

# UNIVERSITE CHEIKH ANTA DIOP DE DAKAR



FACULTÉ DES SCIENCES ET TECHNIQUES

DEPARTEMENT DE PHYSIQUE

## MEMOIRE DE MASTER II EN PHYSIQUE ET APPLICATION

Spécialité : Electronique – Système et Télécommunications

Option : Réseaux et Télécommunications

### Etude et mise en place d'une plateforme open source de VoIP basée sur un Cloud

Présenté et soutenu le 03 juillet par :

DJIBY SENE

Membres du Jury :

Jury	Prénom(s) et Nom	Grade(Cames)	Etablissement
Président	◀ Fabé Idrissa BARRO ▶	Professeur Titulaire	UCAD/FST
Membres	◀ Ibrahima DIANE ▶	Maître de Conférences	UCAD/FST
	◀ Moustapha NDIAYE ▶	Docteur-Ingénieur	UCAD/ESP
Encadreur	◀ Balla Diop NGOM ▶	Professeur Titulaire	UCAD/FST

2021-2022



## **DEDICACES**

### **À ma mère, Séynabou FAYE,**

Maman, tu es ma force, mon refuge et ma plus grande bénédiction. Ton amour, tes sacrifices et tes prières m'ont porté jusqu'ici. Que Dieu te protège et te comble de bonheur.

### **À la mémoire de mon défunt père,**

Ton absence est un vide, mais ton amour et tes valeurs vivent en moi. Que Dieu t'accorde Sa miséricorde et t'accueille en Son paradis. Ce travail est un humble hommage à ta mémoire.

### **À ma chère épouse AMY,**

Merci pour ton amour, ta patience et ta présence rassurante. Ton soutien indéfectible dans les moments les plus difficiles a été une lumière sur mon chemin. Ce travail t'est aussi dédié.

### **À mes frères et sœurs,**

Pour leur soutien et leurs encouragements constants.

### **À mes encadreurs, Dr-Ing Moustapha NDIAYE et Pr Balla Diop NGOM,**

Pour leur précieuse guidance, leur disponibilité et la confiance qu'ils m'ont accordée tout au long de ce travail.

### **À tous ceux qui m'ont inspiré et soutenu,**

Je vous dédie ce travail avec gratitude.



Avant tout, je rends grâce à **Dieu Tout-Puissant**, créateur de l'univers, pour Sa guidance infinie, Sa clémence et Sa miséricorde.

Je salue également **notre Prophète Mohamed (PSL)**, guide et modèle pour l'humanité, dont l'enseignement, la sagesse et les valeurs demeurent une source d'inspiration éternelle. Que la paix et les bénédictions d'Allah soient sur lui, sa famille et ses compagnons.

Je rends hommage à **Serigne Touba**, le guide spirituel et fondateur de la **Tarîqat Mouride**, dont l'héritage spirituel et l'engagement pour le bien-être de l'humanité sont un phare de lumière et de sagesse. Que ses prières continuent d'éclairer nos vies et que ses bénédictions soient avec nous à chaque étape de notre chemin.

Je tiens à exprimer ma profonde gratitude à **ma mère, Séynabou Faye**, dont l'amour inconditionnel, les prières et les sacrifices ont toujours été ma plus grande source de motivation. Maman, ce travail est en partie le fruit de ton dévouement et de ton soutien indéfectible.

À la mémoire de mon défunt père, qui demeure une source d'inspiration pour moi. Que Dieu l'accueille en Son paradis et lui accorde Sa miséricorde.

Je tiens également à remercier **mon beau-père, Mamadou DIOUF**, pour sa bienveillance et son soutien constant. Ta sagesse et tes encouragements ont été un appui précieux dans ma vie.

Je remercie également **mes frères et sœurs**, qui m'ont toujours soutenu et encouragé avec bienveillance tout au long de mon parcours.

Je suis particulièrement reconnaissant envers **ma promotion de Master II en électronique, systèmes et télécommunications**, pour la camaraderie, l'entraide et la motivation partagée tout au long de cette année universitaire.

Un grand merci à **ma tante Fatou FAYE** et à son mari, pour leur soutien constant et leur encouragement. Votre présence et vos conseils ont été une source de réconfort pour moi.

Ma reconnaissance va aussi à **mes encadreurs, Dr-Ing Moustapha NDIAYE et Pr Balla Diop NGOM**, pour leur patience, leurs conseils éclairés et leur accompagnement tout au long de ce projet. Leur expertise et leur disponibilité ont été essentielles à la réussite de ce travail.

Enfin, je remercie **toutes les personnes** qui, de près ou de loin, ont contribué à la réalisation de ce travail. Votre soutien m'a été précieux, et je vous en suis profondément reconnaissant.

## **SYNOPSIS AVANT-PROPOS**

### **SYNOPSIS**

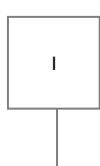
Ce mémoire se focalise sur l'intégration harmonieuse de la Voix sur IP (VoIP) et du Cloud Computing. Notre étude approfondie vise à explorer en profondeur les mécanismes sous-jacents et les avantages découlant de cette convergence technologique innovante.

### **AVANT-PROPOS**

La Faculté des Sciences et Techniques de l'Université CHEIKH ANTA DIOP DE DAKAR (l'UCAD) constituent l'épicentre de cette étude d'envergure. Nous tenons à exprimer notre sincère gratitude envers cette institution académique de renom. Cet avant-propos sert d'introduction chaleureuse, mettant en lumière l'importance cruciale de cette recherche, située au cœur des évolutions technologiques actuelles.

À l'achèvement de cette formation de deuxième cycle (Master 2) en Réseaux Informatiques et Télécommunications, chaque étudiant se voit attribuer un sujet de mémoire. Dans mon cas, j'ai opté pour le thème :

« *Étude et mise en place d'une plateforme open source de VoIP basée sur un Cloud* ». Proposé par deux éminents professeurs encadreurs : Professeur Balla Diop NGOM et Docteur-Ingénieur Moustapha NDIAYE.



# RESUME

Ce mémoire explore la conception, le déploiement et l'optimisation d'une plateforme de Voix sur IP (VoIP) basée sur Astérisk, une solution open source, et hébergée dans un environnement Cloud AWS. Il met en lumière la synergie entre ces deux technologies, offrant une alternative performante, évolutive et économique aux systèmes de communication traditionnels.

L'objectif principal de cette étude est de développer un système de communication vocale fiable et performant, garantissant une transmission de haute qualité sur un réseau IP tout en exploitant les atouts du Cloud. Parmi ces avantages figurent l'évolutivité, qui permet d'adapter dynamiquement les ressources en fonction des besoins, la flexibilité, qui simplifie la gestion et la configuration des services, ainsi que la haute disponibilité, assurant une continuité de service même en cas de panne.

Cette étude approfondit les principes fondamentaux de la VoIP, en détaillant les protocoles de signalisation (SIP, H323, IAX etc.), les codecs audio et vidéo, ainsi que les contraintes liées à la qualité de service (QoS). Elle propose également une analyse comparative des solutions open source disponibles sur le marché et des infrastructures Cloud adaptées aux besoins d'une plateforme VoIP, avant de justifier le choix d'Astérisk et d'AWS.

Enfin, le mémoire décrit en détail l'implémentation technique de la plateforme, en couvrant son architecture globale, le processus d'installation et de configuration, ainsi que les différentes optimisations mises en place pour assurer une meilleure qualité des appels et une gestion efficace des ressources réseau. Des tests de performance ont été réalisés afin d'évaluer la qualité de service et d'identifier les éventuels axes d'amélioration. L'étude met en évidence les défis techniques rencontrés et propose des solutions adaptées pour garantir une expérience utilisateur optimale dans un environnement Cloud.

Ce travail démontre que l'intégration d'Astérisk sur AWS offre une solution VoIP fiable, évolutive et optimisée pour les besoins modernes de communication.

**Mots clef:** VoIP, Cloud AWS, Open source Astérisk, SIP, RTP



## **ABSTRACT**

This thesis explores the design, deployment, and optimization of a Voice over IP (VoIP) platform based on Asterisk, an open-source solution, and hosted in an AWS Cloud environment. It highlights the synergy between these two technologies, offering a high-performance, scalable, and cost-effective alternative to traditional communication systems.

The primary goal of this study is to develop a reliable and efficient voice communication system, ensuring high-quality transmission over an IP network while leveraging the benefits of the Cloud, such as scalability, which allows dynamic resource allocation based on demand, flexibility, which simplifies service management and configuration, and high availability, ensuring continuous operation even in the event of failures.

This study delves into the fundamental principles of VoIP, detailing signaling protocols (SIP, H323, IAX etc.), audio codecs, and quality of service (QoS) constraints. It also provides a comparative analysis of open-source solutions available on the market and Cloud infrastructures suited for a VoIP platform before justifying the choice of Asterisk and AWS.

Finally, the thesis provides a detailed description of the technical implementation of the platform, covering its overall architecture, installation, and configuration process, as well as various optimizations to ensure better call quality and efficient network resource management. Performance tests were conducted to assess the quality of service and identify potential areas for improvement. The study highlights the technical challenges encountered and proposes adapted solutions to guarantee an optimal user experience in a Cloud-based environment.

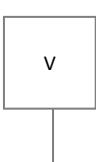
This work demonstrates that the integration of Asterisk on AWS provides a reliable, scalable, and optimized VoIP solution tailored to modern communication needs.

