*Serveur TOIP accessible depuis l'extérieur et chiffrement*

# Introduction

Nous avons déjà mis en place précédemment un serveur TOIP mais n'avons jamsi abordé la notion d'access derrière une box FAI ou celle duchiffrement hors TLS via RTSP (chiffrement syméetrique ARS)

Dans le cadre de la mise en place du serveur Asterisk, une problématique est apparue liée à l'évolution des versions du logiciel. Les versions récentes (notamment Asterisk 21 et supérieures) ont supprimé le module historique **chan_sip** au profit de **PJSIP**, ce qui rend les anciennes commandes et méthodes de configuration (comme l'utilisation de sip.conf et la commande sip show users) inopérantes. Cette incompatibilité a nécessité un retour vers une version antérieure d'Asterisk (18 LTS) et une compilation manuelle avec activation explicite du module chan_sip via menuselect. Une fois ce module correctement compilé et chargé, il a été possible de retrouver un fonctionnement conforme aux anciennes pratiques, permettant une configuration simplifiée des utilisateurs SIP et des tests via un client softphone.

Au niveau des ports  et protocole

# La stack protocolaire

Téléphone SIP → RTP (voix) + SIP (signalisation) → Asterisk → Trunk SIP → PSTN

- **SIP** (port 5060 UDP/TCP) : signalisation (INVITE, ACK, BYE...)
- **RTP** (ports 10000-20000 UDP) : flux audio en temps réel
- **SRTP / TLS** : versions chiffrées des deux

# Le problème de la box FAI (NAT)

C'est le cœur du défi. La box fait du **NAT** (Network Address Translation), donc :

Internet (IP publique) ↔ Box (NAT) ↔ Asterisk (IP privée 192.168.x.x)

## Problèmes typiques :

- Le client SIP externe envoie ses paquets vers ton IP publique, mais la box ne sait pas à qui les transférer
- Asterisk annonce son IP privée dans les headers SIP → le correspondant externe ne peut pas joindre cette IP

## Solutions pour traverser le NAT

### *A) Port Forwarding sur la box*

Rediriger les ports entrants vers Asterisk :

| Port | Protocole | Usage |
|---|---|---|
| 5060 | UDP | SIP |
| 10000-20000 | UDP | RTP (media) |
| 5061 | TCP/TLS | SIP sécurisé |

### *B) Configurer Asterisk côté NAT*

Dans `sip.conf`:

ini

```
[general]
externip=<ton_IP_publique>   ; IP publique de la box
localnet=192.168.0.0/24      ; ton réseau local
nat=yes                      ; active la correction NAT
```

Asterisk va alors substituer l'IP privée par l'IP publique dans les headers SIP.

### *C) IP publique dynamique → DDNS*

Les FAI changent souvent l'IP publique. Solution : un service **DDNS** (No-IP, DynDNS…) avec un client sur la box ou un script :

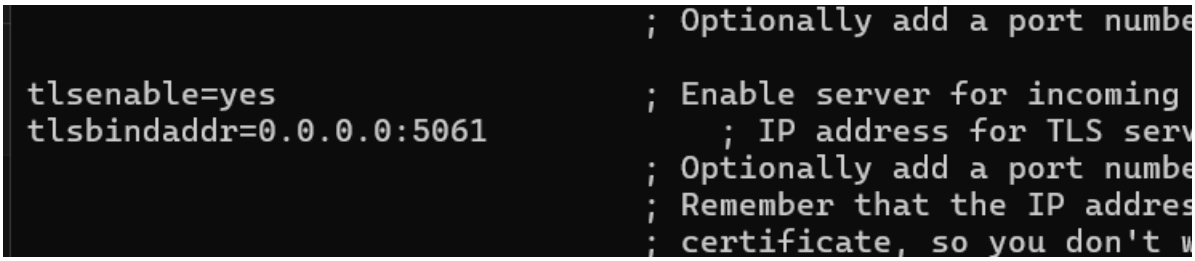
monasterisk.ddns.net → mis à jour automatiquement → pointe vers ton IP publique

---

# Ajout TLS

Je créer un clef privée et un certificat qui contiendra une clef publique

Il faut d'abord activer TLS et lui attrobuer un port




On créer la clef privé ensuite on genere un certif en se bsant sur cette dernière et ne pas oublier l'autorite de certification

openssl req -new -key asterisk.key -out asterisk.csr

openssl x509 -req -day 3655 -in asterisk.csr -signkey asterisk.key -out asterisk.pem

Dans sip.conf

```
tlscertfile=/ssl/asterisk.pem ; Certificate chain (*.pem format only) to use for TLS connections
                                ; The certificates must be sorted starting with the subject's certificate
                                ; and followed by intermediate CA certificates if applicable. If the
                                ; file name ends in _rsa, for example "asterisk_rsa.pem", the files
                                ; "asterisk_dsa.pem" and/or "asterisk_ecc.pem" are loaded
                                ; (certificate, intermediates, private key), to support multiple
                                ; algorithms for server authentication (RSA, DSA, ECDSA). If the chains
                                ; are different, at least OpenSSL 1.0.2 is required.
                                ; Default is to look for "asterisk.pem" in current directory

tlsprivatekey=/ssl/asterisk.pem ; Private key file (*.pem format only) for TLS connections.
                                ; If no tlsprivatekey is specified, tlscertfile is searched for
                                ; for both public and private key.

tlscafile=/ssl/asterisk.crt
;       If the server your connecting to uses a self signed certificate
;       you should have their certificate installed here so the code can
;       verify the authenticity of their certificate.

;tlscapath=</path/to/ca/dir>
;       A directory full of CA certificates.  The files must be named with
;       the CA subject name hash value.
;       (see man SSL_CTX_load_verify_locations for more info)

;tlsdontverifyserver=[yes|no]
;       If set to yes, don't verify the servers certificate when acting as
;       a client.  If you don't have the server's CA certificate you can
;       set this and it will connect without requiring tlscafile to be set.
;       Default is no.

tlscipher=ALL
;       A string specifying which SSL ciphers to use or not use
;       A list of valid SSL cipher strings can be found at:
;               http://www.openssl.org/docs/apps/ciphers.html#CIPHER_STRINGS
;
tlsclientmethod=tlsv1     ; values include tlsv1, sslv3, sslv2.
                          ; Specify protocol for outbound client connections.
                          ; If left unspecified, the default is the general-
                          ; purpose version-flexible SSL/TLS method (sslv23).
                          ; With that, the actual protocol version used will
                          ; be negotiated to the highest version mutually
                          ; supported by Asterisk and the remote server, i.e.
                          ; TLSv1.2. The supported protocols are listed at
```

