.conf
```

4. Remplacer le contenu par :

/etc/asterisk/extensions.conf

```
[general]
static=yes
writeprotect=yes
priorityjumping=no
autofallthrough=no

[globals]
ATTENDED_TRANSFER_COMPLETE_SOUND=beep

[textmessages] ; Permet en plus d'envoyer du texte pour les clients WebRTC
exten => 5002,1,Gosub(send-text,s,1(rpi))
exten => 5003,1,Gosub(send-text,s,1(guillaume))

[subscriptions] ; Permet de connaître l'état d'un point de terminaison (en appel ou
exten => 5001, hint, PJSIP/alcatel ; disponible)
exten => 5002, hint, PJSIP/rpi
exten => 5003, hint, PJSIP/guillaume

[from-extensions]
; Lorsqu'on appelle le 5000, on a de la musique
exten => 5000,1,Gosub(moh,s,1)
; Extensions
exten => 5001,1,Gosub(dial-extension,s,1,(alcatel))
exten => 5002,1,Gosub(dial-extension,s,1,(rpi))
exten => 5003,1,Gosub(dial-extension,s,1,(guillaume))
; Si on a autre chose que le 5000, 5001, 5002 ou 5003 alors c'est un faux numéro, donc on
; raccroche
exten => _[+*0-9].,1,NoOp(You called: ${EXTEN})
exten => _[+*0-9].,n,Hangup(1)

exten => e,1,Hangup()

[moh] ; "fonction" pour la musique (cf. 5000). Rem : "fonction" est abusif, cela s'appelle
; en réalité le contexte.
```

```
exten => s,1,NoOp(Music On Hold)
exten => s,n,Ringing()
exten => s,n,Wait(2)
exten => s,n,Answer()
exten => s,n,Wait(1)
exten => s,n,MusicOnHold()

[dial-extension] ; "fonction" pour appeler un point de terminaison.
exten => s,1,NoOp(Calling: ${ARG1})
exten => s,n,Set(JITTERBUFFER(adaptive)=default)
exten => s,n,Dial(PJSIP/${ARG1},30)
exten => s,n,Hangup()

exten => e,1,Hangup()

[send-text] ; "fonction" pour envoyer du texte.
exten => s,1,NoOp(Sending Text To: ${ARG1} From: ${MESSAGE(from)})
exten => s,n,Set(PEER=${CUT(CUT(CUT(MESSAGE(from),@,1),<,2),:,)})
exten => s,n,Set(FROM=${SHELL(asterisk -rx 'pjsip show endpoint ${PEER}' | grep 'callerid '
| cut -d ':' -f2- | sed 's/^ *//' | tr -d '\n'))
exten => s,n,Set(CALLERID_NUM=${CUT(CUT(FROM,<,1),<2)})
exten => s,n,Set(FROM_SIP=${STRREPLACE(MESSAGE(from),<sip:${PEER}@,<sip:${CALLERID_NUM}@)})
exten => s,n,MessageSend(pjsip:${ARG1},${FROM_SIP})
exten => s,n,Hangup()
```

Pour davantage de détails, se référer à la documentation sur la configuration du plan de numérotation :

<https://wiki.asterisk.org/wiki/display/AST/Contexts%2C+Extensions%2C+and+Priorities>.

5. Redémarrer le service *asterisk*.