**Keywords:** VoIP, AWS Cloud, Open source Asterisk SIP, RTP.

## TABLE DES MATIERES

SYNOPSIS AVANT-PROPOS.....	I
RESUME.....	II
ABSTRAC .....	III
LISTE DES FIGURES .....	VII
LISTE DES TABLEAUX.....	IX
GLOSSIARE .....	X
INTRODUCTION GENERALE.....	1
1. CONTEXTE.....	2
2. PROBLEMATIQUE .....	2
3. OBJECTIFS.....	3
4. METHODOLOGIES .....	4
<b>CHAPITRE I : GENERALITES SUR LA VoIP.....</b>	<b>5</b>
1.1. PRINCIPE DE FONCTIONNEMENT DE LA VOIP .....	6
1.1.1. Acquisition du signal.....	6
1.1.2. Numérisation.....	7
1.1.3. Compression .....	7
1.1.4. Habillage des en têtes .....	7
1.1.5. Emission et transport .....	8
1.1.6. Réception.....	8
1.1.7. Conversation numérique analogique.....	9
1.1.8. Restitution du signal.....	9
1.2. MODES D'ACCES ET ARCHITECTURE.....	9
1.3. LES PROTOCOLES DE SIGNALISATION.....	11
1.3.1. Le protocole SIP.....	11
1.3.2. Le protocole H323.....	13
1.3.3. Le protocole IAX.....	14
1.4. CODECS .....	15
1.5. CONTRAINTES LIEES A LA VOIP.....	17
<b>CHAPITRE II : LE CLOUD COMPUTING.....</b>	<b>18</b>
2.1. CONCEPTS DE LA VIRTUALISATION.....	19
2.1.1. Virtualisation complète.....	20
2.1.2. Para virtualisation.....	20
2.1.3. Virtualisation assistée par extensions matérielles.....	21

2.2. PRESENTATION DU CLOUD COMPUTING.....	22
2.2.1. Les différents services du cloud computing.....	22
2.2.2. Les différentes solutions de cloud computing.....	23
2.3. AVANTAGES ET INCONVENIENTS DU CLOUD.....	24
2.3.1. Avantages du cloud computing .....	24
2.3.2. Inconvénients du cloud computing .....	25
2.4. INTEROPERABILITE DANS LE CLOUD.....	25
2.5. HAUTE DISPONIBILITE.....	26
2.5.1. Les caractéristiques de la haute disponibilité.....	26
2.5.2. Techniques de bases .....	26
<b>CHAPITRE III : ETUDE COMPARATIVE DES DIFFERENTES SOLUTIONS OPEN SOURCE ET DE CLOUD.....</b>	<b>28</b>
3.1. ETUDE COMPARATIVE DES SOLUTIONS OPEN SOURCE.....	29
3.1.1. Etude des différents serveurs de communication Open source PABX.	29
3.1.2. Tableau récapitulatif.....	30
3.1.3. Choix de solution : Cas d'Astérisk pour la VoIP.....	30
3.2. ETUDE COMPARATIVE DU CLOUD COMPUTING .....	31
3.3. CHOIX DE SOLUTION : CAS AWS .....	34
<b>CHAPITRE IV : IMPLEMENTATION DE LA SOLUTION.....</b>	<b>36</b>
4.1. LES ACTEURS INTERVENANT DU SYSTEME.....	37
4.1.1. Administrateur cloud.....	37
4.1.2. Administrateur entreprise .....	37
4.1.3. Utilisateur de VoIP.....	37
4.2. ARCHITECTURE DE LA SOLUTION PROPOSEE.....	37
4.3. IMPLEMENTATION AVEC AWS.....	40
4.3.1. Préparation de l'environnement AWS.....	42
4.3.2. Installation du serveur Astérisk .....	55
4.3.3. Configuration du serveur Astérisk.....	56
4.3.4. Configuration des services téléphonique (fonctionnalités) dans Astérisk .	59
4.3.5. Test des fonctionnalités avec le serveur Astérisk.....	65
4.3.6. Evaluation de la Qualité de service (QoS) .....	69
CONCLUSION GÉNÉRALE .....	71
RÉFÉRENCES.....	73



## LISTE DES FIGURES

Figure 1.1 : Architecture de la transmission VoIP.....	6
Figure 1.2 : Poste PC to PC.....	9
Figure 1.3 : PC to Phone.....	9
Figure 1.4 : Phone to Phone.....	10
Figure 1.5 : Phone to Phone "Boîtier".....	10
Figure 1.6 : Établissement d'une session SIP via un proxy et un registrar.....	11
Figure 1.7 : Principe du protocole SIP.....	12
Figure 1.8 : Pile de protocoles H.323.....	13
Figure 1.9 : Les composants de l'architecture H.323.....	14
Figure 2.1 : Hyperviseur de type 1.....	19
Figure 2.2 : Hyperviseur de type 2.....	20
Figure 2.3 : Virtualisation complète.....	20
Figure 2.4 : Para-virtualisation.....	21
Figure 2.5 : Virtualisation assistée par extensions matérielles.....	21
Figure 2.6 : Les différents niveaux de services du Cloud.....	22
Figure 4.1 : Architecture générale.....	39
Figure 4.2 : Maquette de test.....	41
Figure 4.3 : Création de compte et accès à AWS.....	42
Figure 4.4 : Console d'Amazon Web Service (AWS).....	43
Figure 4.5 : Sécurité, identité et conformité de la console AWS.....	44
Figure 4.6 : Choisir une région AWS pour un VPC.....	44
Figure 4.7 : Création d'un VPC.....	45
Figure 4.8 : Table de routage principale du VPC.....	47

Figure 4.9 : Table de routage publique du VPC.....	48
Figure 4.10 : Liste de contrôle d'accès du VPC.....	48
Figure 4.11 : Groupe de sécurité du VPC.....	49
Figure 4.12 : Création des sous-réseaux du VPC.....	49
Figure 4.13 : Création du sous-réseau (10.0.1.0-us-east-1a) du VPC.....	50
Figure 4.14 : Création du sous-réseau (10.0.2.0-us-east-1b) du VPC.....	50
Figure 4.15 : Création du sous-réseau VPC.....	52
Figure 4.16 : Configuration du sous-réseau public.....	52
Figure 4.17 : Association explicite du sous-réseau public à la table de routage.....	53
Figure 4.18 : Association explicite du sous-réseau public à la table de routage publique.....	53
Figure 4.19 : Passerelle Internet.....	54
Figure 4.20 : Passerelle Internet.....	54
Figure 4.21 : Configuration de la passerelle Internet.....	55
Figure 4.22 : Configuration de la passerelle Internet.....	55
Figure 4.23 : Instance EC2 dans le sous-réseau public.....	56
Figure 4.24 : Instance EC2 dans le sous-réseau public.....	57
Figure 4.25 : Instance EC2 dans le sous-réseau public.....	57
Figure 4.26 : Instance EC2 dans le sous-réseau public.....	58
Figure 4.27 : Instance EC2 dans le sous-réseau privé.....	58
Figure 4.28 : Installation d'Asterisk Ubuntu dans AWS.....	58
Figure 4.29 : Configuration du serveur Asterisk dans AWS.....	59

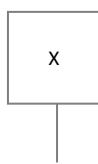
## **LISTE DES TABLEAUX**

Tableau 3.1 : Tableau récapitulatif des différents PAB-X.....29

Tableau 3.2 : Tableau récapitulatif des différents Cloud.....34

## GLOSSAIRE

- **VoIP** : Voice over IP
- **SIP** : Session Initiation Protocol
- **IAX** : Inter-Asterisk eXchange
- **QoS** : Quality of Service
- **RTP** : Real-time Transport Protocol
- **RTCP** : Real-time Transport Control Protocol
- **PABX** : Private Automatic Branch Exchange
- **IP-PBX** : Internet Protocol Private Branch Exchange
- **AWS** : Amazon Web Services
- **IaaS** : Infrastructure as a Service
- **PaaS** : Platform as a Service
- **SaaS** : Software as a Service
- **EC2** : Elastic Compute Cloud
- **S3** : Simple Storage Service
- **VPC** : Virtual Private Cloud
- **VM** : Virtual Machine
- **HA** : High Availability (Haute Disponibilité)
- **CLI** : Command Line Interface
- **NAT** : Network Address Translation
- **SMTP** : Simple Mail Transfer Protocol
- **SSH** : Secure Shell
- **ITU** : International Telecommunication Union
- **SDP** : Session Description Protocol
- **RTC** : Réseau Téléphonique Comité
- **RTCP** : Réseau Téléphonique Comité Public
- **ISDN** : Integrated Services Digital Network
- **PSTN** : Public Switched Telephone Network
- **RSVP** : Resource Reservation Protocol).
- **MCU** : Multipoint Control Unit
- **RNIS** : Réseau Numérique à Intégration de Services





## **INTRODUCTION GENERALE**

Ce projet vise à concevoir une plateforme VoIP innovante basée sur le Cloud, dans le but de moderniser la communication d'entreprise. Dans un monde en pleine transformation numérique et face à la mondialisation des échanges, les entreprises doivent répondre à des besoins de communication toujours plus complexes : collaboration à distance, gestion d'équipes géographiquement dispersées, interactions internationales... Dans ce contexte, la rapidité, la fiabilité et la flexibilité sont devenues des exigences incontournables. Pourtant, les solutions classiques de communication, souvent coûteuses et rigides, ne répondent plus efficacement à ces besoins.

La VoIP (Voice over IP) s'impose dès lors comme une alternative puissante, permettant l'unification de la voix, de la vidéo et des données à moindre coût. En parallèle, le développement du Cloud Computing a révolutionné la manière de concevoir, déployer et consommer les services informatiques, en offrant évolutivité, accessibilité et réduction des charges matérielles. La convergence de ces deux technologies ouvre donc la voie à de nouvelles opportunités pour les entreprises, en particulier à travers la mise en place de plateformes VoIP hébergées dans le cloud.

Cependant, malgré ce potentiel, plusieurs limites freinent encore l'adoption efficace de telles solutions. En effet, l'infrastructure traditionnelle repose souvent sur une architecture rigide : chaque service est hébergé sur un serveur dédié, ou dans les petites structures, tous les services sont regroupés sur un seul serveur. Cette organisation peut entraîner une surcharge importante en cas de forte sollicitation. De plus, lorsqu'aucun mécanisme de répartition intelligente de la charge n'est en place entre plusieurs serveurs, des déséquilibres apparaissent, dégradant la qualité de service.

Par ailleurs, l'accès à la téléphonie IP est généralement restreint à l'intérieur de l'entreprise, rendant les communications externes plus complexes. En cas de défaillance du système principal, la reprise automatique des services n'est pas assurée, exposant l'entreprise à des pertes importantes. Ces défis révèlent une problématique centrale : comment concevoir une architecture VoIP à la fois souple, scalable et résiliente, capable de s'adapter aux nouvelles exigences métiers, tout en assurant une continuité de service optimale ?

C'est dans cette optique que ce travail propose la conception et le déploiement d'une plateforme VoIP open source hébergée dans un environnement cloud, répondant aux besoins croissants en mobilité, performance et haute disponibilité. L'objectif est d'élaborer une

architecture évolutive, facilitant le déploiement de la VoIP, accessible depuis n'importe où, et dotée d'une gestion intelligente des ressources, avec répartition de charge, tolérance aux pannes et continuité de service.