Ensuite

chown -R asterisk:asterisk /ssl

chmod 755 /ssl

chmod 600 /ssl/asterisk.key /ssl/asterisk.pem

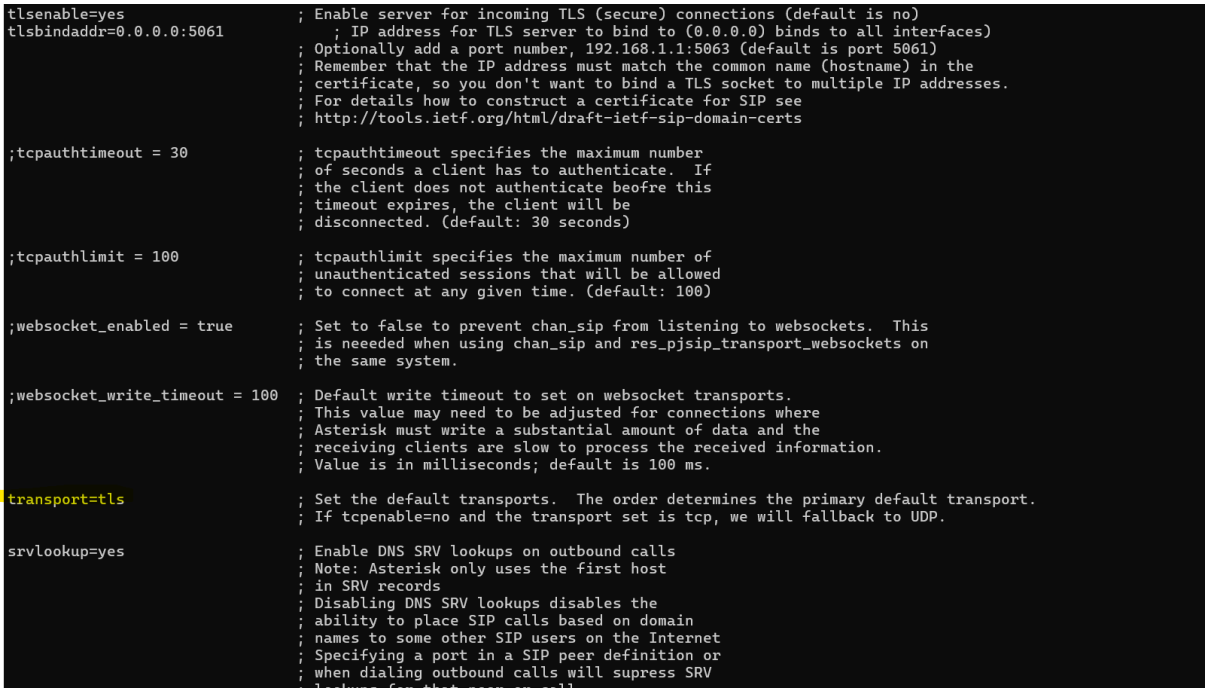
chmod 644 /ssl/asterisk.crt

C’est le même pem pour certificat et key dans le fichier de conf

Et dans console asterisk

Reload

Ensuite vérifier que le service ecoute bien sur le port  5061 et dans transport mettre : TLS puis relancer

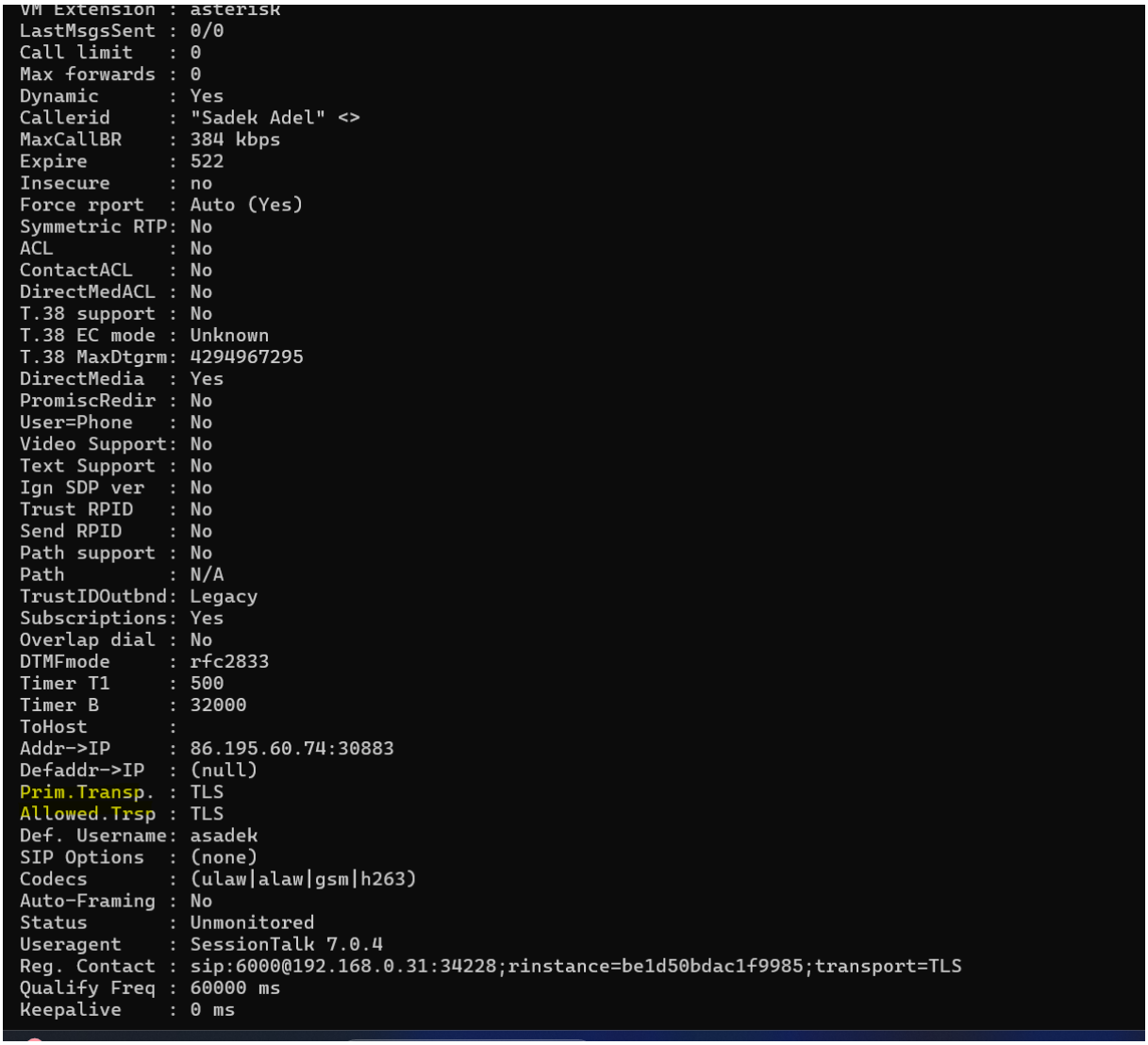



Je me suis connecté avec user 6000

Pour vérifier si il est bien en TLS j execute cette commande

sip show peer 6000

Je vois bien ici TLS




## Chiffrement des appels après signaux SIP

Le chiffrement des appels (la voix elle-même) ne passe pas par TLS mais par **SRTP (Secure RTP)**. Dans Asterisk avec chan_sip, on l'active côté utilisateurs avec encryption=yes (et côté client il faut aussi activer SRTP). SRTP chiffre le flux audio RTP avec des algorithmes standards : en pratique **AES** (souvent AES-128) en mode compteur (**AES-CM**) pour le chiffrement, et **HMAC-SHA1** pour l'authentification/intégrité des paquets. Les clés de session SRTP sont généralement négociées via **SDES dans le SDP**, qui est lui-même protégé par TLS (d'où l'intérêt d'avoir TLS actif). Ainsi, TLS sécurise la signalisation (échanges SIP, mots de passe, clés), et SRTP sécurise le contenu de l'appel (voix chiffrée de bout en bout entre client et serveur).

Je retourne dans le fichier de configuration des utilisateurs

Et je rajoute ses directives

```
[6000]
type=friend
host=dynamic
allow=ulaw
fullname = Sadek Adel
username=asadek
secret=test
context=work

encryption=yes
avpf=yes
force_avp=yes
icesupport=yes
[6001]
type=friend
host=dynamic
allow=ulaw
fullname = Choucheh Sadek
username=choucheh
secret=test
context=work
encryption=yes
avpf=yes
force_avp=yes
icesupport=yes
```

**encryption=yes**
Active **SRTP** pour ce peer. Le média (voix RTP) est chiffré (AES-CM 128 + HMAC-SHA1). Les clés SRTP sont échangées via **SDES** dans le SDP (donc à utiliser avec **TLS** pour protéger cet échange).

**avpf=yes**
Active le profil **RTP/AVPF** (RFC 4585) au lieu de RTP/AVP. Ça permet le **feedback RTCP rapide** (utile avec WebRTC et certains clients modernes).

**force_avp=yes**
Force l'utilisation du **profil AVP/AVPF** même si le client ne le propose pas clairement. Utile pour assurer la compatibilité avec SRTP/clients WebRTC, mais peut casser avec de vieux téléphones SIP.

**icesupport=yes**
Active **ICE (Interactive Connectivity Establishment)** pour aider à traverser le NAT. Le client propose des candidats (IP/ports) et Asterisk choisit le meilleur chemin média. Utile surtout pour clients mobiles/WebRTC.

**En résumé**

- encryption=yes → **chiffre la voix (SRTP)**
- avpf=yes / force_avp=yes → **profil RTP moderne + feedback**
- icesupport=yes → **meilleure traversée NAT**


Sur ce test, nous avons utilisé le logiciel **MicroSIP** sur Windows afin de simuler un client SIP simple compatible avec TLS et SRTP.