```
asterisktz@asterisktz:~$
```

Redémarrage du service *asterisk*

```
sudo systemctl restart asterisk
```

Tests de communication avec le client Web *Browser Phone*

Après avoir redémarré le service *asterisk*. Nous allons nous rendre sur le Raspberry Pi et installer Mozilla Firefox via la commande `sudo apt install firefox-esr`.

Ouvrons donc maintenant Mozilla Firefox et entrons dans la barre d'adresse l'URL suivante : <https://www.innovateasterisk.com/phone/>.

Il faut configurer les champs de la manière suivante puis cliquer sur **Save** :

Asterisk Server Address:
192.168.1.80

WebSocket Port:
8089

WebSocket Path:
/ws

Subscribe Extension (Internal):
5002

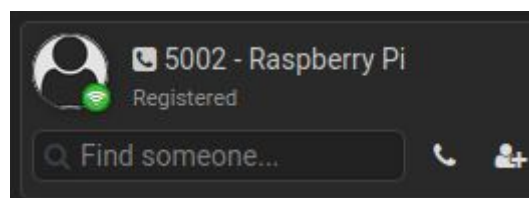
Full Name:
Raspberry Pi

SIP Username:
rpi

SIP Password:
.....

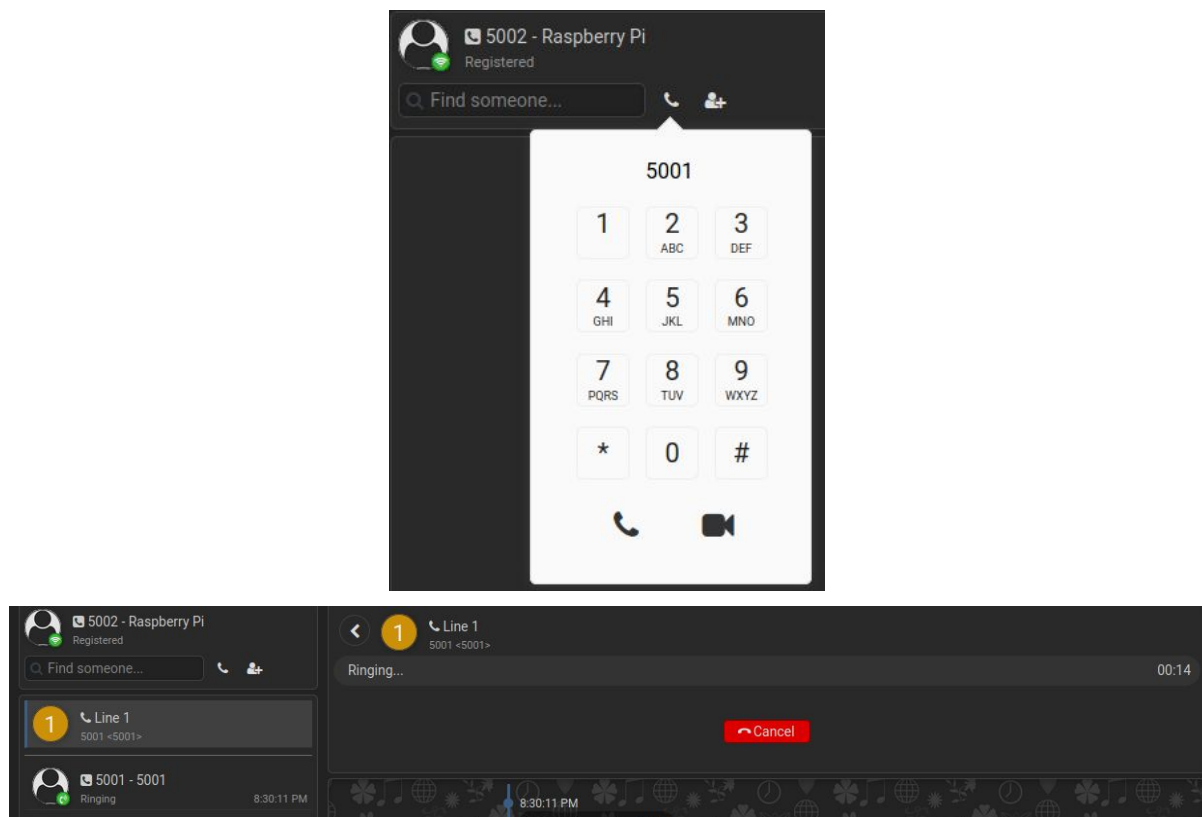
(Figure 19 - Configuration du client WebRTC Browser Phone)

L'indication *Registered* indique que le client est bien connecté au serveur Asterisk.



(Figure 20 - Enregistrement du client *rpi* sur le serveur Asterisk)

Nous pouvons donc appeler l'Alcatel IP Touch.



(Figure 21 - Appel Raspberry Pi vers Alcatel IP Touch 4018 EE)

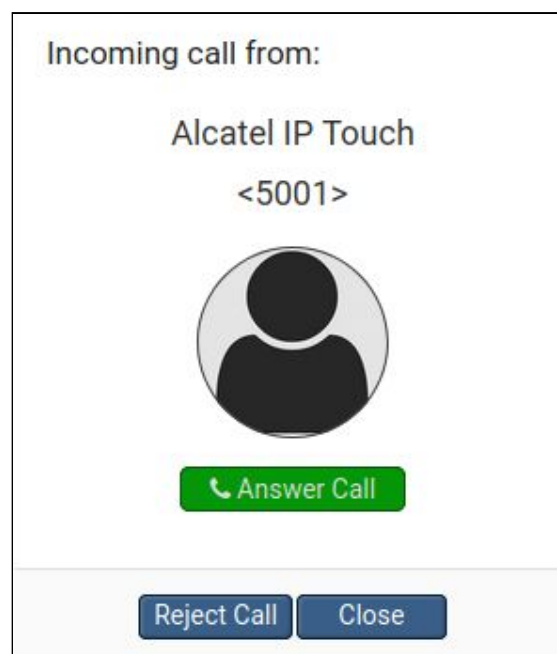


(Figure 22 - Réception de l'appel du Raspberry Pi)

Ou bien depuis l'Alcatel appeler le Raspberry Pi.



(Figure 23 - Appel Alcatel IP Touch 4018 EE vers Raspberry Pi)



(Figure 24 - Réception de l'appel de l'Alcatel IP Touch)

Développement d'un client SIP JavaScript

Compte tenu du projet et des temps, nous n'avons eu le temps que de tester des solutions déjà existantes (*Browser Phone*).

Le développement d'un client SIP peut se faire via les librairies déjà existantes telles que : [SIP.js](#), [JsSip](#), [sipML5](#)...

Pour information *Browser Phone* utilise SIP.js.

Conclusion

Ce projet a permis l'élaboration d'une communication en VoIP entre un Raspberry Pi et un téléphone IP Alcatel IP Touch 4018 EE. Nous avons découvert de nouveaux protocoles tels que SIP ou RTP et l'API WebRTC permettant d'encapsuler SIP pour établir une communication entre plusieurs points de terminaisons, utilisant un navigateur web ou non. On se rend bien compte du potentiel de l'IP pour les communications, et de l'indispensabilité de l'utilisation de ce type de technologie dans les entreprises ou services publics.

Ce projet pourrait être amélioré par la mise en place de chiffrement notamment avec le protocole SIPS (SIP over SSL/TLS) au niveau de l'établissement de la communication entre deux points de terminaison (fait en partie lorsqu'il est encapsulé dans une WebSocket over TLS), mais cela pourrait être intégral, puisque l'Alcatel supporte très bien SIP over TLS. Enfin, le protocole RTP peut aussi être chiffré, il s'agit du protocole SRTP (*Secure Real-time Transport Protocol*) lorsqu'il y a un appel en cours.