Pour atteindre cet objectif, une revue approfondie de l'état de l'art sera d'abord menée afin de recenser les meilleures pratiques en matière d'intégration VoIP/Cloud. Ensuite, une étude comparative des technologies cloud existantes permettra d'identifier la solution la plus adaptée à cette intégration. Enfin, des expérimentations pratiques et modélisations viendront valider les choix techniques, avec des tests de performance concrets. Cette approche vise à démontrer les bénéfices techniques et économiques d'une migration vers une VoIP cloud, en réduisant les coûts d'infrastructure et de maintenance, tout en améliorant la productivité et la qualité des services de communication d'entreprise.

**CHAPITRE I :**  
**GENERALITES SUR LA VoIP**

Le terme VoIP (Voix sur IP, ou Voice Over Internet Protocol) est utilisé dans un sens large pour décrire l'ensemble des solutions permettant la transmission de la voix sur un réseau IP. Cependant, il est possible d'utiliser à d'autres termes ou expressions pour spécifier cette notion, par exemple :

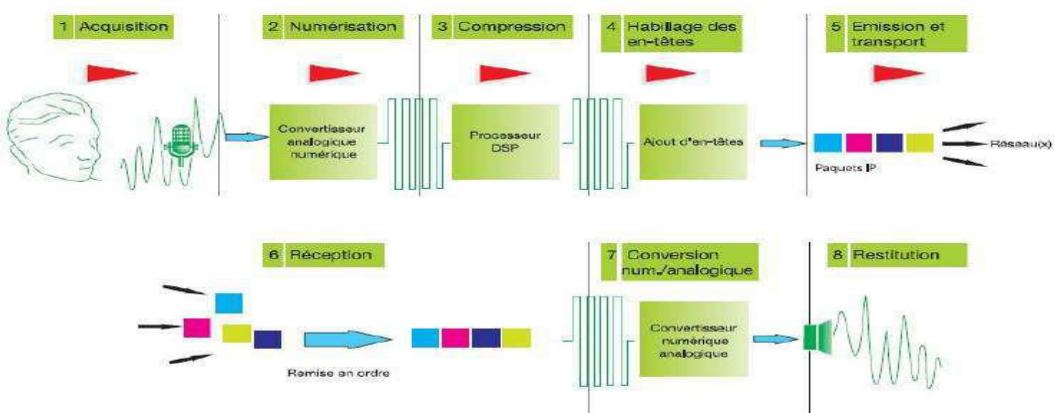
- ☛ **Téléphonie sur IP(ToIP)** : Cette expression met davantage l'accent sur l'utilisation de la technologie IP pour la téléphonie.
- ☛ **Communication vocale via Internet** : Cette formulation insiste sur l'utilisation d'Internet comme moyen de communication vocale.
- ☛ **Transmission de voix numérique sur réseau IP** : Cette description détaille le processus technique sous-jacent à la VoIP.

Les distinctions sont si notables que nous retenons cependant qu'il y a trois grandes catégories de communication vocale sur IP.

### 1.1. PRINCIPE DE FONCTIONNEMENT DE LA VoIP

La Voix sur IP consiste à numériser la parole en transformant un signal analogique en numérique, puis à le compresser à l'aide de codecs pour réduire le volume de données. Le signal est ensuite segmenté en paquets avec des en-têtes réseau (IP, UDP, RTP) avant d'être acheminé. À destination, les paquets sont réassemblés, décompressés et reconvertis en signal analogique pour restituer le message d'origine.

La structure de transfert pour la VoIP (Voix sur IP) revêt une importance cruciale dans l'assurance de la qualité et de la fiabilité des conversations vocales transmises par le biais d'Internet. Voici une explication globale de l'organisation de la transmission pour la VoIP [2] :



*Figure 1.1 Architecture de la transmission VoIP*

### **1.1.1. Acquisition du signal**

La voix et la vidéo sont des signaux continus, c'est-à-dire analogiques, qui doivent être convertis en signaux numériques avant d'être transmises. La première étape du processus implique la capture de ces signaux à l'aide d'un microphone, que ce soit celui d'un téléphone ou celui intégré à un ordinateur [2].

### **1.1.2. La numérisation**

Le signal capté à l'aide d'un microphone est échantillonné à intervalles de temps réguliers, puis converti en une séquence de chiffres binaires à chaque échantillon, c'est la quantification. Une grande quantité d'échantillons, réalisés avec un nombre élevé de bits, permet d'améliorer la qualité du signal. En règle générale, la voix est encodée à un débit de 64 Kbit/s, ce qui équivaut à un échantillonnage à 8 kHz avec chaque échantillon composé de 8 bits [2].

### **1.2.3. La compression**

Après l'échantillonnage et la quantification du signal, il est ensuite soumis à un processus de compression visant à réduire la quantité d'informations contenues dans le signal. Pour ce faire, diverses normes de compression et de décompression, également appelées codecs, sont utilisées pour le traitement de la voix. L'objectif principal de cette compression est de diminuer la bande passante nécessaire pour la transmission [2].

### **1.2.4. Habillage des en-têtes**

Les données numériques générées par le processeur de signal (DSP) ont nécessité une étape d'enrichissement en informations avant d'être converties en paquets de données destinés à être transmis sur le réseau. Cette opération s'effectue au sein de trois couches superposées, dont la couche IP qui se charge de l'assemblage des données en paquets. Chaque paquet débute par un en-tête qui indique le type de trafic utilisé, dans ce cas, le trafic UDP [2].

#### **❖ Le protocole IP**

Internet Protocol (IP) est un protocole, ou un ensemble de règles, appliqué au routage et à l'adressage des paquets de données afin qu'ils puissent traverser les réseaux et arriver à la bonne destination [4].

### ❖ Le protocole RTP

RTP (Real-time Transport Protocol), standardisé en 1996 par l'IETF, est un protocole de niveau application conçu pour le transport en temps réel des flux audio et vidéo sur les réseaux IP, généralement au-dessus d'UDP pour minimiser la latence. Il assure la gestion du temps réel et l'administration des sessions multipoints, tout en garantissant une certaine qualité de service (QoS).

Son rôle principal est d'organiser et de contrôler les paquets afin de reconstituer les flux avec leurs caractéristiques d'origine. Il assure le séquencement des paquets pour détecter les pertes, identifier le contenu des données pour sécuriser le transport et synchroniser les flux, et permet l'identification de la source, essentielle dans les communications multicast. Enfin, il facilite le transport des applications audio et vidéo en intégrant les trames numérisées dans des paquets adaptés à la transmission et au décodage [3].

### ❖ Le protocole RTCP

RTCP (Real-time Transport Control Protocol) complète RTP en envoyant périodiquement des paquets de contrôle aux participants d'une session. Il permet aux récepteurs de transmettre aux émetteurs des rapports sur la QoS, y compris le nombre de paquets perdus, la gigue et le délai aller-retour. Ces informations fournissent la source pour ajuster la transmission, par exemple en modifiant le niveau de compression, afin de maintenir une qualité de service optimale. RTCP utilise UDP pour le multiplexage avec RTP, facilitant ainsi la gestion et l'optimisation des flux multimédias en temps réel [4].

#### **1.2.5. Emission et transport**

Les paquets sont transportés du point d'émission au point de réception sans qu'un itinéraire spécifique soit réservé pour leur transfert. Ils circulent sur le réseau (qu'il s'agisse d'un réseau local, étendu, voire Internet) en utilisant les ressources disponibles, et ils parviennent à leur destination sans garantie d'arriver dans un ordre particulier [2].

#### **1.2.6. Réception**

Lorsque les paquets atteignent leur destination, il est crucial de les réorganiser correctement et de manière rapide. Sinon, une dégradation de la qualité de la voix sera perceptible [2].

#### **1.2.7. Conversation numérique analogique**

Après la transmission, le signal numérique compressé est reconvertis en analogique grâce à un convertisseur numérique-analogique (CNA). Ce processus quantifie les données numériques

pour leur attribution des valeurs analogiques correspondantes, puis reconstitue une onde continue. Une fois restauré, le signal peut être diffusé par un haut-parleur [2].

#### 1.2.8. La restitution du signal

Le haut-parleur reçoit ce signal analogique et le transforme en vibrations mécaniques, créant ainsi des ondes sonores qui peuvent être entendues. Cela permet la restitution du signal d'origine, que ce soit de la musique, de la voix ou d'autres types de sons, pour une expérience auditive complète et fidèle.

### 1.3. MODES D'ACCES ET ARCHITECTURE

Dans un réseau de VoIP, l'architecture d'une solution de VoIP est un élément clé, aussi bien en interne qu'en termes d'accès multiples. Elle permet d'utiliser la VoIP de différentes manières : de téléphone à téléphone, de PC à PC ou encore de PC à téléphone.

Cependant, avant toute mise en place, la connaissance des équipements tels que le PABX IP, le routeur et le switch est cruciale, car ils assurent la communication au sein de l'architecture.

Selon le type de terminal utilisé, on distingue trois scénarios possibles de téléphonie sur IP [4] [5].



**Entre deux postes ordinateurs (pc to pc) :** Dans ce scénario le but sera de transformer son ordinateur en un poste téléphonique en lui ajoutant une carte son full-duplex pour garantir une conversation simultanée, un micro et un logiciel de voix sur IP compatible. Le correspondant quant à lui, doit disposer des mêmes outils et surtout du même logiciel de téléphonie. A cet instant, le poste numérique, compresse et encapsule les échantillons de voix dans des paquets IP avant de les envoyer sur Internet. L'accès se fait via un fournisseur d'accès à internet IAP/ISP [4].

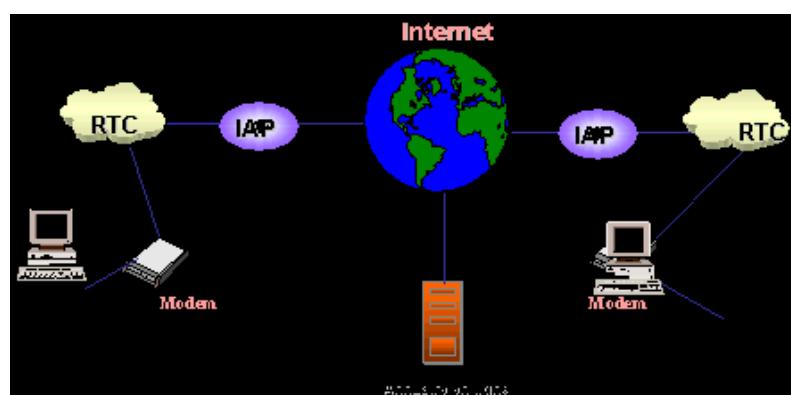


Figure 1.2 : poste pc to pc



**Entre pc et un poste téléphonique :** Ce sont donc à la fois le réseau Internet et le réseau téléphonique commuté qui sont utilisés dans ce mode de communication. Le service n'est plus gratuit puisque le réseau RTC est généralement facturé à l'usage et non forfaitairement. Grâce à ces crédits, les utilisateurs peuvent communiquer partout dans le monde, à des tarifs très avantageux, une bonne partie de la communication transitant sur le réseau IP, y compris la partie qui relie l'abonné appelant à son opérateur [4].