La capture montre le journal d'exécution d'Asterisk lors d'un appel entre deux extensions. On y observe que la signalisation est bien établie, les deux clients sont enregistrés et l'appel est accepté (answered), puis les canaux sont reliés dans un

simple_bridge. On voit également les mécanismes de **Strict RTP learning**, où Asterisk apprend dynamiquement les adresses sources des flux audio, ce qui confirme que le trafic média circule.

Cependant, un problème de flux est apparu : les clients tentaient initialement de communiquer **directement entre eux (peer-to-peer)** pour le média RTP. Ce comportement est classique en SIP mais pose problème dans un contexte réel, notamment derrière du NAT ou lorsque certains flux VoIP ne sont pas autorisés ou filtrés.

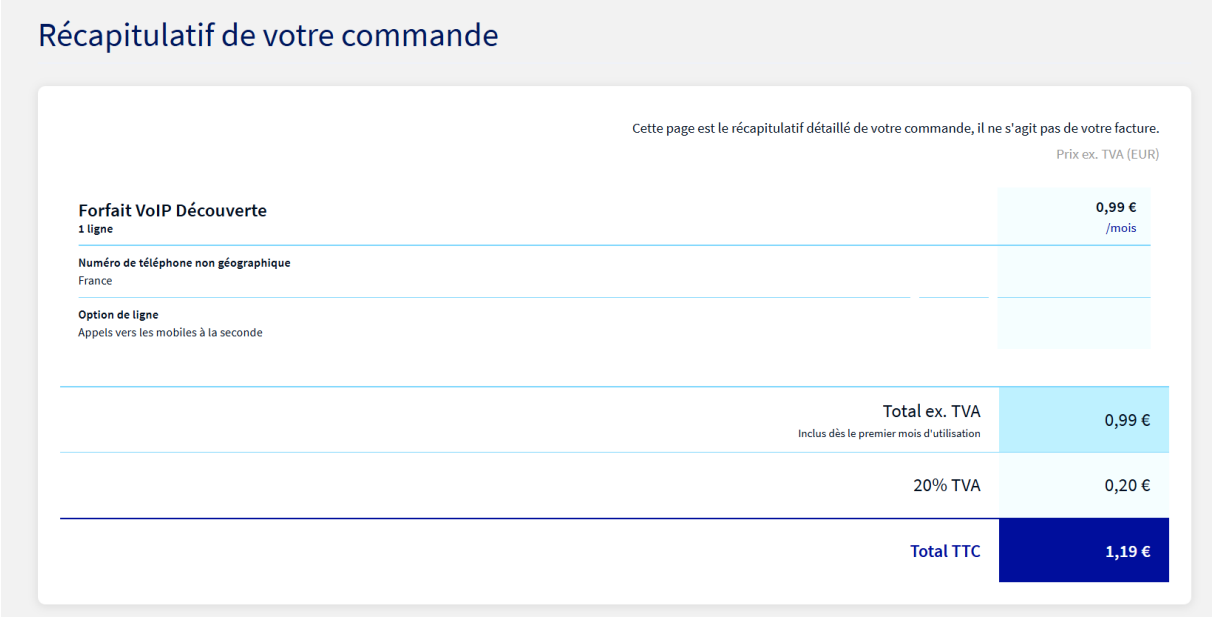
Pour corriger cela, nous avons appliqué des directives côté Asterisk (directmedia=no, nat=force_rport,comedia, etc.) afin de **forcer le passage du flux audio via le serveur Asterisk**. Cette approche garantit un meilleur contrôle du trafic, une compatibilité accrue avec les environnements réseau contraints et permet d'assurer la continuité du service même lorsque les communications directes entre clients ne sont pas possibles.

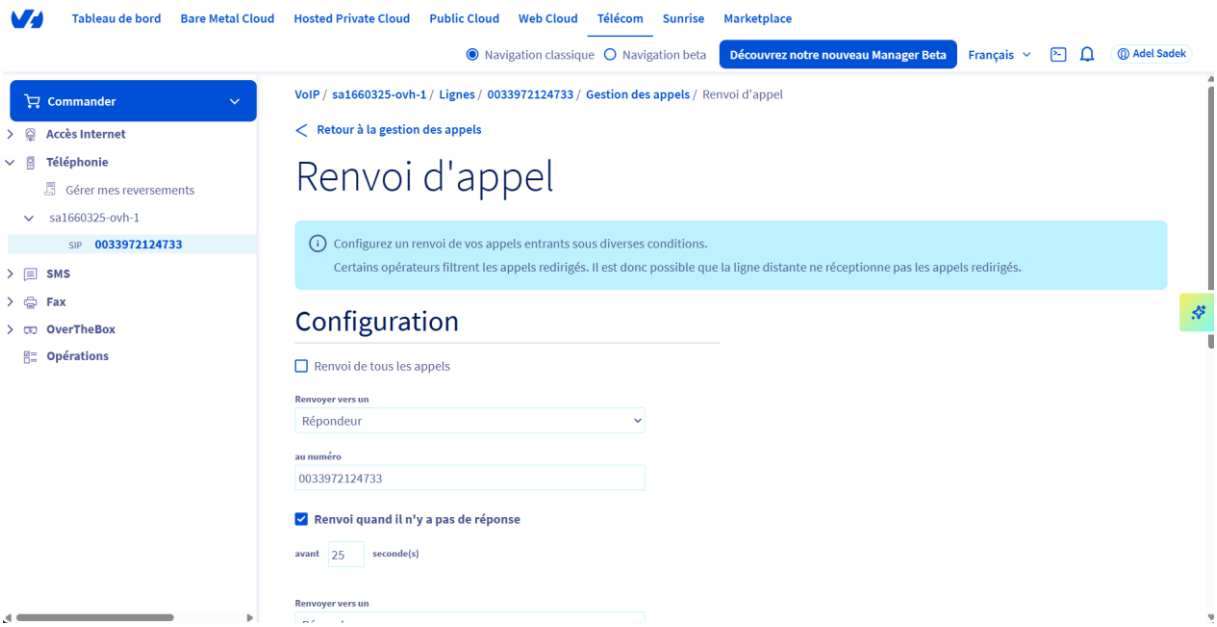
## Ajout d'un trunk OVH

Je vais me baser sur cette documentation : https://help.ovhcloud.com/csm/fr-voip-creer-redirection-presentation?id=kb_article_view&sysparm_article=KB0039282

Je dois disposer d'un numéro alias OVH + une ligne SIP ovh cloud

Je choisis cette offre je vais voir si ça englobe tout le numero alias et la ligne VOIP ou juste la ligne VOIP