Je remercie Monsieur Lounis de m'avoir proposé cette TZ, particulièrement enrichissante, j'ignorais une grande partie du fonctionnement de la téléphonie sur IP. Ce sujet m'a permis de manipuler à la fois l'aspect back end et front end du système et de mesurer la puissance de cette architecture lorsqu'elle est intégrée dans le réseau de l'Internet.

Table des illustrations (hors annexes)

Figure 1 - Schéma global de l'infrastructure.....	4
Figure 2 - Pile protocolaire d'internet.....	6
Figure 3 - Établissement et fin de la communication - crédits : 3cx.fr...	7
Figure 4 - Établissement d'une communication téléphonique entre le Raspberry Pi et le téléphone Alcatel IP Touch 4018 EE.....	7
Figure 5 - PABX.....	8
Figure 6 - Paramétrage indicatif téléphonique.....	11
Figure 7 - Sélection des modules de sons d'Asterisk.....	13
Figure 8 - Sélection des modules de sons extra d'Asterisk.....	14
Figure 9 - Configuration d'asterisk en temps qu'utilisateur par défaut du service Asterisk.....	16
Figure 10 - Configuration d'asterisk en temps qu'utilisateur par défaut du service Asterisk.....	17
Figure 11 - Lancement du service Asterisk avec erreurs.....	18
Figure 12 - Lancement du service Asterisk sans erreur.....	19
Figure 14 - Alcatel-Lucent, 3. SIP stand-alone mode In : IP Touch 4008/4018 Extended Edition - SIP Phone Installation Guide - 8AL90824AAAA ed02, p.4-5, Août 2010.....	29
Figure 15 - Appel de Raspberry Pi vers Alcatel IP Touch 4018 EE.....	40
Figure 16 - Appel de Alcatel IP Touch 4018 EE vers Raspberry Pi.....	41
Figure 17 - Établissement d'une connexion entre deux clients utilisant WebRTC.....	43
Figure 18 - Raspberry Pi appelle Alcatel IP Touch depuis un client WebRTC.....	44
Figure 19 - Configuration du client WebRTC BrowserPhone.....	54
Figure 20 - Enregistrement du client rpi sur le serveur Asterisk.....	54
Figure 21 - Appel Raspberry Pi vers Alcatel IP Touch 4018 EE.....	55
Figure 22 - Réception de l'appel du Raspberry Pi.....	55
Figure 23 - Appel Alcatel IP Touch 4018 EE vers Raspberry Pi.....	56
Figure 24 - Réception de l'appel de l'Alcatel IP Touch.....	56

Sigles (hors annexes)

Sigle	Description
802.11	Norme IEEE 802.11 (Wi-Fi).
802.3	Norme IEEE 802.3 (Ethernet).
ANSSI	Agence nationale de la sécurité des systèmes d'information.
ECDSA	Elliptic curve digital signature algorithm, algorithme asymétrique de signature numérique utilisant la cryptographie sur les courbes elliptiques.
FTP	File Transfer Protocol.
HDMI	High-Definition Multimedia Interface.
HTTP	Hypertext Transfer Protocol.
HTTPS	HyperText Transfer Protocol Secure.
IEEE	Institute of Electrical and Electronics Engineers.
IP	Internet Protocol, couche réseau du modèle TCP/IP.
IPBX	Internet Protocol Private Branch eXchange, c'est un PABX fonctionnant sur la pile internet (aussi appelé PABX-IP).
ISDN	Integrated Services Digital Network : RNIS.
LTS	Long Term Support, version d'un logiciel/système supportée à long terme.
MAC	Media Access Control, protocole de la couche liaison du modèle OSI.