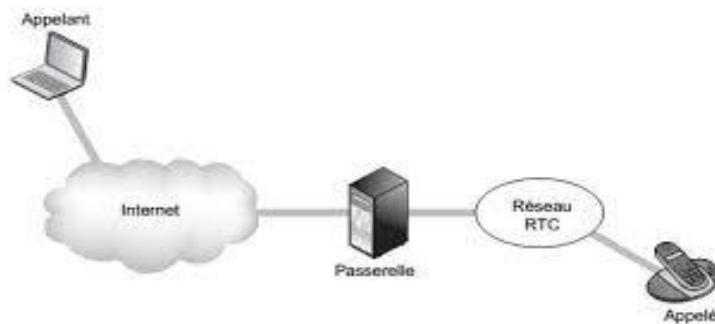


Figure 1.3 : pc to phone



**Entre deux postes téléphoniques :** Dans ce cas l'appelant et l'appelé sont tous les deux des abonnées du réseau téléphonique commuté public (RTCP) et utilisent de manière classique leur appareil téléphonique pour la communication vocal.



**En utilisant des passerelles** Dans ce cas, les passerelles ainsi que le réseau IP géré pourraient appartenir à des acteurs différents selon qu'il s'agit: D'un usage purement interne de la voix sur IP au sein du réseau d'un opérateur téléphonique unique (usagers A et B ainsi gérés). De la fourniture d'un service de voix longue distance par un opérateur longue distance utilisant la technologie de la voix sur IP (les usagers A et B appartenant alors à des réseaux distincts). [4]

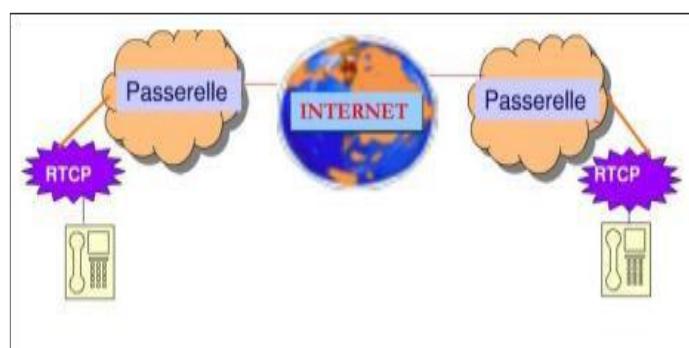
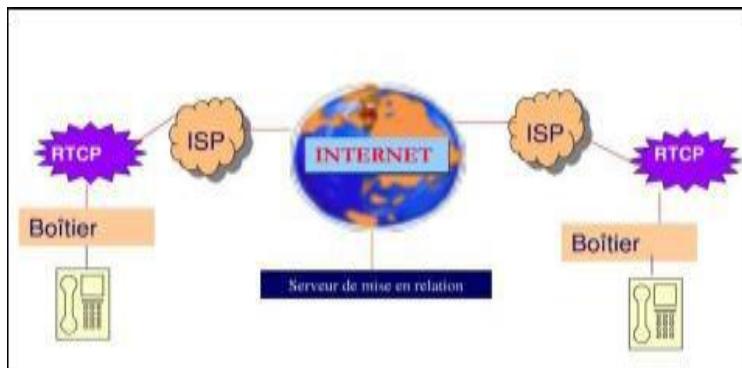


Figure 1.4: Phone to phone

- En utilisant des boîtiers d'adaptation : Pour faire bénéficier de ce service, un certain nombre de sociétés commercialisent des boîtiers ressemblant à des modems et qui s'interpose entre le poste téléphonique de l'usager et son branchement au réseau téléphonique public commuté. [4]



*Figure 1.5: Phone to phone «boitier»*

#### 1.4. LES PROTOCOLES DE SIGNALISATION

Les protocoles de signalisation quant à eux, transportent l'information sur un réseau IP. Ce type de protocoles est spécifique à la voix sur IP et aux applications nécessitant le transit de l'information en temps réel comme par exemple, la vidéo conférence [11].

##### 1.4.1. SIP (Protocole d'Initiation de Session)

SIP (Session Initiation Protocol), standardisé par l'IETF (RFC 3261), est utilisé pour établir, modifier et terminer des sessions multimédias. Il gère l'authentification, la localisation des participants et la négociation des types de médias via SDP. Indépendant du transport des données, il est souvent associé à RTP pour les sessions audio et vidéo. Standard ouvert et interopérable, SIP remplace progressivement H.323 et s'étend à diverses applications comme la visiophonie, la messagerie instantanée et les jeux en ligne [9].

##### ❖ Rôle des composants

Pour fonctionner, SIP se base sur une architecture comportant trois principaux acteurs : Les User-Agents, les Registrars et les Proxys.

- User-Agents : L'User-Agent, côté utilisateur, est un dispositif logiciel ou matériel gérant le protocole SIP sur un téléphone, un logiciel VoIP ou un smartphone. Il peut agir comme client ou serveur selon qu'il initie ou reçoit un appel. S'il connaît l'IP de son correspondant, il le contacte directement ; sinon, il passe par un registraire. Il est également appelé terminal [10].

**☞ Registrars :** Les registrars sont des annuaires de la VoIP où un User-Agent s'inscrit pour être répertorié selon son URI (par exemple, sip :nom .utilisateur@domaine.fr ). Cela permet de contacter un User-Agent via son URI SIP. Les contacts sont enregistrés dans une base de données et mis à jour régulièrement pour refléter les changements d'adresses IP. C'est le client qui informe le registraire de tout changement d'adresse via une requête "register". [10]

**☞ Proxys** Les proxys transmettent les demandes d'appels entre les User-Agents. Lorsqu'un proxy reçoit une demande, il consulte un registrar pour obtenir l'adresse IP associée à l'URI et la renvoyer au terminal demandeur. Ensuite, l'User-Agent contacte directement le destinataire sans passer par le proxy.

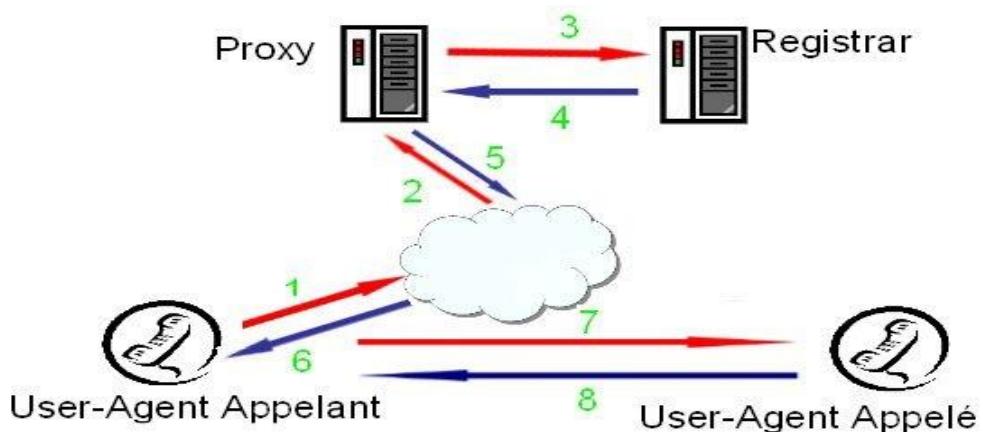


Figure 1.6 : Etablissement d'une session SIP via un proxy et un registra

#### ❖ Principe du protocole SIP

Le client peut envoyer au format ASCII les commandes suivantes à un serveur :

**Invite** : Le client demande au serveur une nouvelle session.

**Ack** : Une fois la session établie, le client confirme la connexion.

**Cancel** : Le client annule la session avant même la confirmation de l'invitation.

**Bye** : Le client termine la session établie.

**De la même manière, le serveur peut répondre des codes au client:**

**100** : trying

**200** : Ok

## 404 : Not Found (Erreur)

Ces codes sont les mêmes que le protocole HTTP. Il en existe cependant des propres à SIP comme : 180 - Sonnerie ou 486 pour Occupé. [10]

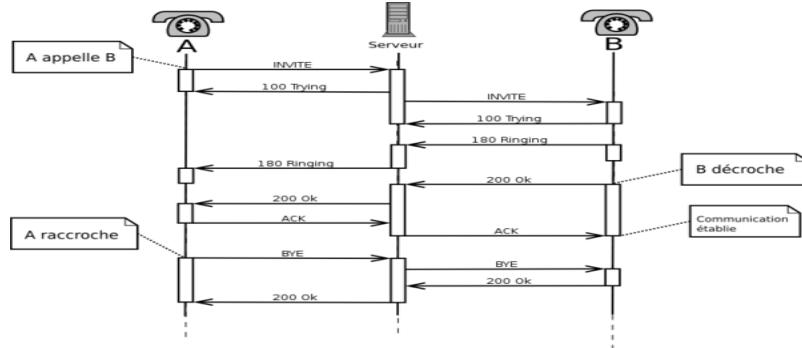


Figure 1.7: Principe du protocole SIP

### 1.4.2. H.323

H.323 est un ensemble de protocoles de signalisation plus anciens, développés par l'ITU, qui couvrent une large gamme de médias, y compris la voix et la vidéo. Il inclut des protocoles pour la signalisation, la négociation et le transport. Faisant partie de la série H.32x, H.323 traite de la vidéoconférence sur divers réseaux, et intègre H.320 et H.324 pour les réseaux ISDN et PSTN [4].

Plus qu'un protocole, H.323 crée une association de plusieurs protocoles différents et qui peuvent être regroupés en trois catégories : la signalisation, la négociation de codec, et le transport de l'information [3].