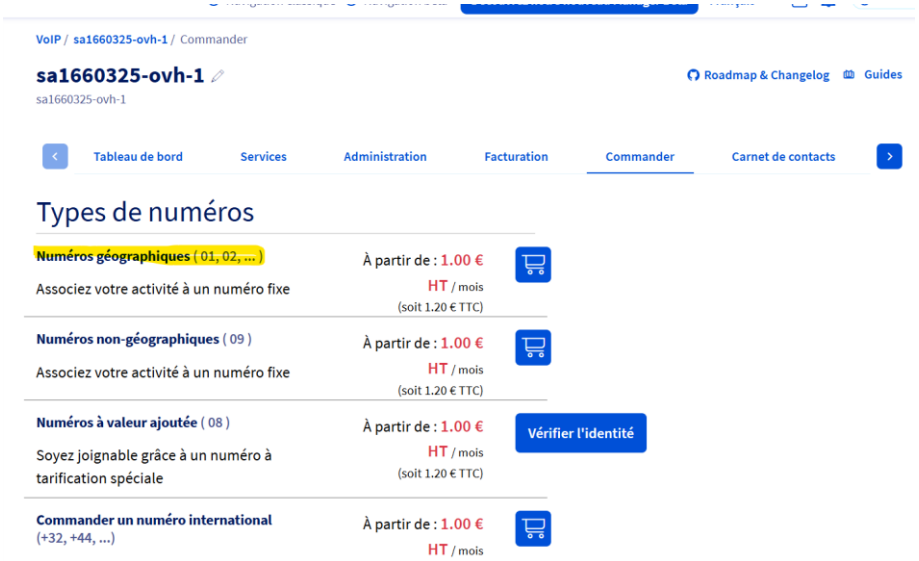



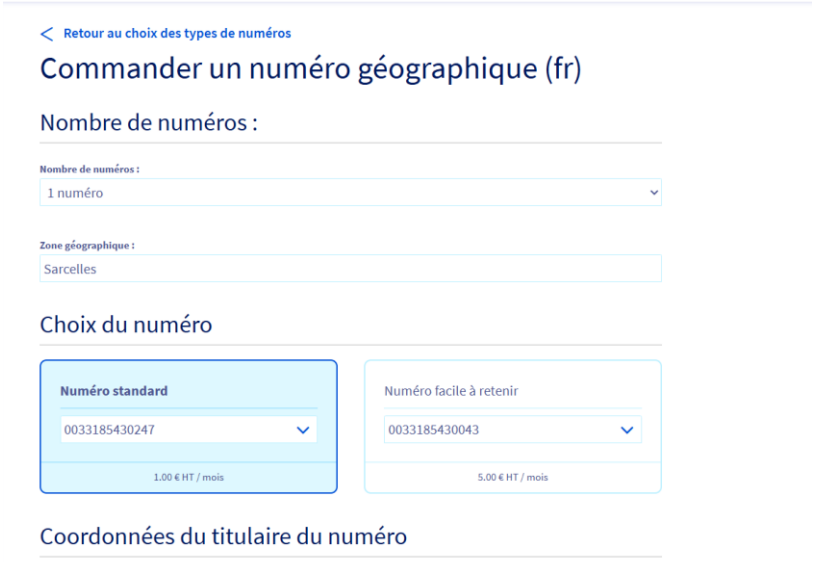



Je ne peux pas configurer maintenant car il faut que je prenne un numero d’alias

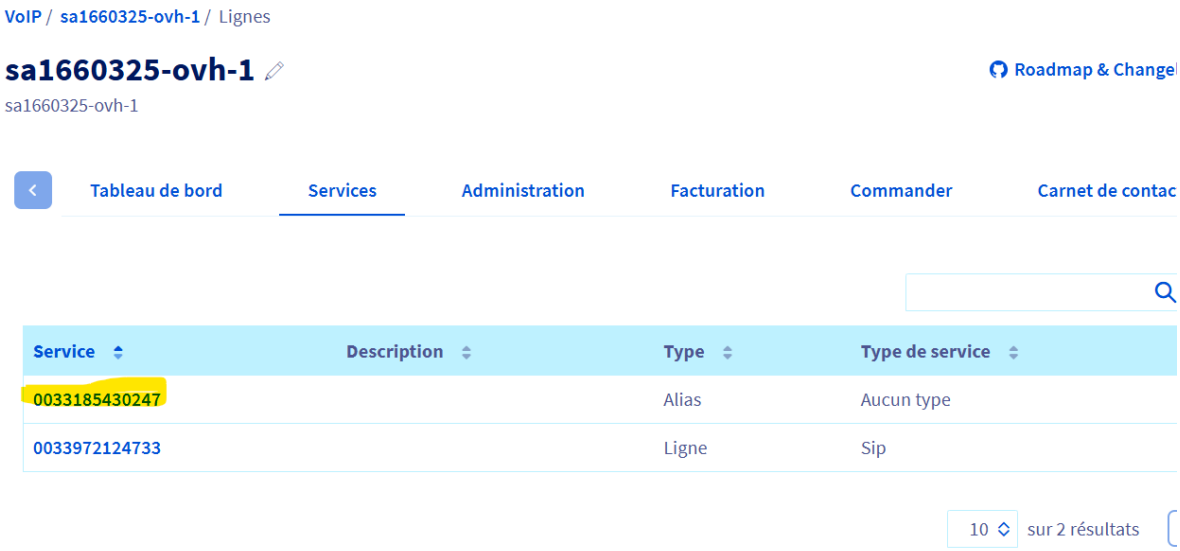
https://www.ovhcloud.com/fr/phone/numeros/