NAT	Network Address Translation.
P-521	Algorithme de chiffrement utilisant des courbes elliptiques développé par la National Institute of Standards and Technology.
PABX	Private Automatic Branch eXchange (Autocommutateur téléphonique privé), permet de relier les points de terminaison téléphoniques.
PABX-IP	cf. IPBX.
PBX	Private Branch eXchange, cf. PABX.
PHY	Couche physique du modèle OSI.
PoE	Power over Ethernet, norme IEEE 802.3af, un périphérique PoE permet d'alimenter un périphérique via Ethernet tout en conservant la capacité à transférer des données.
RFC	Request for comments, document officiel spécifiant des technologies de l'Internet.
RNIS	Réseau Numérique à Intégration de Services, réseau téléphonique numérique avec des débits pouvant atteindre 2 Mbit/s (Wikipédia).
RPi	Raspberry Pi.
RSA	"Chiffrement RSA (nommé par les initiales de ses trois inventeurs : Ronald Rivest, Adi Shamir et Leonard Adleman) est un algorithme de cryptographie asymétrique" - Wikipédia .
RTC	Réseau Téléphonique Commuté, réseau téléphonique analogique.
RTP	Real-time Transport Protocol, protocole applicatif permettant entre autres le transfert de

	flux audio ou vidéo.
SIP	Session Initiation Protocol, protocole établissant la communication VoIP entre deux points de terminaison.
SIPS	Session Initiation Protocol over SSL/TLS.
SMTP	Simple Mail Transfer Protocol.
SRTP	Secure Real-time Transport Protocol.
SSH	Secure Shell, protocole de communication applicatif.
SSL	Secure Socket Layer, protocole de sécurisation des échanges.
TCP	Transmission Control Protocol, protocole de la couche transport du modèle OSI (mode connecté).
TFTP	Trivial File Transfer Protocol, protocole applicatif permettant le transfert de fichiers par UDP.
TLS	Transport Layer Security, successeur de SSL.
UDP	User Datagram Protocol, protocole de la couche transport du modèle OSI (mode non connecté).
VM	Virtual Machine (machine virtuelle).
VoIP	Voice over Internet Protocol.
WebRTC	Web Real-Time Communication. "Interface de programmation (API) JavaScript développée au sein du W3C et de l'IETF." - Wikipédia
WS	"L'API WebSocket est une technologie évoluée qui permet d'ouvrir un canal de communication bidirectionnelle entre un navigateur (côté client) et un serveur."
WSS	Web Socket Secure.

Références

- (1) Wikipédia, *Généralités In : Autocommutateur téléphonique privé*, 26 novembre 2020, Disponible sur : https://fr.wikipedia.org/wiki/Autocommutateur_t%C3%A9l%C3%A9phonique_priv%C3%A9.
- (2) Secteur de la normalisation des télécommunications de l'UIT, *Spécification de la couche 3 de l'interface utilisateur-réseau RNIS pour la commande de l'appel de base*, UIT, Mai 1998, Disponible sur : <https://www.itu.int/rec/T-REC-Q.931-199805-I/fr>.