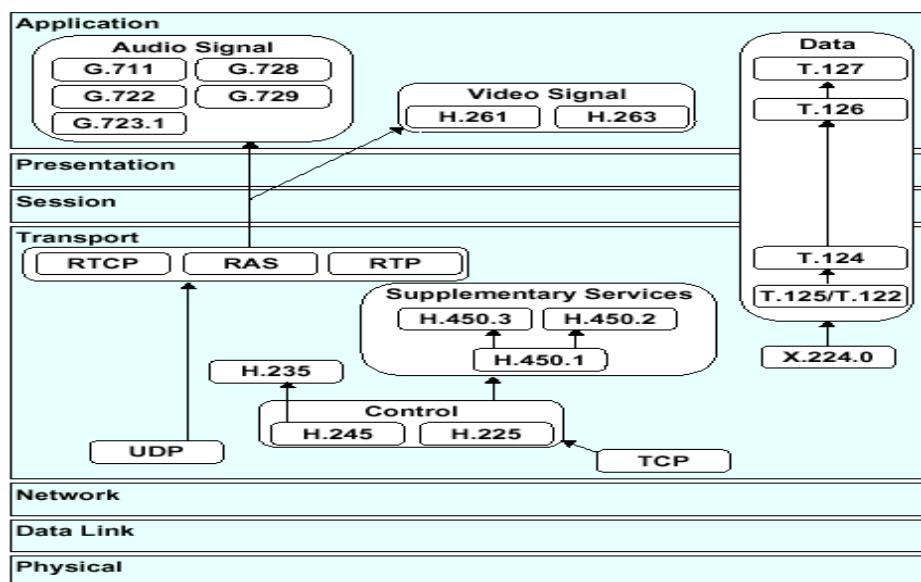
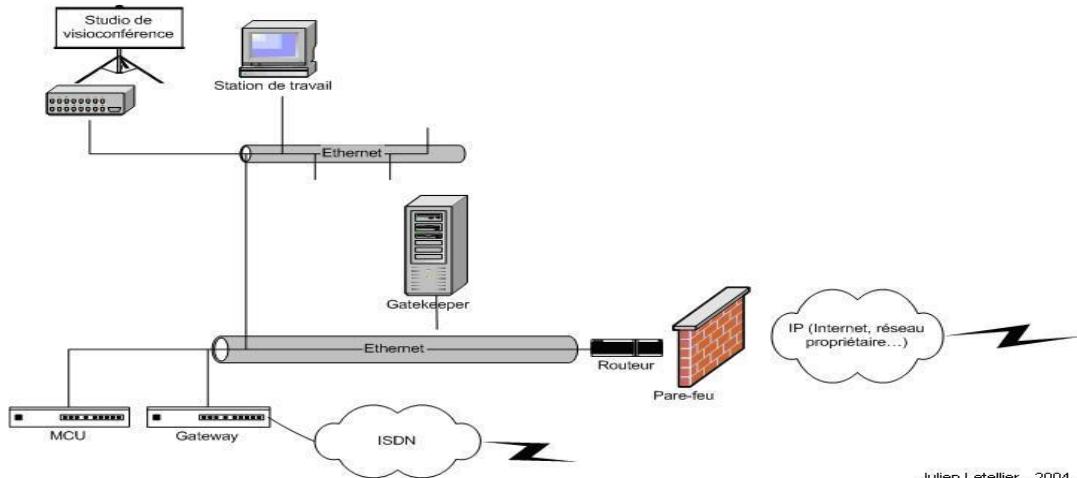


Figure 1.8 : Pile de protocoles H.323

## ❖ Rôle des composants

L'infrastructure H.323 repose sur quatre composants principaux : les terminaux, les Gateways, les Gatekeepers, et les MCU (Multipoint Control Units).



Julien Letellier - 2004

Figure 1.9: Les composants de l'architecture H.323

☞ **Les terminaux H.323 :** Le terminal H.323 peut être un ordinateur, un téléphone, un terminal de vidéoconférence sur Internet. Il doit supporter le codec G.711, utiliser H.245 pour la négociation des canaux et Q.931 pour la signalisation des appels. [3]

☞ **Gateway ou les passerelles :** Les passerelles H.323 permettent l'interconnexion entre les réseaux VoIP et les réseaux classiques (RTC, RNIS, etc.). Elles assurent la conversion des protocoles de signalisation (ex : Q.931), l'adaptation des signaux de contrôle et la gestion des médias (multiplexage, correspondance des débits, transcodage audio).

☞ **Gatekeeper ou les portiers :** Le Gatekeeper est le point d'entrée du réseau H.323. Il gère l'enregistrement des terminaux et passerelles, traduit les alias en adresses IP, contrôle l'accès, gère la bande passante et limite les visioconférences pour préserver les ressources réseau.

☞ **Les MCU :** Les contrôleurs multipoint (MCU) permettent la visioconférence entre plusieurs terminaux en « présence continue » ou « activation à la voix ». Une MCU se compose d'un contrôleur multipoint gérant les négociations H.245 et les ressources, et d'un ou plusieurs processeurs multipoints traitant les flux audio, vidéo et données. [3]

### 1.4.3. IXA (Inter Asterisk eXchange)

L'Inter-Asterisk eXchange (IAX) est un protocole de VoIP conçu spécifiquement pour Asterisk, un logiciel PBX (Private Branch Exchange). IAX est utilisé pour connecter des

serveurs et des terminaux VoIP et est réputé pour son efficacité en termes de bande passante et de gestion des NAT (Network Address Translation).

 **La séquence de messages d'enregistrement** : L'enregistrement permet à un client IAX de s'enregistrer et de fournir son adresse IP et ses informations d'authentification à l'appelé. [13]

 **La signalisation des terminaux IAX** : Le protocole IAX peut être utilisé pour mettre en place une liaison entre deux clients IAX avant d'établir la communication. Cette procédure appelée « Call legs » se compose de deux messages principaux :

- ⊕ Message **New** : contient les informations de destination. La réponse peut être une demande d'authentification, rejet de demande ou « Call legs » est établi.
- ⊕ Message **Accept** : indique que l'établissement d'appel au niveau « Call legs » s'est bien déroulé.

Call legs permet de relier les deux entités communicantes et faire passer les messages de contrôle d'appel jusqu'à la fin d'appel [13].

Les protocoles de signalisation SIP, H.323 et IAX assurent l'établissement, la gestion et la fin des appels en VoIP. SIP est le plus utilisé pour sa flexibilité, H.323 reste présent dans les systèmes traditionnels, et IAX se démarque par sa bonne gestion de la bande passante et sa facilité à franchir les pare-feu.

### 1.5. CODECS UTILISES

Pour que la VoIP fonctionne efficacement, plusieurs codecs (codeurs-décodeurs) sont utilisés pour compresser et décompresser les données audio et video. Les codecs sont essentiels pour optimiser la qualité de la voix, la bande passante utilisée et la latence dans les communications VoIP. [12]

Ce pendant Pour la VoIP, plusieurs codecs peuvent servir. Voici leurs détails :

#### ❖ Codecs audio

 **G.711** : Ce codec, premier utilisé en VoIP, reste implémenté pour assurer la compatibilité entre les équipements. Il utilise un codage logarithmique avec deux variantes : la loi  $\mu$  (USA, Japon) et la loi A (reste du monde), presque identiques. Il génère un flux de 64 kbit/s et offre une bonne qualité audio avec un score MOS de 4,2, bien que les fréquences au-delà de 4 kHz soient éliminées.

 **G.722** : A la différence du G.711, ce codec transforme le spectre jusqu'à 7kHz ce qui restitue encore mieux la voix. Les débits que ce codec fournit sont 48,56 ou 64kbit/s. [12]

Une des particularités est de pouvoir immédiatement changer de débit. Ceci est fortement appréciable lorsque la qualité du support de transmission se dégrade.

 **G.722.1** : Dérivé du codec précédent, celui-ci propose des débits encore plus faibles (32 ou 24kbit/s). Il existe même des versions propriétaires de ce codec fournissant un débit de 16kbit/s.

 **G.723.1** : C'est le codec par défaut lors des communications à faible débit. Deux modes sont disponibles. Le premier propose un débit de 6,4kbit/s et le deuxième un débit de 5,3kbit/s. Là aussi, le changement de mode peut se faire en pleine communication.

 **G.729** : Ce codec est avec G.723 un des plus utilisés en VoIP. A l'instar de ce dernier, il ne convient pas pour des transmissions autres que la voix et ne retransmet pas correctement les tonalités DTMF. Le score MOS obtenu est 4.0 [12].

#### ❖ **Codecs vidéo**

 **H.263** : Développé pour la visioconférence et les communications sur réseaux à faible débit, ce codec est une évolution du H.261, avec une meilleure compression et une meilleure qualité d'image pour un même débit. Il est principalement utilisé dans les anciennes applications de VoIP et de visioconférence.

 **H.264 (AVC - Advanced Video Coding)** : Ce codec est largement utilisé pour la compression vidéo haute efficacité. Il offre une bonne qualité d'image avec des débits relativement faibles, ce qui le rend idéal pour la VoIP, la diffusion en streaming et la vidéoconférence. Il est compatible avec une large gamme d'appareils et de réseaux.

 **H.265 (HEVC - High Efficiency Video Coding)** : Successeur du H.264, il améliore considérablement l'efficacité de la compression, permettant une meilleure qualité d'image avec une réduction du débit allant jusqu'à 50 %. Il est adapté aux flux vidéo de haut définition, notamment le 4K et le 8K, bien que sa complexité de traitement soit plus élevée.

 **VP8** : Développé par Google, ce codec open source est conçu pour la diffusion vidéo sur Internet et est particulièrement utilisé dans les applications WebRTC.

 **VP9** : Évolution du VP8, ce codec améliore significativement l'efficacité de la compression, rivalisant avec le H.265.

Les codecs optimisent la qualité et l'efficacité des communications VoIP en compressant la voix pour une meilleure transmission. Leur choix impacte la bande passante, la latence et la clarté audio, garantissant ainsi une communication fluide et adaptée aux besoins du réseau.

## 1.6. CONTRAINTES LIEES A LA VOIP

La VoIP est une technologie relativement récente qui évolue rapidement et comme toute nouvelle technologie, elle révolutionne le secteur de la téléphonie avec des atouts indéniables face au réseau analogique.