Je re appuie sur commander ici

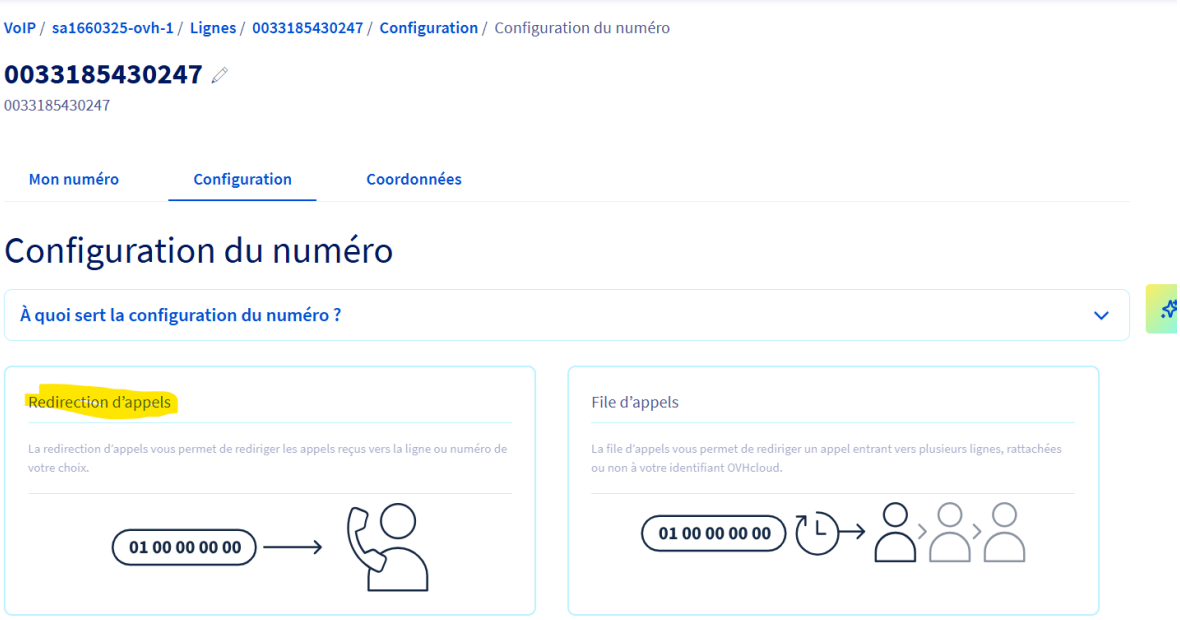


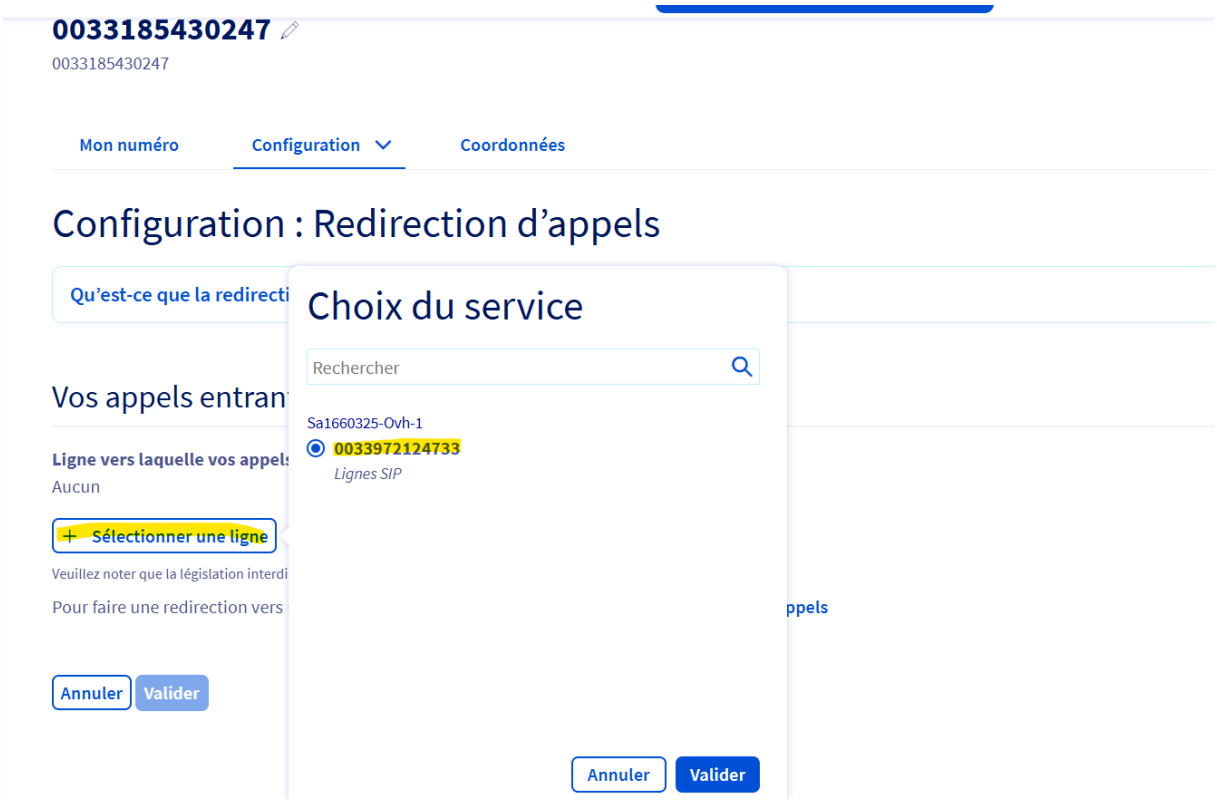

Commande traité je patiente un peu



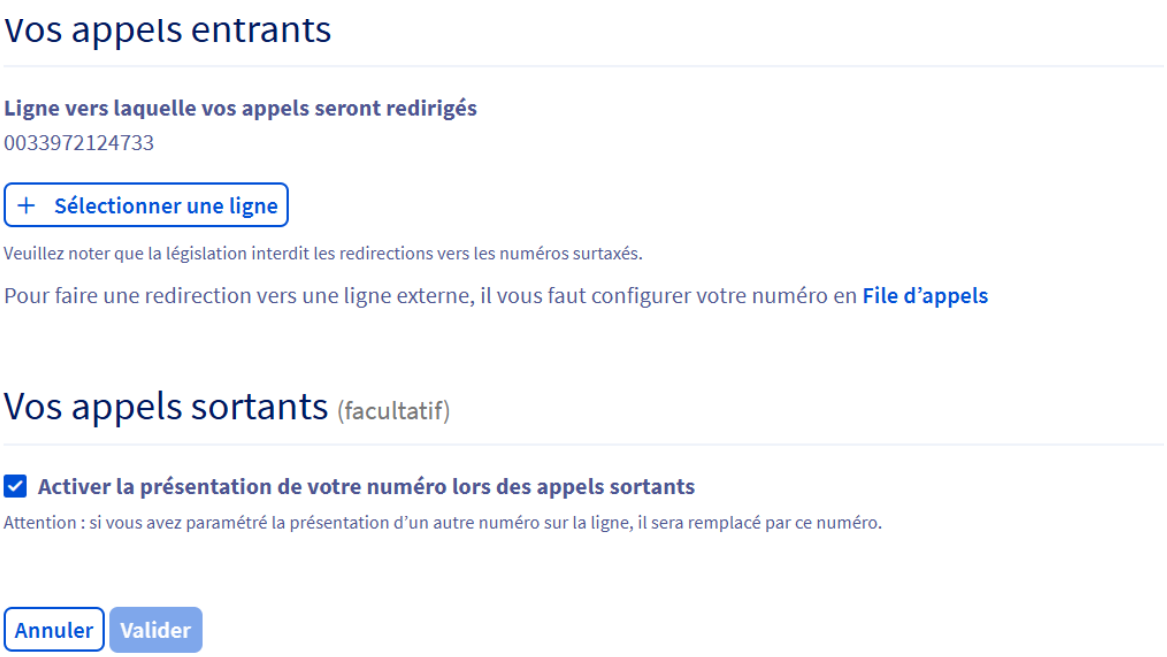

Je configure cette ligne




Ensuite en bas cliquer paramétrer



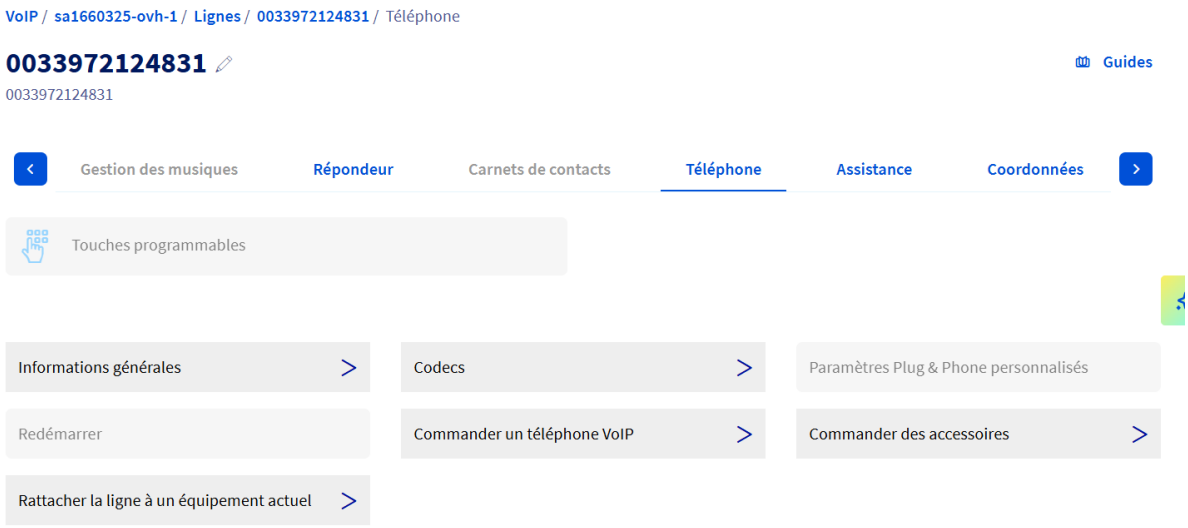

Ensuite je valide



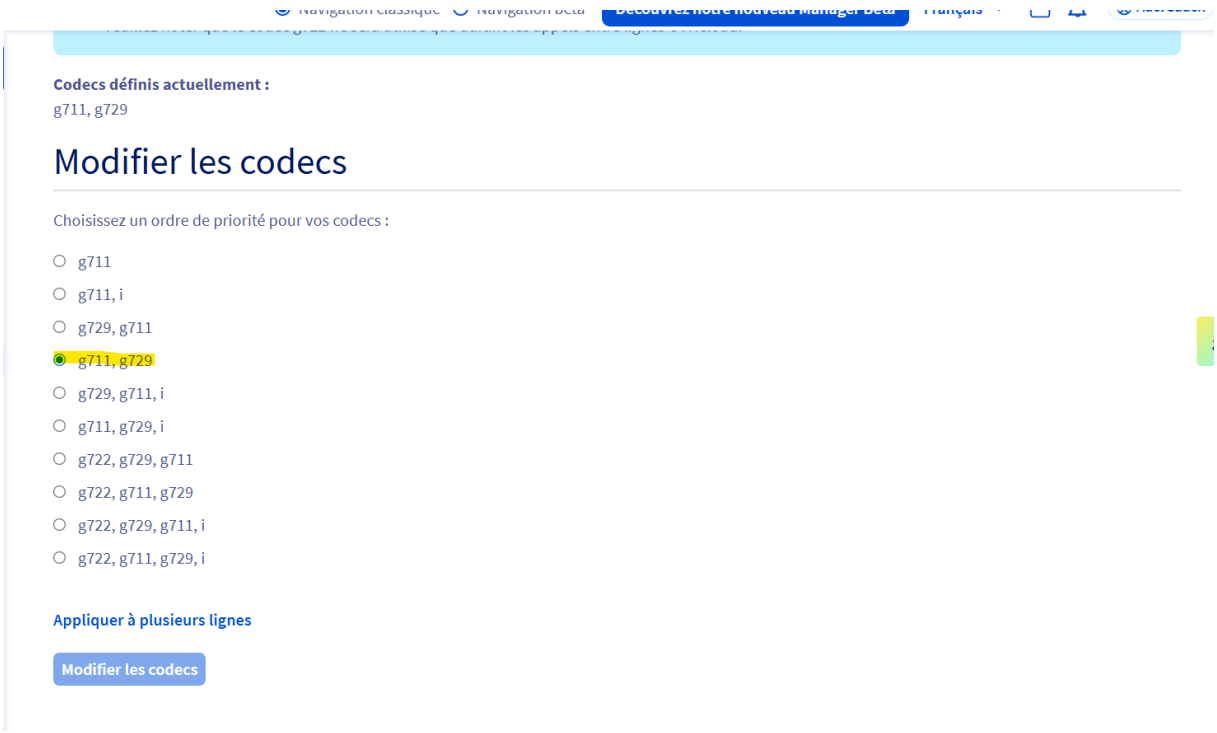

Ensuite il faut acheter un SIP TRUNK

https://www.ovhcloud.com/fr/phone/sip-trunk/

- Connectez-vous à votre espace client OVHcloud : https://www.ovhtelecom.fr/espaceclient/
- Cliquez sur le lien "**Accéder à l'ancienne interface**".
- Cliquez sur l'icône "**Téléphonie**".
- Cliquez sur votre trunk.
- Cliquez sur "**Téléphone**" dans le menu "**Navigation**".
- Cliquez sur "**Codecs**".
- Cliquez sur "**Gérer**".
- Cochez la case "**Amélioration de la présentation du numéro appelé**".
- Cliquez sur "**Valider**" pour confirmer la configuration.