- (3) Secteur de la normalisation des télécommunications de l'UIT, *Interface entre équipement terminal de traitement de données et équipement de terminaison de circuit de données pour terminaux fonctionnant en mode paquet et raccordés par circuit spécialisé à des réseaux publics pour données*, UIT, Octobre 1996, Disponible sur : <https://www.itu.int/rec/T-REC-X.25-199610-I/fr>.
- (4) Matt Fredrickson, *PSA: chan_sip status changed to "deprecated" & Asterisk 17.0.0-rc2 Release*, Asterisk.org, 25 septembre 2019, Disponible sur : https://www.asterisk.org/deprecating-chan_sip-asterisk-17-0-0-rc2-release/
- (5) Odin Gremaud, *Introduction In : La traversée de NAT en VoIP SIP*, NEXCOM Systems, p.1, 20 juin 2012, Disponible sur : https://www.nexcom.fr/wp-content/uploads/whitepapers/bcp_nat_traversal.pdf.
- (6) Alcatel-Lucent, *Caractéristiques audio In : Téléphones Alcatel-Lucent IP Touch 4008/4018 Extended Edition*, p.2, 2010, Disponible sur : <https://www.bureautique-communication.fr/fileuploader/download/download/?d=0&file=custom%2Fupload%2F1%2F6%2F16590.pdf>.
- (7) Wikipédia, *G.711*, 17 juillet 2017, Disponible sur : <https://fr.wikipedia.org/wiki/G.711>.
- (8) Malcolm Davenport, *Answer, Playback, and Hangup Applications In : Asterisk Documentation*, 19 décembre 2013, Disponible sur : <https://wiki.asterisk.org/wiki/display/AST/Answer%2C+Playback%2C+and+Hangup+Applications>.
- (9) bluesman, *64 bit Raspberry Pi OS is here!*, Audiophile Style, 4 juin 2020, Disponible sur : <https://audiophilestyle.com/forums/topic/59499-64-bit-raspberry-pi-os-is-here/>.
- (10) Alcatel-Lucent, *3.3. Initializing an IP Touch 40x8 EE phone In : IP Touch 4008/4018 Extended Edition - SIP Phone Installation Guide - 8AL90824AAAA ed02*, p.7-8, Août 2010, Disponible sur : <https://www.cluster2.hostgator.co.in/files/writeable/uploads/hostgator136107/file/iptouchsipphoneinstallationguide-ed02.pdf>.
- (11) SECTIGO Store, *ECDSA vs RSA: Everything You Need to Know*, 9 juin 2020, Disponible sur : <https://sectigostore.com/blog/ecdsa-vs-rsa-everything-you-need-to-know/>.
- (12) Agence nationale de la sécurité des systèmes d'information, *Logarithme discret dans les courbes elliptiques définies sur GF(p) In : Référentiel Général de Sécurité version 2.0 - Annexe B1*, p.19-20, 21 février 2014, Disponible sur : https://www.ssi.gouv.fr/uploads/2014/11/RGS_v-2-0_B1.pdf.

(13) Fondation Raspberry Pi, *Static IP address* In : *TCP/IP networking*, raspberrypi.org, 18 novembre 2020, Disponible sur : <https://www.raspberrypi.org/documentation/configuration/tcpip/>.

Toutes les références ont été consultées le 12 février 2021.

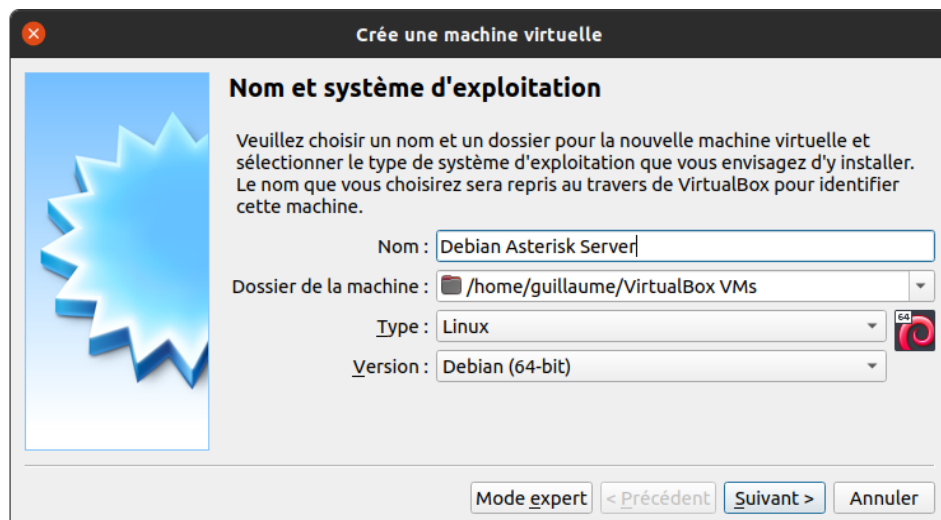
Annexes

Annexe A1 - Installation d'une machine virtuelle sous Debian 10 Buster

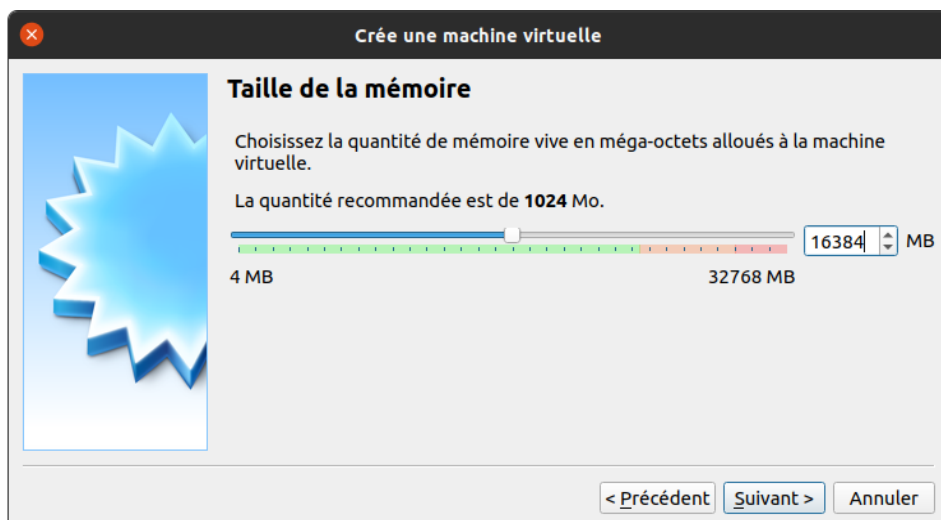
I - Préparation de la VM

Prérequis : avoir installé VirtualBox (<https://www.virtualbox.org/wiki/Downloads>) et disposer d'une connexion Internet.

1. Télécharger Debian 10 Buster **AMD64 netinst image** :
<https://cdimage.debian.org/debian-cd/current/amd64/iso-cd/>