- ☞ **Taux de perte de paquets** : Le taux de perte de paquets correspond au pourcentage de paquets RTP perdus en cours de transmission. En VoIP, toute perte au-delà de 1 % entraîne des « trous » dans l'audio, des coupures ou un effet « haché », et dès 3–5 % la conversation devient difficilement intelligible, car les paquets manquants ne peuvent être récupérés en temps réel sans perturber le flux vocal [54].
- ☞ **Gigue (jitter)** : La gigue mesure la variation du délai d'arrivée des paquets (en ms). Lorsqu'elle dépasse environ 30 ms, le buffer doit être accru pour lisser ces écarts, ce qui ajoute de la latence et peut lui-même provoquer des pertes supplémentaires si le buffer déborde, détériorant ainsi la continuité et la fluidité de la voix [55].
- ☞ **Latence** : La latence one-way désigne le délai entre l'émission de la voix et sa réception. Pour que la conversation reste fluide, elle doit rester inférieure à 150 ms ( $\leq 300$  ms aller-retour) ; au-delà, on perçoit un écho ou un décalage, ce qui gêne l'interaction et peut provoquer des interruptions involontaires de la parole [56].
- ☞ **Bandé passante** : La bande passante requise dépend du codec : G.711 consomme environ 80–100 kb/s par appel, G.729 autour de 24–32 kb/s et Opus de 6 à 42 kb/s selon le profil. Si le débit alloué est insuffisant, les paquets sont retardés ou abandonnés, générant latence, perte de paquets et dégradation de la qualité audio [57].

La VoIP permet de transmettre la voix via les réseaux IP grâce à une série d'étapes techniques (numérisation, compression, transport, etc.) et à des protocoles comme SIP ou H.323. Malgré quelques contraintes (latence, sécurité), elle offre une solution de communication moderne, économique et efficace.



## **CHAPITRE II :**

### **LE CLOUD COMPUTING**

Le cloud computing offre un accès à la demande, via Internet, à des ressources informatiques hébergées par un fournisseur de services cloud. Des entreprises comme AWS, Google Cloud ou Azure proposent ces services sous forme de paiement à l'usage ou d'abonnement. Cela permet aux entreprises de stocker et gérer leurs données sur des serveurs distants, réduisant ainsi les coûts d'infrastructure sur site [14].

## 2.1. CONCEPTS DE LA VIRTUALISATION

La virtualisation permet de créer plusieurs environnements simulés à partir d'un seul système physique grâce à un hyperviseur. Celui-ci divise et alloue les ressources matérielles aux machines virtuelles (VM), optimisant ainsi l'utilisation du matériel et prolongeant la valeur des investissements existants [1].

 **Hyperviseur:** L'hyperviseur est un composant logiciel qui gère plusieurs machines virtuelles sur un ordinateur. Il garantit que chaque machine virtuelle reçoit les ressources allouées et n'interfère pas avec le fonctionnement des autres machines virtuelles. Il existe deux types d'hyperviseurs. [15]

 **Hyperviseurs de type 1 :** Un hyperviseur de type 1, ou hyperviseur de matériel nu, est un programme d'hyperviseur installé directement sur le matériel plutôt que sur le système d'exploitation. Par conséquent, les hyperviseurs de type 1 offrent de meilleures performances et sont couramment utilisés par les applications d'entreprise. [15]

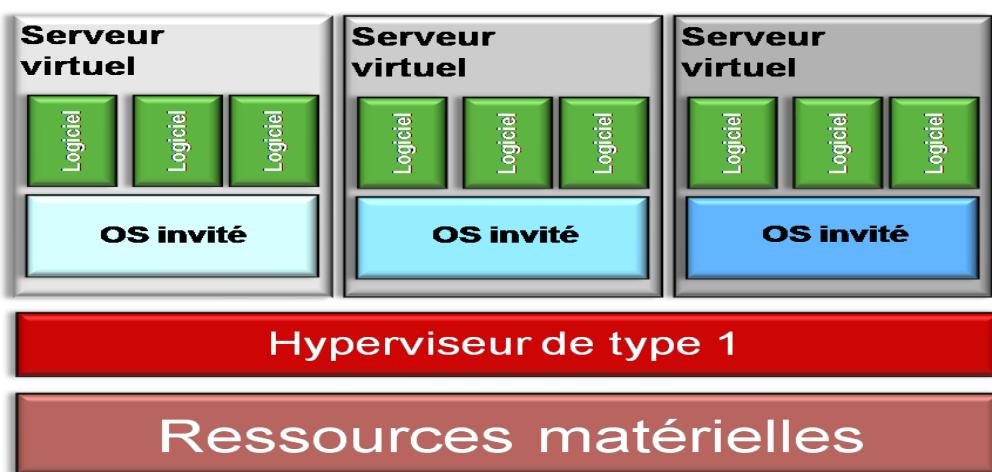
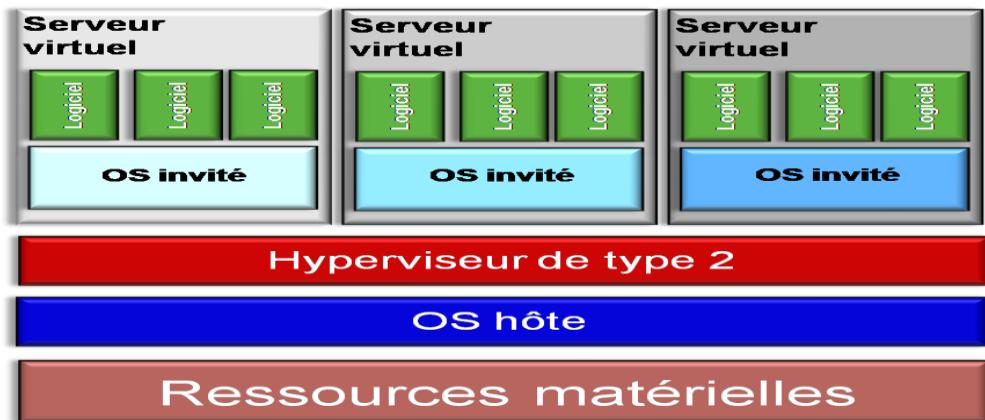


Figure 2 .1 : hyperviseur de type 1

 **Hyperviseurs de type 2 :** Également appelé hyperviseur hébergé, l'hyperviseur de type 2 est installé sur un système d'exploitation. Les hyperviseurs de type 2 sont adaptés à l'informatique pour l'utilisateur final. [15]

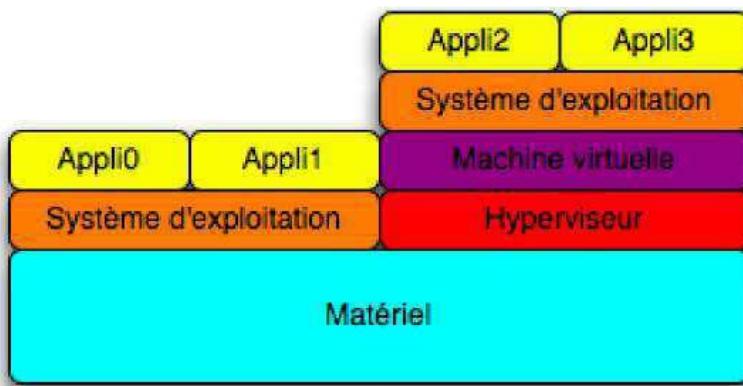


*Figure 2 .2 : hyperviseur type 2*

### 2.1.1. Virtualisation complète

L'émulation avec un hyperviseur installé au niveau du système d'exploitation hôte permet un accès plus rapide au matériel, car les applications du système invité n'ont qu'une seule couche à traverser. Cependant, cette méthode est plus restrictive, car le processeur de la machine virtuelle doit être compatible avec celui de l'hôte (Voir la figure 2.3 ci-après). [16]

☞ **Utilisations :** faire tourner des systèmes d'exploitation virtuellement en les laissant accéder aux différents périphériques au travers des pilotes installés sur le système hôte. Des logiciels tels que VMWare Workstation ou VirtualPC de Microsoft l'utilisent. [16]



*Figure2.3 : Virtualisation complète*

### 2.1.2. Para virtualisation

On fait tourner l'hyperviseur sur le matériel et les systèmes d'exploitation invités au-dessus de ce dernier (Voir. la figure 2.4 ci-après).

☞ **Avantages :** meilleures performances que la virtualisation totale.



**Inconvénients :** les systèmes d'exploitation invités doivent être modifiés afin de tourner avec l'hyperviseur. La machine virtuelle doit pouvoir tourner sur le processeur (physique) de la machine hôte. [16]

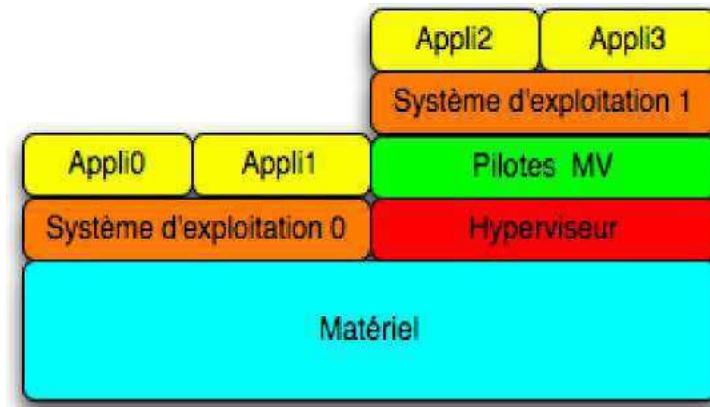


Figure 2.4 : Para virtualisation

### 2.1.3. Virtualisation assistée par extensions matérielles

Des instructions sont ajoutées au processeur pour qu'il serve d'hyperviseur à l'aide du « HAL » (Hardware Abstraction Layer). Les systèmes d'exploitation invités sont au même niveau que ceux hôtes (Voir la figure 2.5 ci-après).



**Avantages :** certains processeurs permettent un accès direct à la mémoire des invités. Les performances sont optimales. Les processeurs ne sont pas émulés et les systèmes d'exploitation inchangés.



**Inconvénient :** il faut un processeur spécifique qui supporte les nouvelles instructions (HAL).

Ainsi, si une machine ne comporte pas ce type de processeur, alors il est nécessaire de s'équiper d'une machine récente qui comportera un processeur compatible.