On met ce codec pour améliorer la compatibilité




Ce rendre ensuite dans le fichier **sip.conf**

- Pour copier votre fichier : **cp /etc/asterisk/sip.conf /etc/asterisk/sip.conf.bak**
- Pour ouvrir votre fichier sip.conf : **vim /etc/asterisk/sip.conf**

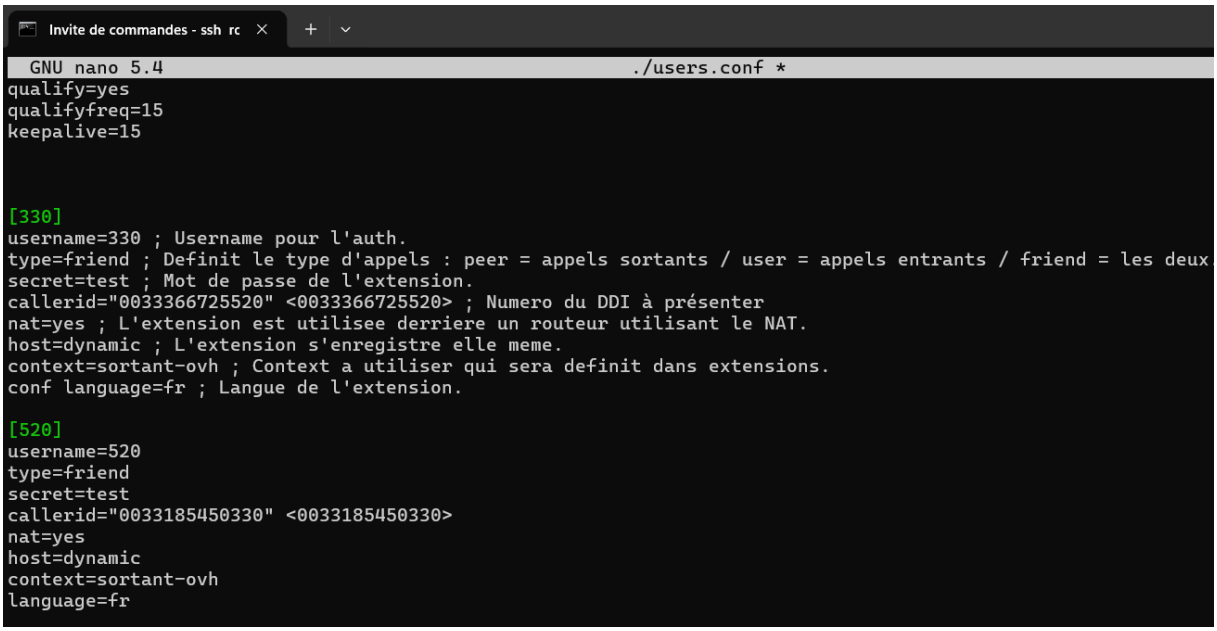
On utilise un nouveau fichier vierge

Je modifie le mot de passe SIP ici

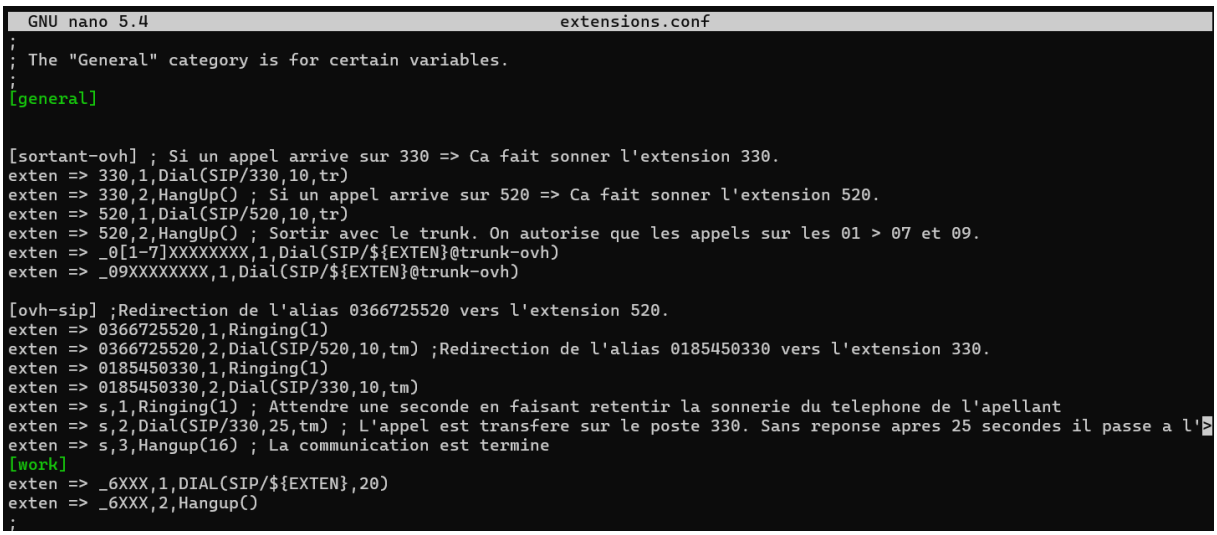

```
[general]
defaultexpiry=1800 ; Temps de register de la ligne.
context=trunk-ovh ; Nom du context pour le trunk dans sip.conf
bindport=5060 ; Port d'ecoute.
bindaddr=0.0.0.0 ; Adresse d'ecoute.
srvlookup=no ; Autoriser les appels via noms DNS
register => 0033972124831[redacted]@siptrunk.ovh.net ; Authentfication du trunk. La syntaxe est username:p
disallow=all ; Gestion des codecs pour autoriser que le G7111
allow=ulaw   ; Gestion des codecs pour autoriser que le G7111
allow=alaw   ; Gestion des codecs pour autoriser que le G7111

[trunk-ovh]
type=friend ; Definit le type d'appels : peer = appels sortants / user = appels entrants / friend = les c
host=siptrunk.ovh.net ; Nom du serveur SIP du trunk.
context=ovh-sip ; Nom du contexte pour le trunk dans extensions.conf et gérer les appels entrants.
language=fr ; Langue de la ligne.
insecure=invite,port
username=0033972124831 ; Username du trunk.
secret=[redacted] ; Mot de passe du trunk.
```

User.conf



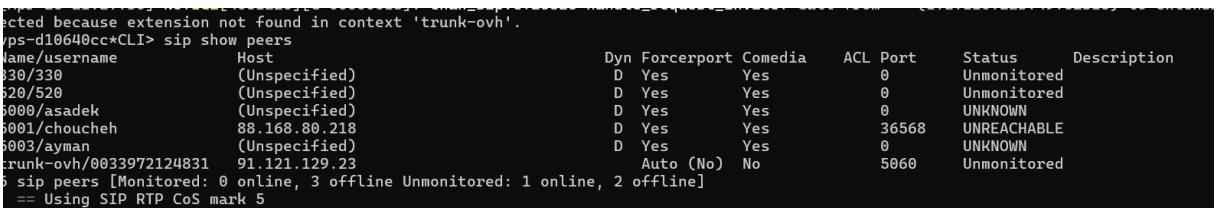

Extension.conf




Ensuite je redemarre asterisk

Pour afficher tout les trunk

sip show peers



Voici un bloc de doc sérieux que tu peux intégrer directement.