2. Créer une VM : **Lancer VirtualBox** > **Nouvelle** puis cliquer sur **Suivant**.



3. Allouer au minimum **8 Gio** de mémoire vive puis cliquer sur **Suivant**.



4. Sélectionner **Créer un disque dur virtuel maintenant** puis cliquer sur **Créer**.
5. Choisir le type **VDI (VirtualBox Disk Image)** puis cliquer sur **Suivant**.
6. Sélectionner **Dynamiquement alloué** puis cliquer sur **Suivant**.
7. Sélectionner une taille minimum de **25 Gio** pour le projet puis cliquer sur **Créer**.

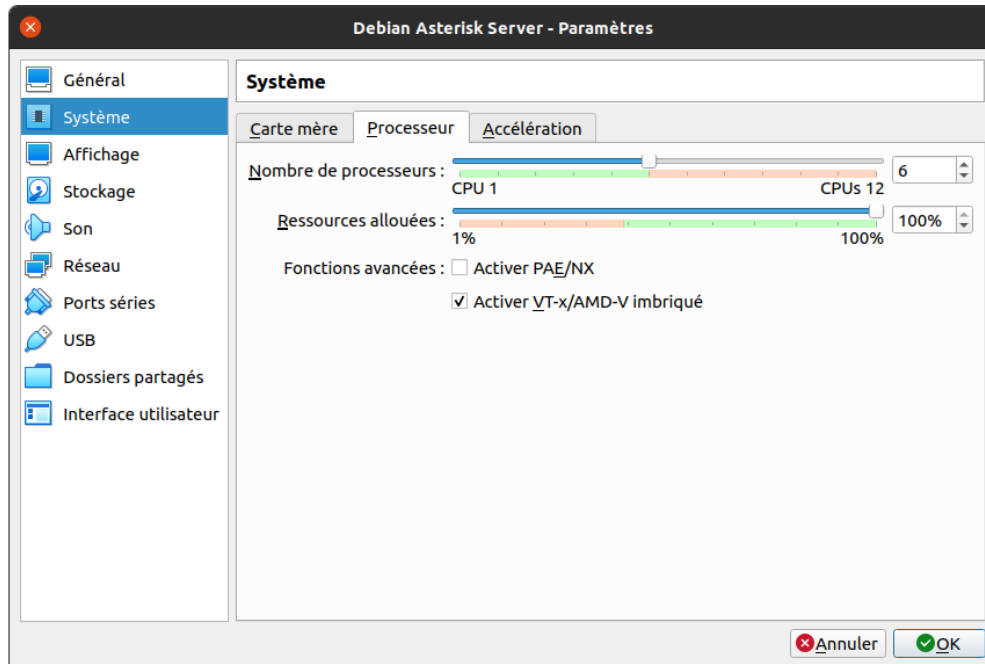


8. (Optionnel) Afin d'améliorer les performances de la VM il est intéressant d'exploiter la virtualisation. Pour ce faire, il faut préalablement activer Intel VT-x ou AMD-V dans le BIOS/UEFI (1).

Pour les utilisateurs Ubuntu seulement, il faudra effectuer une manipulation supplémentaire en ouvrant un terminal et en tapant la commande suivante :

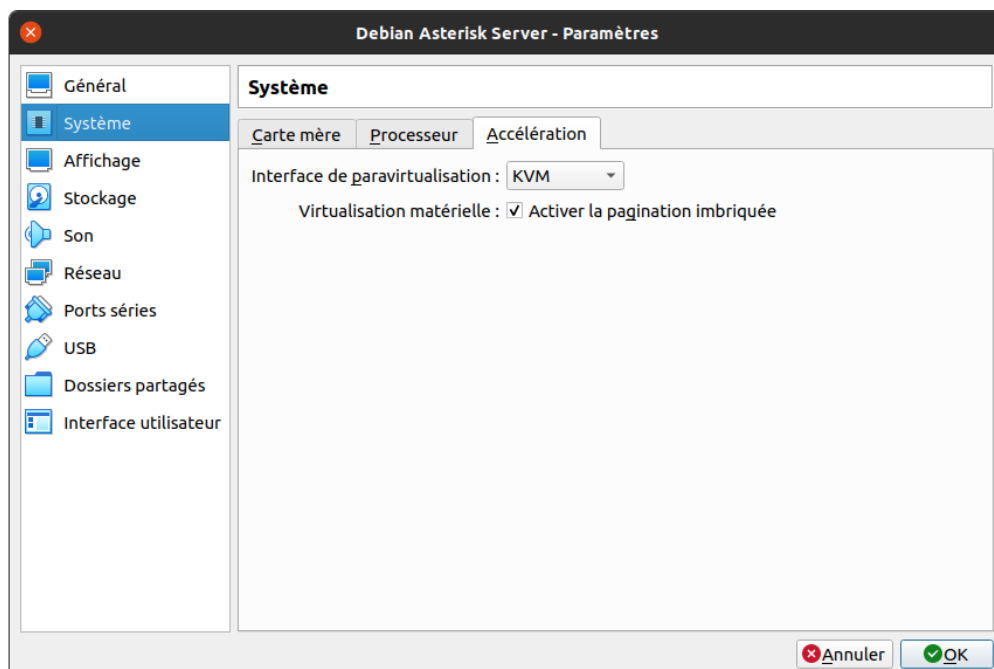
```
VBoxManage modifyvm "Debian Asterisk Server" --nested-hw-virt on
```

9. Lancer VirtualBox, sélectionner la machine **Debian Asterisk Server**, cliquer sur **Configuration** > **Système** > onglet **Processeur**. Mettre la moitié de CPU disponibles sur votre PC et cocher **Activer VT-x/AMD-V imbriqué**.

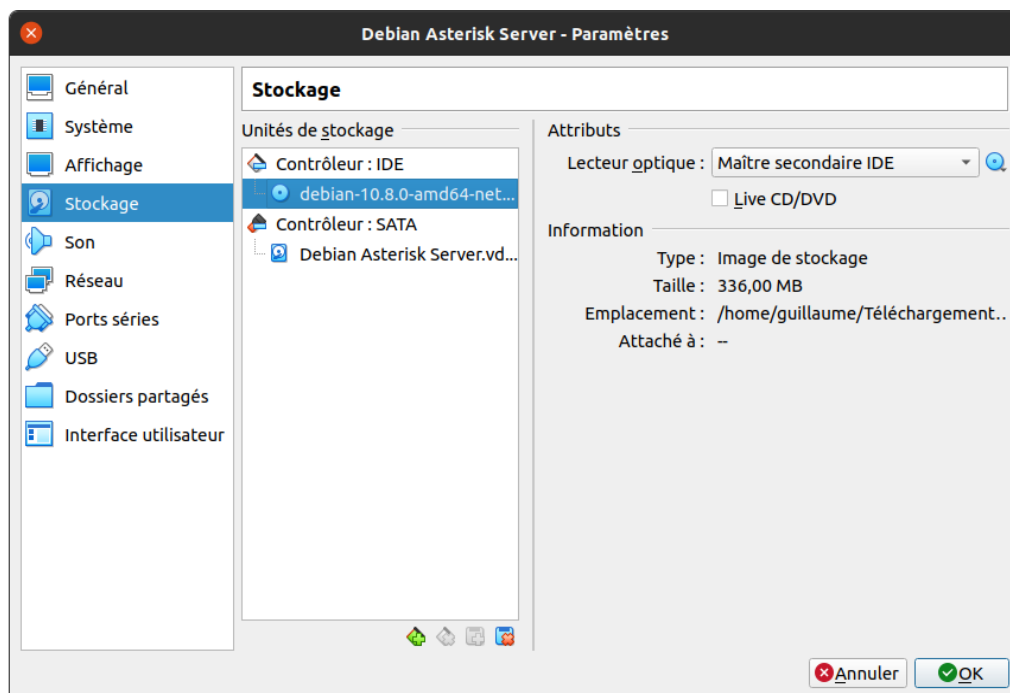


10. Dans l'onglet Accélération, sélectionner l'interface de paravirtualisation (2) :

- **KVM** si la machine hôte est un système Linux ;
- **Hyper-V** si la machine hôte est un système Windows ;
- **Minimal** si la machine hôte est un système macOS.

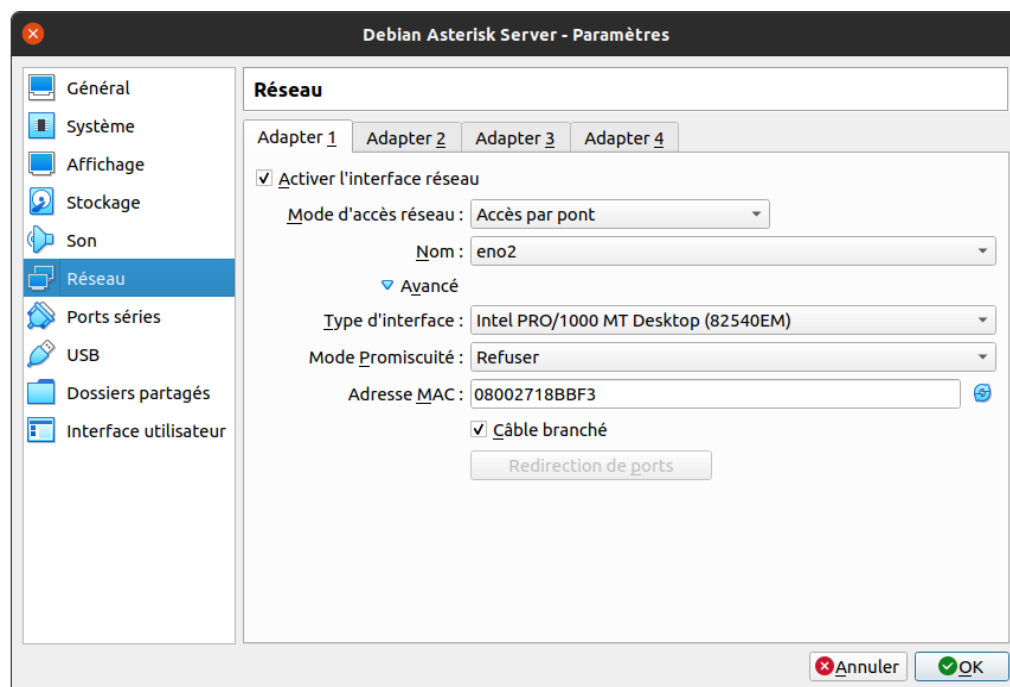


11. Dans **Stockage**, sélectionner l'image disque contenant Debian 10 Netinst.



12. Dans **Réseau**, onglet **Adapter 1**, sélectionner le mode d'accès réseau **Accès par pont**, puis cliquer sur **OK**.

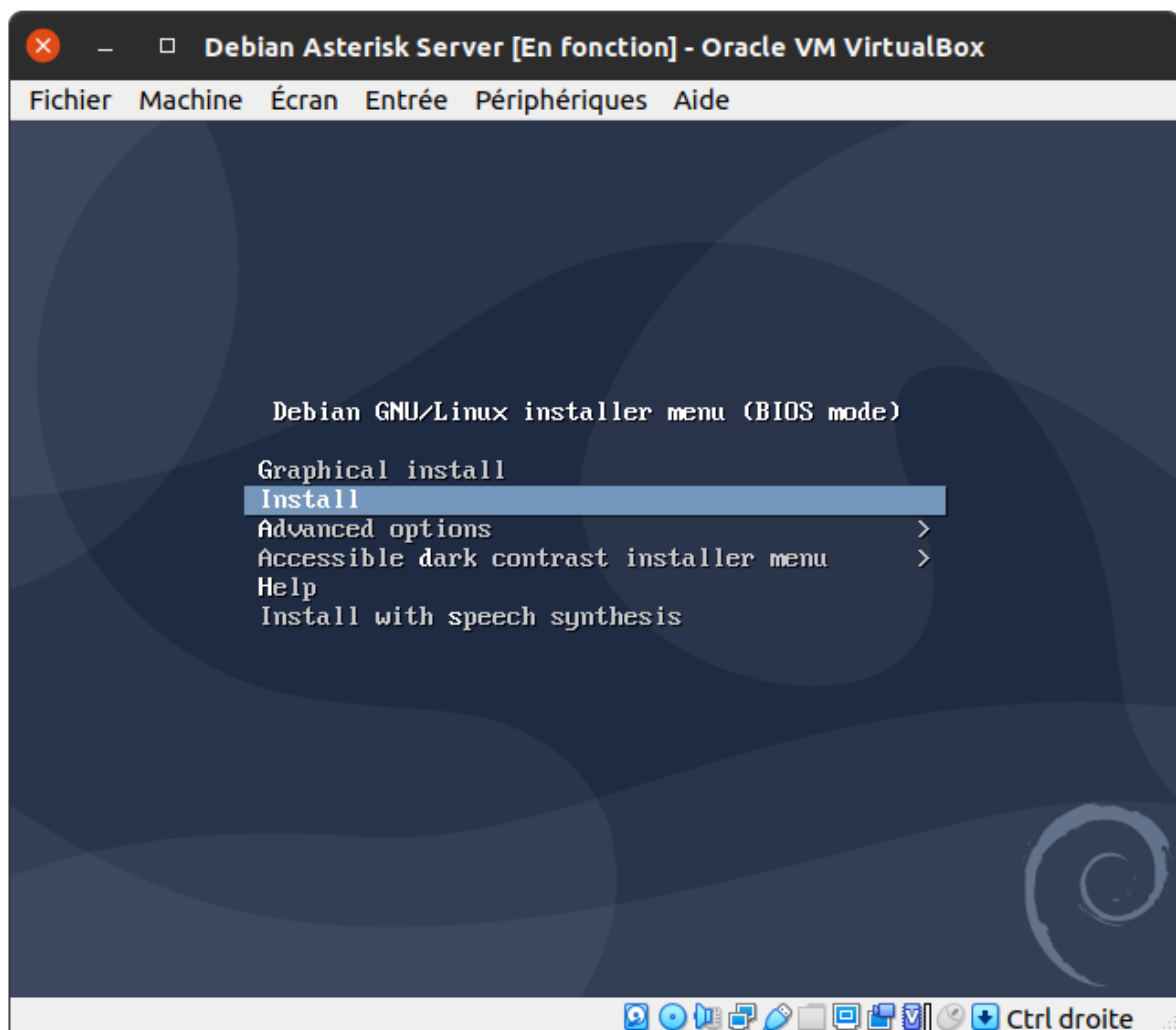
Remarque : C'est un choix arbitraire, nous avons fait ce choix pour qu'il soit plus facile d'accéder à la plateforme depuis le réseau local. Il est tout à fait possible d'utiliser le mode NAT et de faire des redirections de ports, cependant il faudra réaliser des opérations supplémentaires sur Asterisk.



La configuration de la VM est terminée, passons à l'installation de Debian !

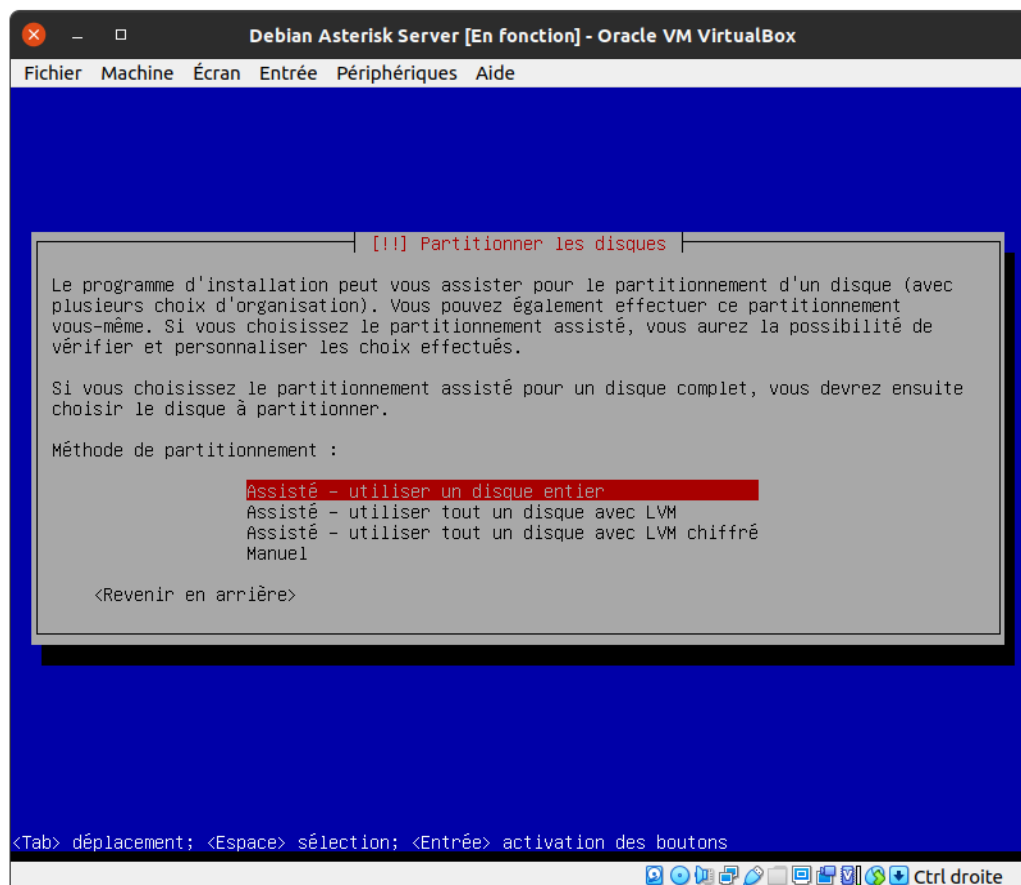
II - Installation et configuration de Debian 10

1. Lancer la VM en cliquant sur **Démarrer**, et choisir l'ISO de Debian Netinst comme disque de démarrage
2. Sélectionner **Install**.

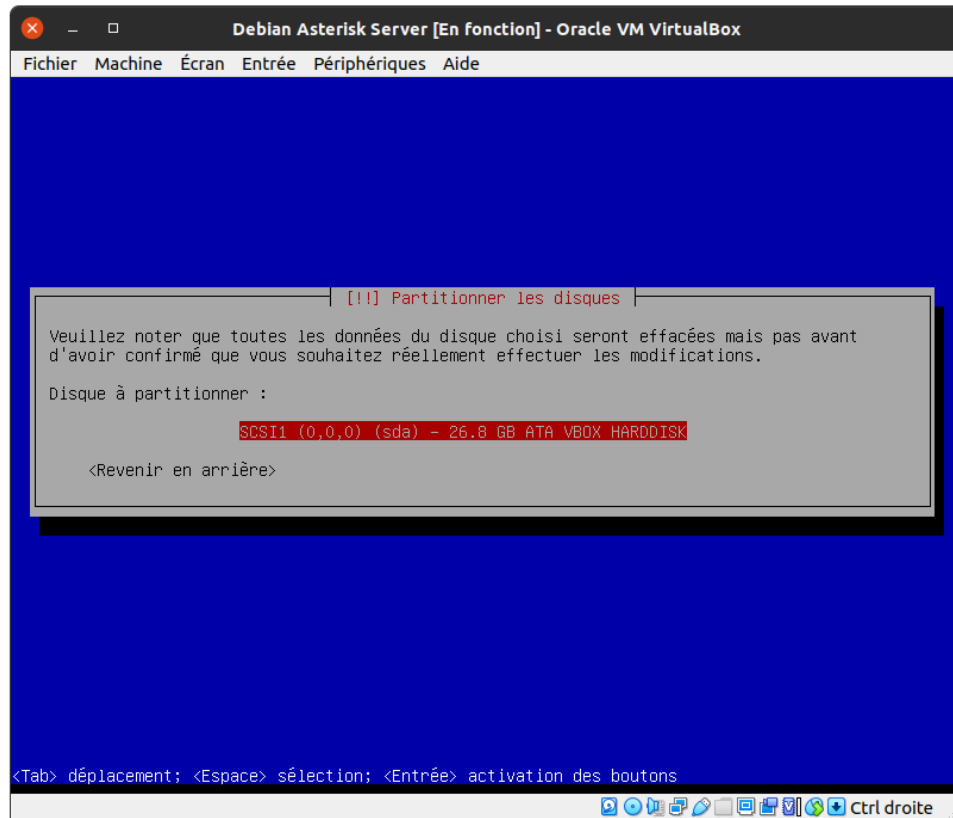


3. **[!!!] Select a language**
Sélectionner une langue, dans ce tutoriel ce sera le français puis taper sur **Enter**.
4. **[!!!] Choix de votre situation géographique**
Sélectionner un pays, dans ce tutoriel ce sera **France**.
5. **[!!!] Configurer le clavier**
Sélectionner la disposition de clavier correspondante à votre clavier.

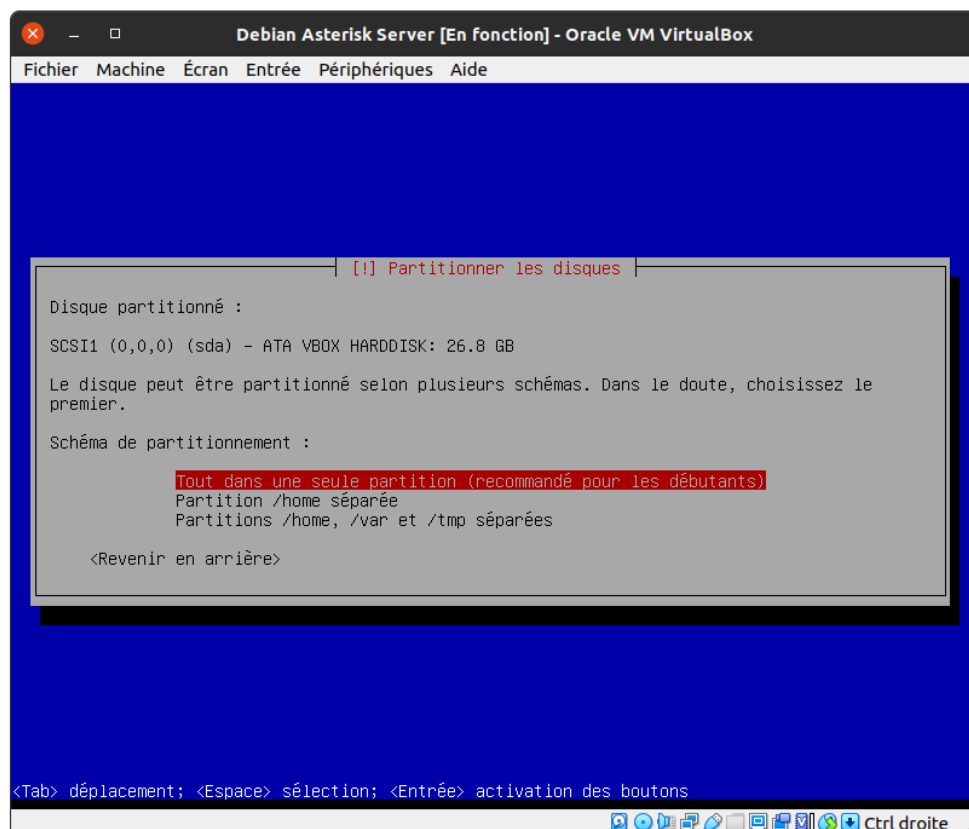
6. **[[!]] Configurer le réseau - nom de machine**