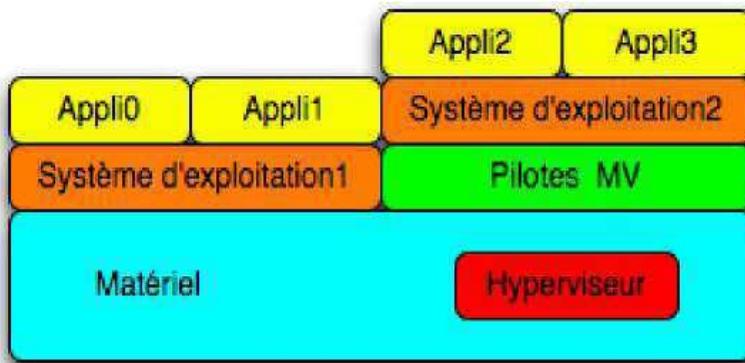


Figure 2.5 : Virtualisation assistée par extensions matérielle

## 2.2. PRESENTATION DU CLOUD COMPUTING

Le cloud computing permet d'accéder à des ressources informatiques (matériel, stockage, logiciels) via Internet, sans avoir à gérer des infrastructures physiques sur site. Il aide les entreprises à stocker et accéder aux données à distance, tout en étant géré par un fournisseur tiers. Le cloud offre des options de déploiement public, privé ou hybride, selon les besoins de flexibilité et de contrôle, permettant ainsi de réduire les coûts d'investissement et de gestion des systèmes. Les trois principaux types de modèles de services cloud sont les suivants : [21]

### 2.2.3. Les différents services du cloud computing

Il existe trois principaux types de service cloud : logiciel en tant que service (SaaS), plateforme en tant que service (PaaS) et infrastructure en tant que service (IaaS). Il n'existe pas d'approche unique du cloud, il s'agit plutôt de rechercher la solution adaptée à vos besoins.

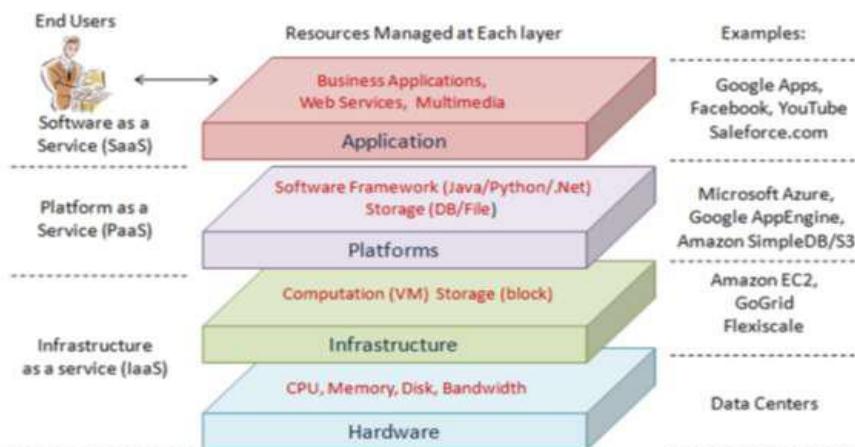


Figure 2.6 : Les différents niveaux services du cloud

#### ❖ SAAS (Software as a Service)

Le SaaS (Software as a Service) est un modèle où les applications sont hébergées par le fournisseur de cloud et accessibles via Internet. Les clients paient un abonnement à l'utilisation plutôt que de gérer leur propre infrastructure. Ce modèle permet aux entreprises d'accéder rapidement aux technologies innovantes, avec des mises à jour automatiques et une flexibilité pour adapter les services selon la croissance. Il offre une suite complète de logiciels pour divers besoins d'entreprise, comme la gestion de la relation client, les finances, les ressources humaines, la supply chain, etc [17].

### ❖ PAAS (Plateform as a Service)

Le PaaS (Platform as a Service) permet aux clients de développer et gérer des applications web et mobiles sans gérer l'infrastructure sous-jacente. Le fournisseur héberge l'infrastructure et le middleware, tandis que le client y accède via un navigateur. Le PaaS offre des outils prêts à l'emploi pour intégrer des fonctionnalités comme l'IA et l'IoT, et inclut des solutions pour l'analyse des données, la gestion de contenu, des bases de données, des systèmes et la sécurité [17].

### ❖ IAAS (Infrastructure as a Service)

IaaS permet aux clients d'accéder à des services d'infrastructure à la demande via Internet. Le principal avantage est que le fournisseur Cloud héberge les composants d'infrastructure fournissant la capacité de calcul, de stockage et de réseau, de sorte que les abonnés puissent exécuter leurs charges de travail dans le cloud. L'abonné au cloud est généralement responsable de l'installation, de la configuration, de la sécurisation et de la maintenance de tout logiciel sur les solutions cloud native, comme les bases de données, le middleware et les logiciels d'application [17].

#### 2.2.4. Les différentes solutions de cloud computing

Le cloud public, privé et hybride sont des modèles de déploiement du cloud qui absorbent comment les ressources informatiques sont hébergées et accessibles :

### ❖ Cloud Public

Le principe est d'héberger des applications, web en général, sur une plateforme où l'accès se fait uniquement par internet et où l'intérêt est la mutualisation optimale des ressources. L'environnement est de ce fait partagé avec un nombre virtuellement illimité d'utilisateurs sur la même plateforme.

Les avantages de ce type de cloud sont de pouvoir « louer » de la ressource à l'heure et de ne payer que celle utilisée. L'aspect négatif est un bridage très fort des possibilités puisqu'il est pratiquement impossible d'y héberger toute une infrastructure informatique [18].

### ❖ Le cloud privé

Le cloud privé est un environnement dédié à une seule entreprise, offrant un contrôle total sur l'infrastructure et la gestion. Il peut être hébergé sur site ou par un fournisseur de services cloud. Ce modèle permet à l'entreprise de personnaliser son environnement selon ses besoins

spécifiques, tout en bénéficiant d'un réseau exclusif et sécurisé. Le paiement est généralement basé sur un abonnement continu, avec des ressources virtuelles et physiques adaptées [19].

#### ❖ Cloud hybride

Le cloud hybride combine plusieurs environnements, généralement des clouds publics et privés, ainsi que des infrastructures sur site. Ces environnements sont étroitement interconnectés pour fonctionner comme une infrastructure unifiée. Les entreprises peuvent utiliser un cloud privé pour certains services et un cloud public pour d'autres, comme sauvegarde ou pour gérer les pics de demande, tout en maintenant la majorité des opérations dans leur cloud privé [20].

### 2.3. AVANTAGES ET INCONVENIENTS DU CLOUD

Le cloud computing offre des avantages majeurs comme la flexibilité, la réduction des coûts et l'accessibilité à distance. Toutefois, il comporte des inconvénients, notamment des préoccupations concernant la sécurité des données, la dépendance aux fournisseurs et les risques liés à la confidentialité.

#### 2.3.1. Avantages du cloud computing

👉 **Temps de production réduit :** Le cloud computing permet de créer ou supprimer des instances rapidement, accélérant ainsi le travail des développeurs. Il favorise l'innovation en facilitant l'expérimentation et la création d'applications sans les contraintes matérielles ni les processus d'approvisionnement lents. [21]

👉 **Évolutivité et flexibilité :** Le cloud computing offre une grande flexibilité en permettant aux entreprises d'ajuster rapidement leurs ressources et leur stockage selon les besoins, sans investir dans une infrastructure physique. Elles peuvent ainsi gérer les pics de charge sans coûts supplémentaires et réduire les ressources lorsque celles-ci ne sont plus nécessaires.

[21]

👉 **Économies :** Quel que soit le modèle de service cloud que vous choisissez, vous ne payez que pour les ressources que vous utilisez réellement. Vous évitez ainsi de surévaluer vos besoins et de super visionner votre centre de données, et vos équipes informatiques gagnent un temps précieux qui leur permet de se concentrer sur des tâches plus stratégiques. [21]

👉 **Collaboration plus efficace :** Le stockage dans le cloud vous permet de rendre des données disponibles partout et à tout moment. Au lieu d'être liées à un emplacement

ou un appareil spécifiques, les données sont accessibles aux utilisateurs du monde entier depuis n'importe quel appareil, à condition de disposer d'une connexion Internet. [21]

 **Sécurité avancée :** Malgré les idées reçues, le cloud computing peut en réalité renforcer votre stratégie de sécurité grâce à la profondeur et la couverture des fonctionnalités de sécurité, de la maintenance automatique et de la gestion centralisée qu'il intègre. Les fournisseurs cloud réputés emploient également des experts de premier plan dans le domaine de la sécurité, et font appel aux solutions les plus avancées, offrant ainsi une protection renforcée. [21]

 **Prévenir la perte de données :** Les fournisseurs cloud proposent des fonctionnalités de sauvegarde et de reprise après sinistre. Le stockage de données dans le cloud plutôt qu'en local peut contribuer à éviter toute perte de données dans les situations d'urgence, par exemple une défaillance matérielle, des menaces malveillantes ou même une simple erreur utilisateur. [21]

### **2.3.2. Inconvénients du cloud computing**

Le cloud computing présente des avantages, mais dépend d'une connexion Internet stable. Une mauvaise connexion ou des interruptions dues à des problèmes techniques ou des catastrophes naturelles peuvent empêcher l'accès aux informations ou applications, créant des périodes de panne jusqu'à la résolution du problème.

## **2.4. INTEROPERABILITE DANS LE CLOUD**

L'interopérabilité dans le cloud désigne la capacité des différents services, applications ou systèmes présents dans un environnement cloud à communiquer, échanger des données et fonctionner ensemble de manière fluide, même s'ils proviennent de sources ou de technologies différentes [45].

Elle est essentielle car elle permet de créer des solutions flexibles, évolutives et efficaces. Grâce à l'interopérabilité, les utilisateurs peuvent intégrer facilement divers outils, automatiser des processus, éviter la dépendance à un seul système, et améliorer la collaboration entre plusieurs applications ou équipes. Elle garantit également la continuité des services en cas de mise à jour ou de changement de composant [45].

Dans un même environnement cloud, l'interopérabilité se traduit par la manière dont différents services (comme le calcul, le stockage, les bases de données ou les outils d'analyse) s'intègrent naturellement les uns aux autres. Par exemple, une application peut stocker ses données dans

un service de base de données, déclencher des traitements automatiques via un service de fonction, et envoyer des notifications via un autre service, le tout en restant dans le même écosystème, grâce à des standards communs et des interfaces bien définies [45].

## 2.5. HAUTE DISPONIBILITE

La haute disponibilité est un concept clé dans les infrastructures informatiques. Elle vise à garantir qu'un système, service ou application est accessible et fonctionnel le plus possible, même en cas de défaillances.