---

**Configuration du trunk SIP OVH sur Asterisk avec chan_sip**

Dans notre cas, la ligne OVH SIP Trunk fournit les informations suivantes :

Numéro / Login SIP : 0033972124831

Authorization user name : 0033972124831

Domain / Registrar : sip-domain.io

Proxy sortant : sa1660325-ovh-1.sip-proxy.io

IP résolue du proxy : 137.74.239.8

IP publique du VPS Asterisk : 144.217.14.63

Le problème rencontré était que le domaine sip-domain.io ne se résolvait pas correctement depuis le VPS. Asterisk essayait donc d'envoyer le REGISTER vers une destination nulle, ce qui produisait des erreurs de type :

getaddrinfo("sip-domain.io"): No address associated with hostname

sip_xmit ... to (null) returned -1: Invalid argument

En testant directement le proxy sortant OVH, on constatait qu'il répondait bien :

nslookup sa1660325-ovh-1.sip-proxy.io

Résultat :

sa1660325-ovh-1.sip-proxy.io canonical name = asbc.n11.prod.nowi.ovh

Address: 137.74.239.8

Cependant, enregistrer directement le trunk sur le proxy sortant provoquait une erreur :

SIP/2.0 404 Domain not bound

Cela signifie que le proxy est bien le point d'entrée réseau, mais que le domaine SIP attendu par OVH reste sip-domain.io.

La solution consiste donc à forcer localement la résolution de sip-domain.io vers l'IP du proxy OVH :

echo "137.74.239.8 sip-domain.io" >> /etc/hosts

Ainsi, Asterisk continue d'utiliser le bon domaine SIP sip-domain.io, tout en envoyant réellement les paquets vers l'infrastructure OVH.

---

**Fichier /etc/asterisk/sip.conf**

```
[general]
context=default
bindport=5060
bindaddr=0.0.0.0
defaultexpiry=1800
srvlookup=yes

; NAT / IP publique du serveur Asterisk
externip=144.217.14.63
localnet=127.0.0.0/255.0.0.0

; Sécurité de base
allowguest=no
alwaysauthreject=yes

; Codecs autorisés
disallow=all
allow=alaw
allow=ulaw

; Enregistrement SIP OVH
; Format :
; register => login:motdepasse@registrar/numero
register => 0033972124831:votremotdepasse@sip-domain.io/0033972124831



[trunk-ovh]
```

type=peer

; Proxy OVH utilisé comme point d’entrée réseau

host=sa1660325-ovh-1.sip-proxy.io

; Domaine SIP attendu par OVH pour l'identité de la ligne

fromdomain=sip-domain.io

; Identifiants SIP OVH

username=0033972124831

fromuser=0033972124831

authuser=0033972124831

secret=votremotdepasse

; Contexte utilisé pour router les appels entrants dans extensions.conf

context=ovh-sip

language=fr

; Options SIP

insecure=invite,port

qualify=yes

nat=force_rport,comedia

; Codecs du trunk

disallow=all

allow=alaw

allow=ulaw

---

**Vérification de l’enregistrement**

Après modification du fichier :

asterisk -rx "sip reload"

asterisk -rx "sip show registry"

Le résultat attendu est :

Host Username Refresh State Reg.Time

sip-domain.io 0033972124831 1800 Registered

Dans le debug SIP, la confirmation correcte apparaît sous cette forme :

SIP/2.0 200 OK

Outbound Registration: Expiry for sip-domain.io is 1800 sec

---

**Explication du problème**

Au départ, plusieurs erreurs pouvaient faire croire à un problème d’identifiants ou de configuration Asterisk. En réalité, il y avait deux points distincts :

1. sip-domain.io ne se résolvait pas correctement depuis le VPS.
2. Le proxy OVH sa1660325-ovh-1.sip-proxy.io répondait bien, mais ne devait pas remplacer le domaine SIP dans le REGISTER.

Quand le REGISTER était envoyé directement vers :

sa1660325-ovh-1.sip-proxy.io

OVH répondait :

404 Domain not bound

Cela indiquait que le domaine utilisé dans le REGISTER n’était pas celui attendu par l’infrastructure OVH.

La bonne logique est donc :

Domaine SIP / Registrar : sip-domain.io

Proxy réseau réel : sa1660325-ovh-1.sip-proxy.io

Comme sip-domain.io ne se résolvait pas, on l’a associé localement à l’IP du proxy OVH dans /etc/hosts.

---

**Validation finale**

Après correction du mot de passe SIP et de la résolution locale, OVH a répondu :

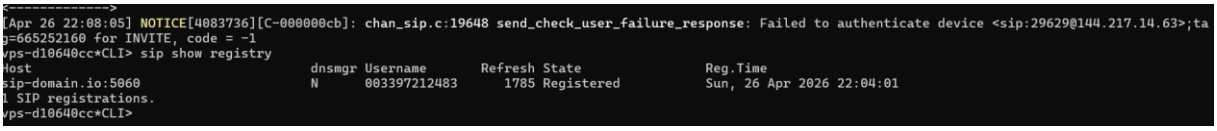
SIP/2.0 401 Unauthorized

Cette réponse est normale : le serveur demande une authentification SIP Digest.

Asterisk a ensuite renvoyé un REGISTER avec l'en-tête Authorization, puis OVH a répondu :

SIP/2.0 200 OK

Ce 200 OK confirme que le trunk SIP OVH est correctement enregistré sur Asterisk.




Fichier extension.conf

```
[sortant-ovh]
; Appels internes
exten => 330,1,Dial(SIP/330,10,tr)
 same => n,Hangup()

exten => 520,1,Dial(SIP/520,10,tr)
 same => n,Hangup()

; Sortie via trunk OVH
; Autorise 01 à 07
exten => _0[1-7]XXXXXXXX,1,NoOp(Appel sortant OVH vers ${EXTEN})
 same => n,Dial(SIP/trunk-ovh/${EXTEN},60)
 same => n,Hangup()

; Autorise 09
exten => _09XXXXXXXX,1,NoOp(Appel sortant OVH vers ${EXTEN})
 same => n,Dial(SIP/trunk-ovh/${EXTEN},60)
 same => n,Hangup()


[ovh-sip]
; Appel entrant sur le numéro OVH au format international
exten => 0033972124831,1,NoOp(Appel entrant OVH 0033972124831)
 same => n,Ringing()
 same => n,Dial(SIP/520,25,tm)
 same => n,Hangup()

; Appel entrant sur le numéro OVH au format national
exten => 0972124831,1,Goto(ovh-sip,0033972124831,1)

; Fallback si OVH envoie l'appel sur "s"
exten => s,1,NoOp(Appel entrant OVH sans DID)
 same => n,Ringing()
 same => n,Dial(SIP/520,25,tm)
 same => n,Hangup()
```

J’ai relaod le diaplan et je l’affiche pour voir si la cofn a bien été prise en compte

```
root@vps-d10640cc:/etc/asterisk# asterisk -rx "dialplan reload"
asterisk -rx "dialplan show ovh-sip"
asterisk -rx "dialplan show sortant-ovh"
Dialplan reloaded.
[ Context 'ovh-sip' created by 'pbx_config' ]
  '0033972124831' => 1. NoOp(Appel entrant OVH 0033972124831)     [extensions.conf:37]
                     2. Ringing()                                 [extensions.conf:38]
                     3. Dial(SIP/520,25,tm)                       [extensions.conf:39]
                     4. Hangup()                                  [extensions.conf:40]
  '0972124831' =>    1. Goto(ovh-sip,0033972124831,1)             [extensions.conf:43]
  's' =>             1. NoOp(Appel entrant OVH sans DID)          [extensions.conf:46]
                     2. Ringing()                                 [extensions.conf:47]
                     3. Dial(SIP/520,25,tm)                       [extensions.conf:48]
                     4. Hangup()                                  [extensions.conf:49]

-= 3 extensions (9 priorities) in 1 context. =-
[ Context 'sortant-ovh' created by 'pbx_config' ]
  '330' =>           1. Dial(SIP/330,10,tr)                       [extensions.conf:17]
                     2. Hangup()                                  [extensions.conf:18]
  '520' =>           1. Dial(SIP/520,10,tr)                       [extensions.conf:20]
                     2. Hangup()                                  [extensions.conf:21]
  '_09XXXXXXXX' =>   1. NoOp(Appel sortant OVH vers ${EXTEN})     [extensions.conf:30]
                     2. Dial(SIP/trunk-ovh/${EXTEN},60)           [extensions.conf:31]
                     3. Hangup()                                  [extensions.conf:32]
  '_0[1-7]XXXXXXXX' => 1. NoOp(Appel sortant OVH vers ${EXTEN})     [extensions.conf:25]
                     2. Dial(SIP/trunk-ovh/${EXTEN},60)           [extensions.conf:26]
                     3. Hangup()                                  [extensions.conf:27]

-= 4 extensions (10 priorities) in 1 context. =-
root@vps-d10640cc:/etc/asterisk#
```

Parfait, là on va faire une **suite propre, claire et exploitable** pour ta doc, avec **explication du dialplan + ce qu’on a corrigé**.