Choisir un nom pour la machine, ici ce sera : **asterisktz**, puis sélectionner **<Continuer>**.
7. **[[!]] Configurer le réseau - domaine**
Laisser vide, puis sélectionner **<Continuer>**.
8. **[[!]] Créer les utilisateurs et choisir les mots de passe - root**
Choisir un mot de passe pour le superutilisateur (root), ici ce sera **voiputc**.
Confirmer le mot de passe puis **<Continuer>**.
9. **[[!]] Créer les utilisateurs et choisir les mots de passe - utilisateur non root**
Choisir un nom et un identifiant pour le nouvel utilisateur non root, ici ce sera **asterisktz** ainsi qu'un mot de passe, dans notre cas ce sera également **voiputc**.
Bien évidemment, dans un environnement de production le mot de passe root et le mot de passe de l'utilisateur classique ne doivent pas être les mêmes. Ici le but est pédagogique.
10. **[[!]] Partitionner les disques**
Choisir **Assisté - utiliser un disque entier**.



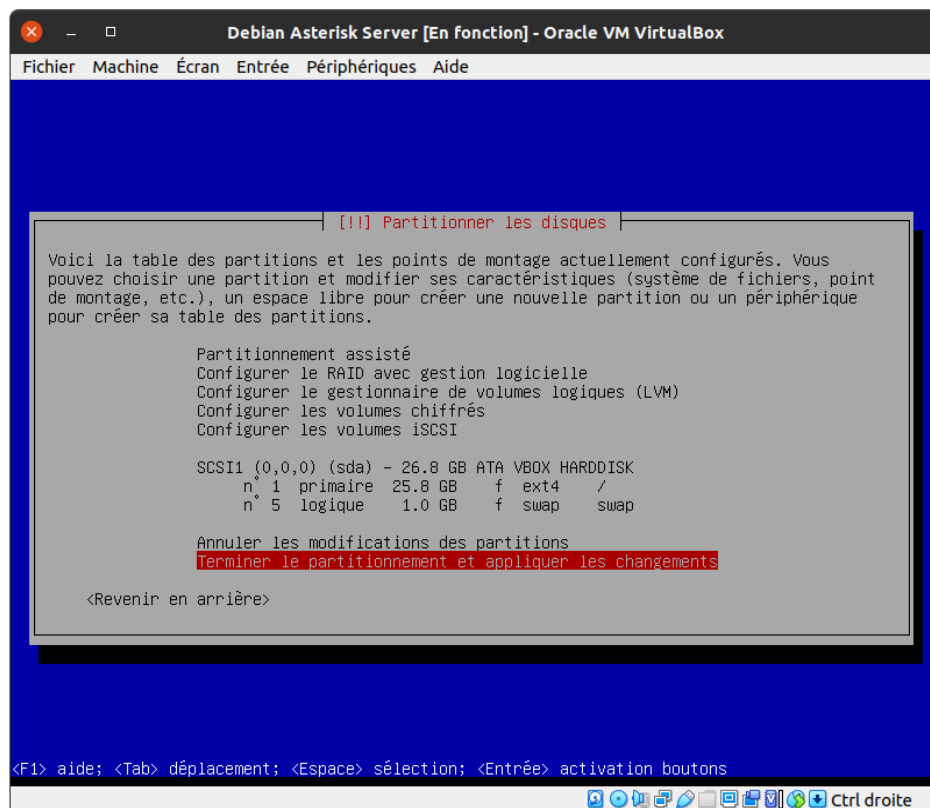
Sélectionner le seul disque à partitionner.



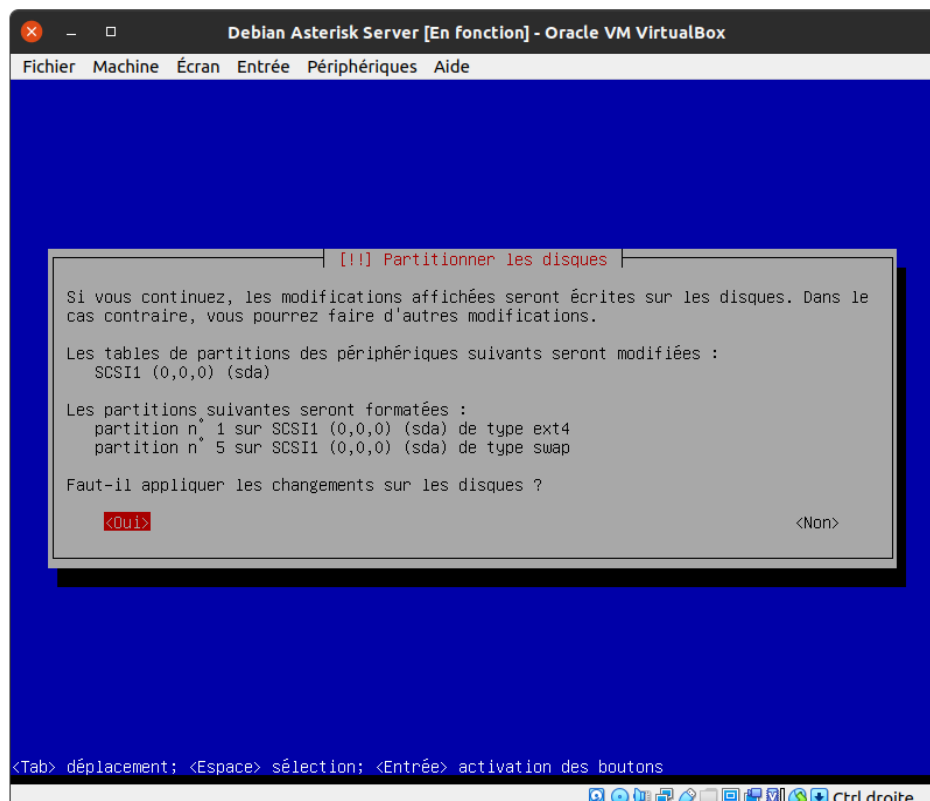
Choisir le schéma de partitionnement : **Tout dans une seule partition.**



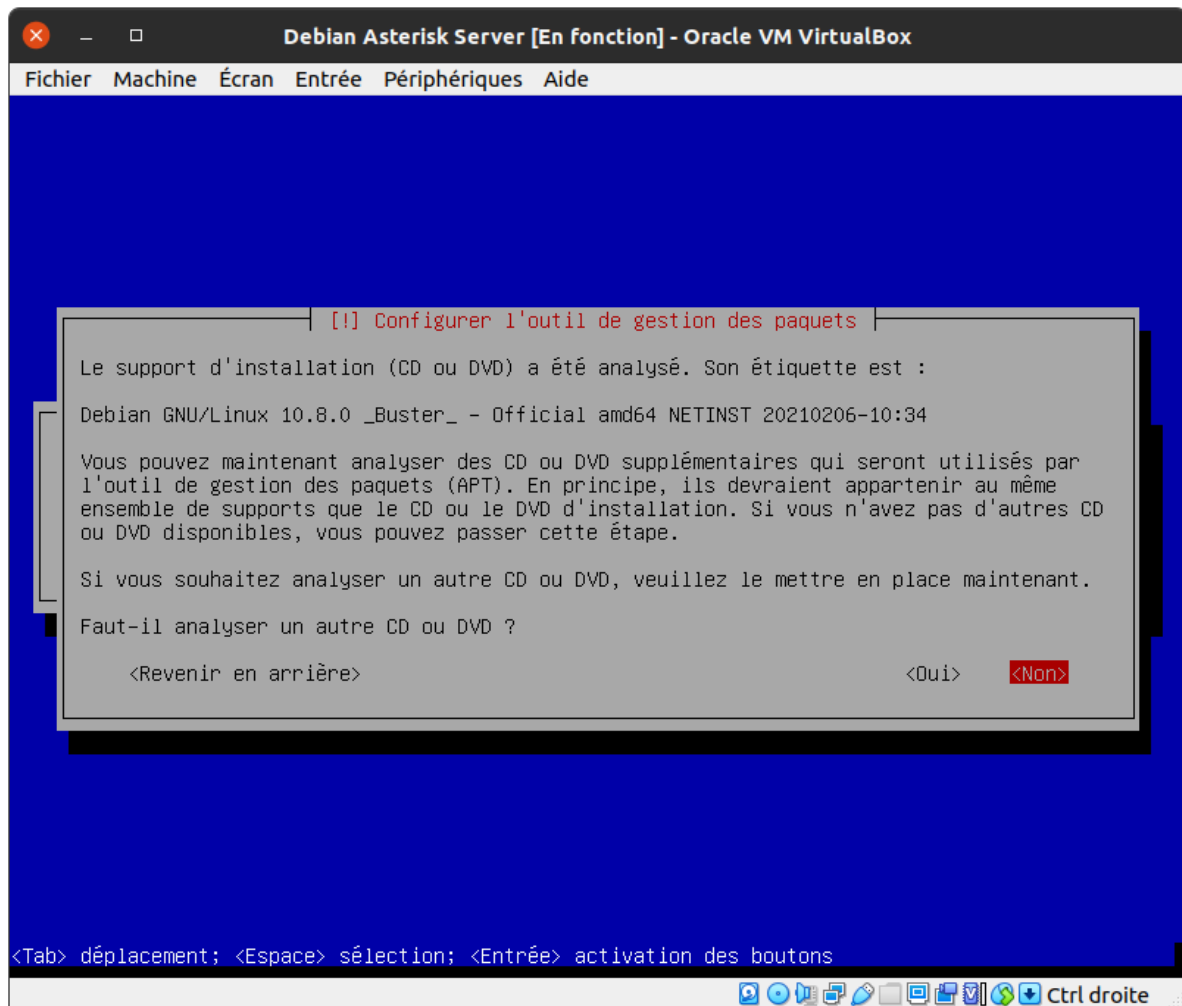
Puis sélectionner **Terminer le partitionnement et appliquer les changements**.



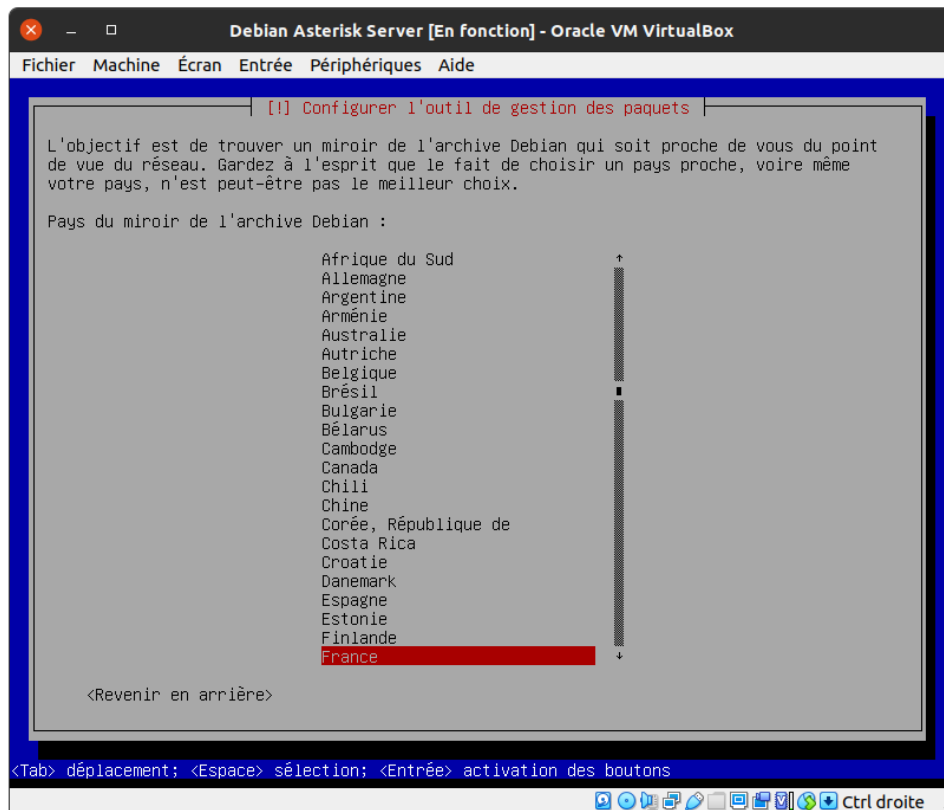
Confirmer les changements en sélectionnant **<Oui>**.



Le système s'installe. À la fin de l'installation, l'installateur propose d'analyser un autre CD ou DVD, il faut choisir **<Non>**.



11. **[!!] Configurer l'outil de gestion des paquets**
Ici ce sera **France**...



...puis le miroir **deb.debian.org**.

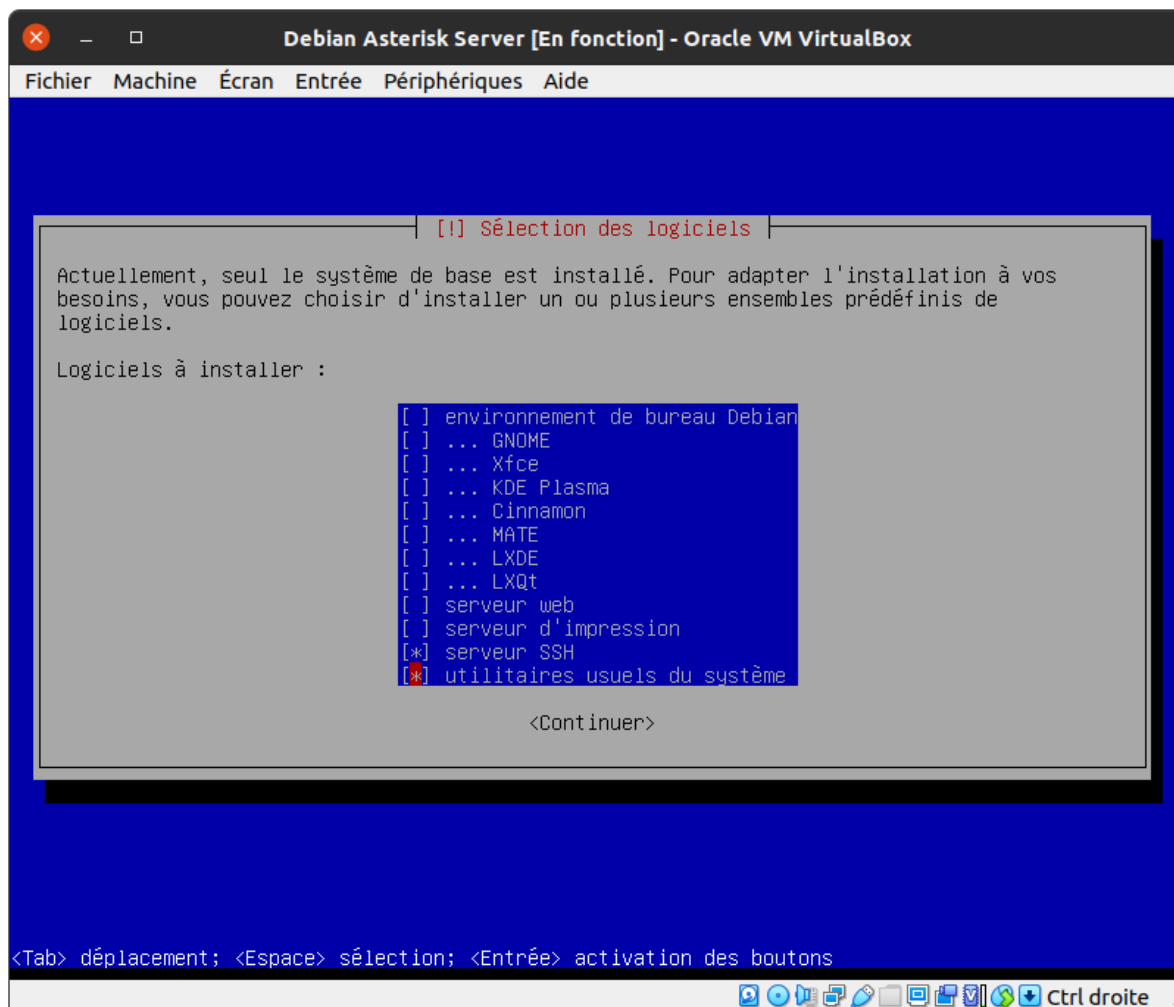


Après avoir choisi le miroir, une fenêtre s'affiche pour configurer le proxy HTTP, configurer cette partie si besoin est, puis sélectionner **<Continuer>**.

12. **[!!] Sélection des logiciels**

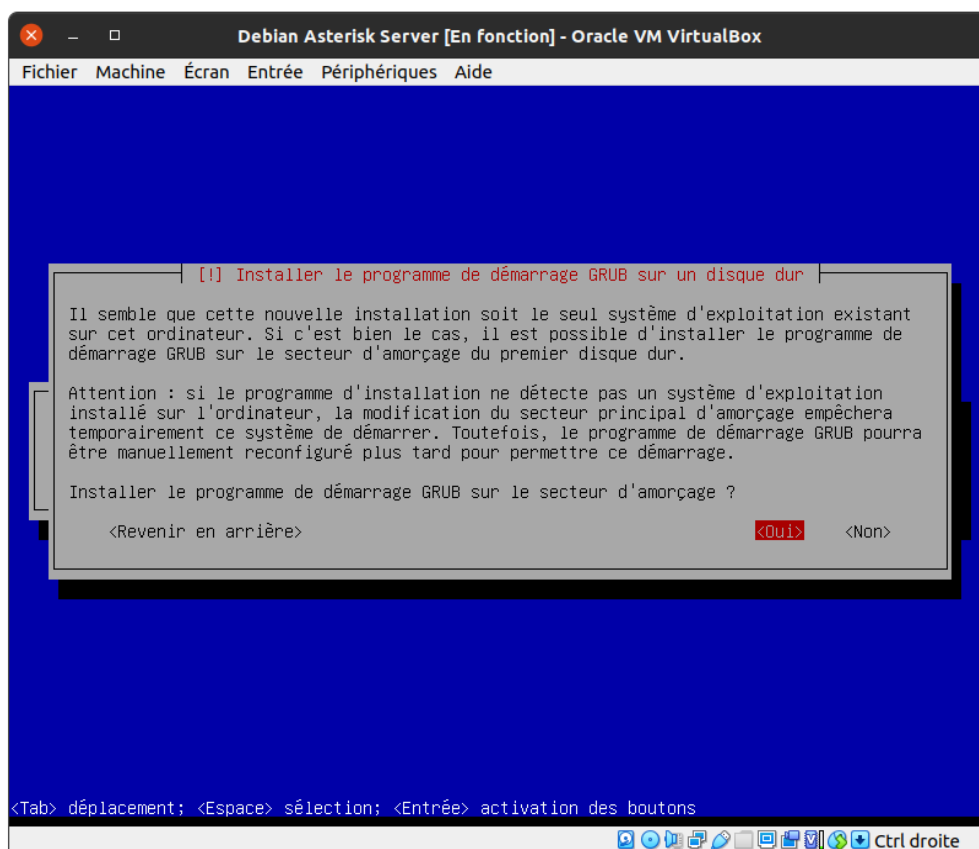
Sélectionner seulement le serveur SSH et les utilitaires usuels du système comme ci-dessous, puis cliquer sur **<Continuer>**.

Remarque : La sélection se fait avec la barre Espace.

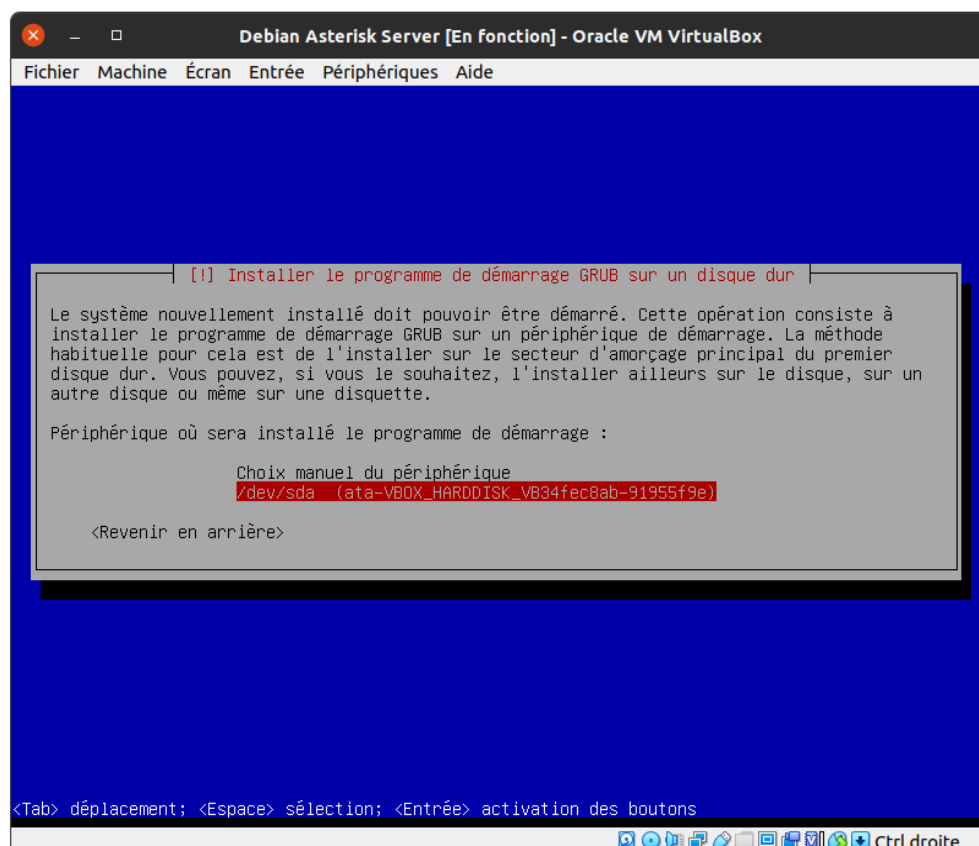


13. **[!!] Installer le programme de démarrage GRUB sur un disque dur**

Sélectionner **<Oui>**.



Puis sélectionner le disque dur `/dev/sda`.



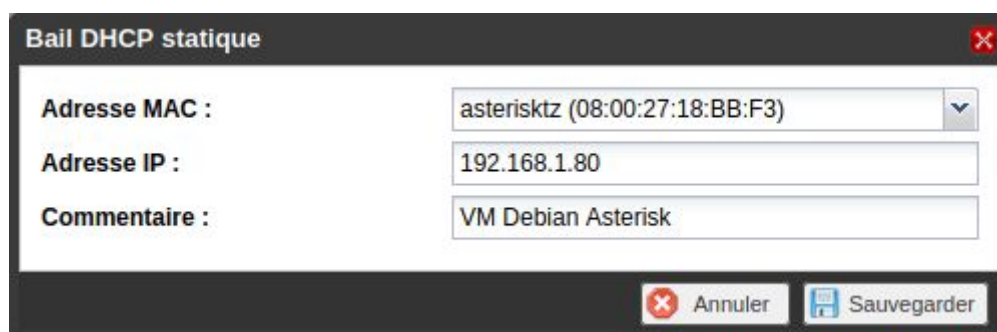
14. **[!!] Terminer l'installation**

Retirer le support d'installation (VirtualBox le fait automatiquement en général) puis sélectionner **<Continuer>**.



15. Après le redémarrage, configurer une adresse IP statique depuis le routeur (ici, il s'agit d'une Freebox). C'est possible car nous avons configuré un accès par pont depuis l'interface de VirtualBox. Plus d'informations sur la procédure d'attribution de bail DHCP statique sur les routeurs Freebox :

<https://wxfrantzconcept.wordpress.com/2016/10/18/assigner-une-adresse-ip-fixe-avec-sa-freebox/>.



16. Redémarrer la machine virtuelle puis se connecter en SSH.

```
# ssh login@adresse_ip_vm -p 22  
ssh asterisktz@192.168.1.80 -p 22
```

17. Installer les mises à jour et sudo.

```
asterisktz@asterisktz:~$
```

Se connecter en root

```
su
```

```
root@asterisktz:/home/asterisktz#
```

Installer les mises à jour et sudo

```
apt update && apt upgrade -y && apt dist-upgrade -y && apt autoremove -y  
&& apt install sudo -y
```

```
root@asterisktz:/home/asterisktz#
```

Configurer sudo

```
sudo visudo -f /etc/sudoers.d/asterisktz
```

```
/etc/sudoers.d/asterisktz
```

Ajouter l'utilisateur **asteriskz** pour qu'il ait les droits
de superutilisateur puis enregistrer via **Ctrl + O**

```
asterisktz    ALL=(ALL:ALL) ALL
```

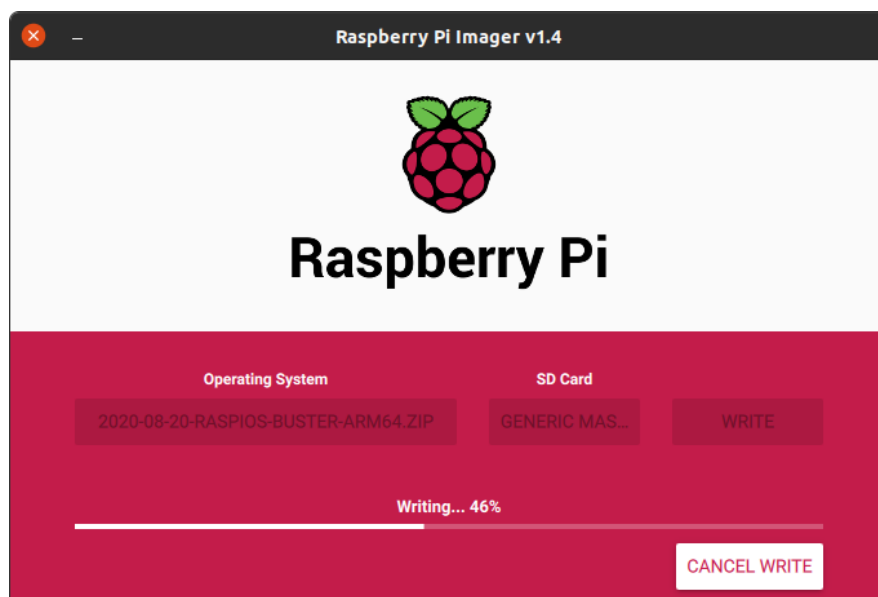
18. Redémarrer la machine virtuelle puis se connecter en SSH.

```
ssh asterisktz@192.168.1.80 -p 22
```

La machine Debian est prête, vous pouvez revenir à l'[installation du serveur Asterisk](#).

Annexe A2 - Installation et configuration de Raspberry Pi OS Buster (64 bits) pour Raspberry Pi 3B+

1. Télécharger et flasher²¹ l'image `2020-08-20-raspbios-buster-arm64.zip` disponible ici :
https://downloads.raspberrypi.org/raspbios_arm64/images/raspbios_arm64-2020-08-24/.



À la fin du processus, le Raspberry Pi est fonctionnel, mais il est intéressant d'activer le serveur SSH et de configurer une IP statique sur les interfaces Ethernet et/ou Wi-Fi.

2. Pour mettre en place le SSH, il suffit de créer un fichier nommé `ssh` ne contenant rien à la racine du disque `boot` de la carte microSD.
Remarque : cette étape est facultative
3. Pour configurer une IP statique sur les deux interfaces, il suffit d'aller dans le disque rootfs et d'ajouter les lignes suivantes²² à la fin du fichier `/etc/dhcpd.conf` (13) :
Remarque : cette étape est facultative ; vous pouvez aussi configurer une IP statique via le routeur (cf. Annexe A1.15).

²¹ Il est recommandé d'utiliser Raspberry Pi Imager pour le flashage de la carte microSD :
<https://www.raspberrypi.org/software/>.

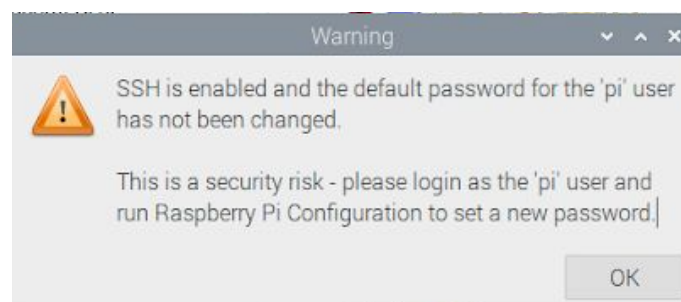
²² Nous avons choisi arbitrairement ces adresses IP statiques car elles sont en dehors de la plage d'attribution du serveur DHCP du routeur sur lequel est connecté le Raspberry Pi. Il faudra les droits root pour effectuer cette modification. Le premier serveur DNS est celui de la Freebox et le deuxième de DNS.WATCH : <https://dns.watch/>.

/rootfs/etc/dhcpd.conf

```
# IP statique interface Ethernet
interface eth0
static ip_address=192.168.1.82/24
static routers=192.168.1.254
static domain_name_servers=192.168.1.254 84.200.69.80

# IP statique interface Wi-Fi
interface wlan0
static ip_address=192.168.1.92/24
static routers=192.168.1.254
static domain_name_servers=192.168.1.254 84.200.69.80
```

4. Brancher la carte microSD dans le Raspberry Pi, brancher un clavier, une souris, un écran et le démarrer.
5. Cliquer sur **OK** sur le message lié à SSH. En pratique, il faudrait changer le login et le mot de passe. Ici, ce n'est pas le but du projet.



6. Configurer le Raspberry Pi, cliquer sur **Next**.



7. Configurer les langues.



The screenshot shows the 'Set Country' window of the Raspberry Pi configuration tool. The title bar says 'Welcome to Raspberry Pi'. The window title is 'Set Country'. The instructions say: 'Enter the details of your location. This is used to set the language, time zone, keyboard and other international settings.' There are three dropdown menus: 'Country' set to 'France', 'Language' set to 'French', and 'Timezone' set to 'Paris'. Below these are two checkboxes: 'Use English language' and 'Use US keyboard', both of which are unchecked. At the bottom, it says 'Press 'Next' when you have made your selection.' and there are 'Back' and 'Next' buttons.

8. Changer le mot de passe par défaut. Ici ce sera **voippiutc**.



The screenshot shows the 'Change Password' window of the Raspberry Pi configuration tool. The title bar says 'Welcome to Raspberry Pi'. The window title is 'Change Password'. The instructions say: 'The default 'pi' user account currently has the password 'raspberrry'. It is strongly recommended that you change this to a different password that only you know.' There are two text input fields: 'Enter new password:' and 'Confirm new password:', both containing the text 'voippiutc'. To the right of the second field is a checkbox labeled 'Hide characters' which is unchecked. At the bottom, it says 'Press 'Next' to activate your new password.' and there are 'Back' and 'Next' buttons.

9. Paramétrer l'écran.



The screenshot shows the 'Set Up Screen' window of the Raspberry Pi configuration tool. The title bar says 'Welcome to Raspberry Pi'. The window title is 'Set Up Screen'. The instructions say: 'The desktop should fill the entire screen. Tick the box below if your screen has a black border at the edges.' There is a checkbox labeled 'This screen shows a black border around the desktop' which is checked. Below this, it says 'Press 'Next' to save your setting.' and 'The change will take effect when the Raspberry Pi is restarted.' At the bottom, there are 'Back' and 'Next' buttons.

10. Sélectionner le réseau Wi-Fi ou non si la connexion est en Ethernet.
11. Mettre à jour le système et les logiciels en cliquant sur **Next**. L'opération peut prendre du temps.



12. Redémarrer le Raspberry Pi.



Le système est opérationnel, vous pouvez vous rendre à l'[installation du client SIP Linphone](#).