**Suite – Routage des appels (Dialplan Asterisk)**

**Introduction**

Une fois le trunk SIP OVH correctement enregistré, la prochaine étape consiste à **router les appels**.

Dans Asterisk, cela se fait via le fichier :

/etc/asterisk/extensions.conf

Ce fichier contient ce qu’on appelle le **dialplan**.

Le dialplan = **la logique du serveur téléphonique**

- Qui appelle qui
- Comment sortir vers l’extérieur
- Comment gérer les appels entrants

**Structure du dialplan**

Un dialplan est organisé en **contextes** :

[sortant-ovh]

[ovh-sip]

Chaque contexte correspond à :

- un type d'appel
- ou une source (interne / externe)

---

**1. Contexte des appels sortants**

[sortant-ovh]

; Appels internes

exten => 330,1,Dial(SIP/330,10,tr)

same => n,Hangup()

exten => 520,1,Dial(SIP/520,10,tr)

same => n,Hangup()

; Appels vers l'extérieur (01 à 07)

exten => _0[1-7]XXXXXXXX,1,NoOp(Appel sortant OVH vers ${EXTEN})

same => n,Dial(SIP/${EXTEN}@sip-domain.io,60)

same => n,Hangup()

; Appels vers les numéros en 09

exten => _09XXXXXXXX,1,NoOp(Appel sortant OVH vers ${EXTEN})

same => n,Dial(SIP/${EXTEN}@sip-domain.io,60)

same => n,Hangup()

---

**Explication détaillée**

**Appels internes**

exten => 330,1,Dial(SIP/330,10,tr)

Si quelqu'un compose **330**

- Asterisk appelle l'extension SIP 330
- pendant 10 secondes

**Pattern des numéros**

_0[1-7]XXXXXXXX

Ça veut dire :

- commence par 0
- suivi de 1 à 7
- puis 8 chiffres

✓ Exemples :

- 0123456789
- 0699850123

**Variable ${EXTEN}**

${EXTEN}

C'est le numéro composé

Exemple :

Dial(SIP/${EXTEN}@sip-domain.io)

➡️ devient :

Dial(SIP/0699860123@sip-domain.io)

---

**Pourquoi on a changé ça ?**

**Avant (erreur)**

Dial(SIP/${EXTEN}@trunk-ovh)

Résultat :

400 Wrong domain

👉 OVH refusait car :

- mauvais domaine SIP

---

**Après (corrigé)**

Dial(SIP/${EXTEN}@sip-domain.io)

Là :

- domaine correct
- authentification OK
- appel accepté

---

**2. Contexte des appels entrants OVH**

[ovh-sip]

; Appel entrant (format international)

exten => 0033972124831,1,NoOp(Appel entrant OVH)

same => n,Ringing()

same => n,Dial(SIP/520,25,tm)

same => n,Hangup()

; Format national

exten => 0972124831,1,Goto(ovh-sip,0033972124831,1)

; Fallback

exten => s,1,NoOp(Appel entrant OVH sans DID)

same => n,Ringing()

same => n,Dial(SIP/520,25,tm)

same => n,Hangup()

---

**Explication**

**Cas 1 : OVH envoie ton numéro complet**

0033972124831

👉 Asterisk :

- fait sonner
- envoie vers extension 520

---

**Cas 2 : OVH envoie en local**

0972124831

On redirige vers le format international :

Goto(...)

---

**Cas 3 : OVH envoie "s"**

👉 Ça arrive souvent

exten => s,1,...

On met un fallback

---

**3. Ce qu'on a corrigé (important pour la doc)**

**1. Mauvais domaine SIP**

Erreur :

400 Wrong domain

✓ Correction :

fromdomain=sip-domain.io

Dial(...@sip-domain.io)

---

**2. Mauvais REGISTER**

Erreur :

404 Domain not bound

✓ Correction :

- garder sip-domain.io
- forcer DNS via /etc/hosts

---

**3. Mauvais mot de passe**

Erreur :

403 Wrong login or password

✓ Correction :

- mot de passe corrigé

---

**4. Authentification SIP**

407 authentication required

NORMAL

Asterisk :

- reçoit 407
- renvoie avec auth
- appel passe

---

**5. Résultat final**

183 Progress

180 Ringing

Signifie :

- appel accepté
- en cours
- ça sonne

---

**4. Validation du dialplan**

Commande :

asterisk -rx "dialplan show"

Permet de vérifier que :

- le contexte existe
- les règles sont chargées

---

**Conclusion**

On a mis en place :

✓ un serveur Asterisk fonctionnel
✓ un trunk OVH opérationnel
✓ un routage sortant fonctionnel
✓ un routage entrant prêt

---

**Étape suivante (logique)**

Tester :

1. appel entrant vers ton 09
2. vérifier que ça sonne extension 520
3. sinon debug :

asterisk -rvvv

sip set debug on

---

Si tu veux, prochaine étape :
**sécuriser ton serveur (fail2ban + firewall SIP)**
ou **connecter ton téléphone proprement (Zoiper / MicroSIP / iOS)**

## Appel depuis mon téléphone vers 520


## Gestion de la présentation du numéro (Caller ID) avec OVH

Dans le cadre de l'utilisation d'un trunk SIP OVH, la gestion du **Caller ID** (numéro présenté lors des appels sortants) obéit à certaines contraintes imposées par l'opérateur. Contrairement à une configuration purement locale Asterisk, il n'est pas possible d'afficher arbitrairement n'importe quel numéro. OVH n'autorise que la présentation de numéros **explicitement rattachés au trunk**, à savoir le numéro principal ou des numéros alias préalablement configurés dans l'espace client.

Pour mettre en place cette fonctionnalité, il est nécessaire d'acheter un numéro supplémentaire (alias) et de l'associer au trunk SIP OVH. Une fois cette association réalisée et les options de présentation activées côté OVH, il devient possible d'utiliser ce numéro comme identifiant sortant.

Côté Asterisk, la configuration peut se faire soit dans le fichier des utilisateurs via la directive callerid, soit de manière plus robuste directement dans le dialplan. L'utilisation du dialplan permet de contrôler dynamiquement le numéro présenté selon le contexte d'appel, indépendamment de la configuration des postes SIP.

Exemple dans le dialplan :

```
exten => _0[1-9]XXXXXXXX,1,NoOp(Appel sortant)
```

```
 same => n,Set(CALLERID(num)=0972124831)

 same => n,Dial(SIP/${EXTEN}@sip-domain.io)
```

Dans cet exemple, tous les appels sortants utiliseront le numéro 0972124831 comme identifiant. Il est également possible de définir plusieurs règles pour utiliser différents numéros selon les extensions ou les types d'appels.

En résumé, la gestion du Caller ID avec OVH repose sur une double configuration :

- côté opérateur, en déclarant les numéros autorisés,
- côté Asterisk, en définissant dynamiquement le numéro présenté dans le dialplan.

Toute tentative d'utilisation d'un numéro non autorisé peut entraîner un refus d'appel ou une substitution automatique par le numéro principal du trunk.

## Schema

Client SIP (TLS + SRTP)

↓

Internet

↓

Box NAT

↓

Asterisk (VPS)

↓

Trunk OVH

↓

PSTN

## Conclusion

Cette architecture permet de sécuriser les communications VoIP tout en assurant une compatibilité avec les contraintes réseau (NAT, IP dynamique). Toutefois, elle nécessite un durcissement supplémentaire pour un usage en production.