### 2.5.3. Les caractéristiques de la haute disponibilité

Chaque composant d'un serveur (processeur, disque dur, mémoire, carte réseau) représente une ressource critique susceptible de provoquer une panne. Leur fiabilité dépend de leur conception et de leur entretien. Pour minimiser les pannes dans un système, il est essentiel de comprendre les notions suivantes :

- ☞ **MTBF (Mean Time Between Failures)** : Il représente le temps moyen entre deux pannes successives d'un système. Un MTBF élevé indique une meilleure fiabilité , car les pannes sont moins fréquentes.
- ☞ **MTTF (Mean Time To Failure)** : Ce terme correspond au temps moyen avant qu'un composant ou un système ne tombe en panne pour la première fois. Il s'applique généralement aux équipements ou composants non réparables. Un MTTF élevé reflète une durée de vie plus longue.
- ☞ **MTTR (Mean Time To Repair)** : Il désigne le temps moyen nécessaire pour réparer un système en cas de panne. Un MTTR faible est souhaitable, car il indique que les interruptions sont rapidement résolues, minimisant ainsi les impacts.

### 2.5.4. Techniques de bases de la haute disponibilité

Ces techniques sont souvent utilisées ensemble dans les environnements critiques pour garantir une disponibilité maximale, une tolérance aux pannes, et une continuité de service. En appliquant ces principes, il est possible de construire des systèmes robustes capables de fonctionner efficacement même en cas de défaillance partielle.

- ☞ **La redondance matérielle** : La redondance matérielle consiste à intégrer des composants en double ou en multiples dans l'infrastructure. Ainsi, si un composant tombe en panne, un autre peut immédiatement prendre le relais. Cette approche réduit le risque d'interruption de service causée par une défaillance matérielle [22].



**Mise en cluster :** La mise en cluster regroupe plusieurs serveurs (ou nœuds) qui fonctionnent ensemble pour offrir un service commun. Cette configuration garantit que si un nœud tombe en panne, les autres nœuds peuvent reprendre ses tâches. Les nœuds communiquent entre eux pour surveiller leur état. En cas de panne d'un nœud, un autre prend le relais automatiquement (basculement ou failover).



**Kubernetes :** Kubernetes assure la haute disponibilité des applications en répliquant les Pods, en répartissant la charge et en remplaçant automatiquement les composants défaillants. Il adapte également les ressources en fonction de la charge pour garantir un environnement stable et évolutif [23].



**Sauvegarde :** Il existe trois types de sauvegarde : la complète, qui copie toutes les données ; l'incrémentielle, qui ne sauvegarde que les changements depuis la dernière sauvegarde ; et la différentielle, qui enregistre les modifications depuis la dernière sauvegarde complète. Chaque méthode optimise différemment l'espace et le temps de sauvegarde. [23]



**La répartition de charge :** La répartition de charge consiste à distribuer le trafic entre plusieurs serveurs pour éviter la surcharge d'un seul serveur, améliorant ainsi la performance et la fiabilité des applications. [24]

La haute disponibilité est essentielle pour assurer la continuité des services en minimisant les interruptions. Elle repose sur des techniques comme la redondance, le basculement automatique et la répartition de charge.

---

### **CHAPITRE III :**

### **ETUDE COMPARATIVE DES DIFFERENTES SOLUTIONS OPEN SOURCE ET DE CLOUD**

---

Dans ce chapitre, nous nous intéressons à l'interaction entre le cloud computing et les solutions de VoIP, en mettant particulièrement l'accent sur les PBX (Private Branch Exchange) modernes, tels qu'Asterisk, ainsi que d'autres alternatives populaires. Le cloud, en permettant un hébergement flexible, évolutif et sécurisé, a ouvert de nouvelles perspectives pour les solutions de communication, notamment en rendant plus accessibles des infrastructures auparavant réservées aux grandes entreprises.

### **3.1. ETUDE COMPARATIVE DES SOLUTION DE VOIP**

Nous allons étudier et comparer les serveurs de communication open source en fonction de leurs performances, de leur simplicité et de leur interopérabilité, afin d'évaluer leur compatibilité avec différents systèmes de communication VoIP.

#### **3.1.4. Etude des différents serveurs de communication Open source PABX**

Dans le domaine des communications VoIP (Voice over IP), plusieurs solutions open source se distinguent, chacune ayant ses propres caractéristiques, avantages et inconvénients. Voici une comparaison détaillée des IP PBX, des proxys SIP et des clients VoIP open source, en mettant en lumière leurs similitudes et leurs différences.

 **Astérisk** : Astérisk est une plateforme PBX open source très modulable, capable de gérer un large éventail de codecs (G.711, G.722, GSM, Speex, iLBC, et, via licence, G.729 ou G.723.1). Grâce à son architecture en modules, il prend en charge de nombreux protocoles de signalisation (SIP, IAX2, MGCP, H.323) et peut intégrer, par des extensions tierces, des codecs modernes comme Opus. Cette flexibilité en fait un choix privilégié pour des scénarios VoIP variés, allant du simple serveur de petites entreprises aux infrastructures d'opérateurs télécom. [49]

 **FreePBX** : FreePBX est une interface web qui s'appuie sur Asterisk pour simplifier la configuration et l'administration. Elle expose tous les paramètres de codecs et de trunks d'Asterisk dans une console graphique intuitive, permettant aux utilisateurs non spécialistes de mettre en place, gérer et maintenir un PBX complet (y compris la licence G.729, l'enregistrement des appels, les menus vocaux interactifs, etc.) sans recourir à la ligne de commande. [50]

 **FusionPBX** : FusionPBX propose une interface web pour FreeSWITCH et hérite de ses performances médias. FreeSWITCH supporte nativement un grand nombre de codecs (G.722, SILK, Opus, Speex, iSAC, BroadVoice, et, en pass-through, G.729 ou AMR) ainsi que des flux vidéo (H.264, VP8, H.263). Cette richesse

fonctionnelle et sa conception multi-locataires en font une solution adaptée aux environnements exigeants et aux déploiements à grande échelle. [51]

 **Issabel** : Issabel est un projet dérivé d'Elastix basé sur Asterisk et intégrant des services complémentaires (fax, messagerie, reporting). Il reprend l'ensemble des codecs et protocoles d'Asterisk, tout en offrant, via une console unifiée, des modules prêts à l'emploi pour la communication unifiée. Issabel se présente comme une distribution ISO clé en main, destinée aux organisations souhaitant déployer rapidement une plateforme de téléphonie et de collaboration. [52]

 **OpenSIPS** : OpenSIPS est un serveur SIP orienté signalisation, agissant comme proxy ou routeur sans traiter directement les flux médias. Il inspecte et manipule la négociation SDP pour diriger les appels et applique des règles avancées de routage, de sécurité ou de contrôle de qualité. Pour le traitement RTP (transcodage, relais, enregistrement), il s'appuie généralement sur des moteurs externes tels que rtpengine, garantissant ainsi haute performance et forte capacité de montée en charge. [53]

### 3.1.5.Tableau récapitulatif

Le tableau 3.1 récapitule les principales solutions open source les plus utilisées en VoIP,

PBX	Codecs audio supportés	Protocoles de signalisation	Complet (fonctionnalités)	Performance VoIP
Asterisk	G.711, G.729, G.722, Opus, iLBC, GSM, Speex, AMR	SIP (PJSIP et chan_sip), IAX2, MGCP, H.323	★★★★★ Complet et extensible	★★★★★ Très performant, modulaire
FreePBX	Basé sur Asterisk : mêmes codecs supportés	SIP (via PJSIP/chan_sip), IAX2	★★★★★ complet, intuitive	★★★★ Bonne pour PME, selon Asterisk
Issabel	G.711, G.729, G.722, GSM, Opus (via Asterisk)	SIP, IAX2	★★★★★ UC intégrée (fax, email, chat)	★★★★ Adaptée aux PME
FusionPBX	G.711, G.722, G.729, Opus, Speex, iLBC	SIP (via FreeSWITCH)	★★★★★ Multi-tenant, puissant	★★★★★ Très scalable
OpenSIP	Dépend du backend (principalement G.711, G.729, Opus)	SIP (registrar, proxy, redirect), WebRTC	★★★ Backend SIP très puissant	★★★★★★ Ultra-performant, haut débit

Tableau 3.1:Tableau récapitulatif des différents PAB-X

### **3.1.6. Choix de solution : Cas d'Astérisk pour la VoIP**

Astérisk est une solution idéale pour la création d'une plateforme VoIP dans un environnement cloud tel qu'AWS, et ce pour plusieurs raisons fondamentales qui combinent flexibilité, performance, évolutivité, et compatibilité avec une grande variété de protocoles et de technologies. Voici un ensemble d'arguments justifiant ce choix :

- ☞ **Flexibilité et modularité** : Astérisk est une plateforme de communication flexible et modulaire, permettant d'intégrer divers services (voix, vidéo, messagerie, etc.) et de personnaliser ses fonctionnalités selon les besoins spécifiques de l'entreprise. Cela en fait une solution VoIP évolutive idéale pour le cloud, avec des fonctionnalités comme la gestion des appels, la messagerie vocale, la mise en attente, et les conférences.
- ☞ **Support de plusieurs protocoles de communication** : Astérisk est compatible avec plusieurs protocoles de signalisation (SIP, IAX, H.323, MGCP), ce qui facilite son intégration avec diverses infrastructures et appareils de communication. Cette polyvalence est essentielle dans un environnement cloud, où la diversité des besoins peut être importante.
- ☞ **Communauté et documentation riches** : Astérisk bénéficie d'une large communauté d'utilisateurs et de développeurs, offrant une abondance de documentation, forums et ressources d'assistance. Cela facilite la mise en œuvre, la configuration et la maintenance, notamment dans des environnements complexes comme AWS, réduisant les risques techniques et améliorant l'efficacité du projet.

Le choix Astérisk pour la création de la plateforme VoIP dans un environnement cloud est une décision stratégique qui combine la puissance de l'open-source, la flexibilité du cloud, et la robustesse d'une solution éprouvée. Ce choix garantit une solution VoIP évolutive, sécurisée, et performante, capable de répondre aux besoins spécifiques d'une entreprise tout en offrant une gestion efficace des coûts.

## **3.2. ETUDE COMPARATIVE DES SOLUTION DU CLOUD COMPUTING**

Le Cloud Computing a connu une forte adoption en raison de ses avantages comme la flexibilité, la scalabilité et la réduction des coûts. Les principales plateformes dominantes sont

Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform (GCP), ainsi que d'autres solutions comme Oracle Cloud, chacune avec ses caractéristiques et avantages spécifiques.

#### ❖ **Amazon Web Services (AWS)**

Amazon Web Services (AWS) est un leader dans le domaine du cloud computing, avec une gamme étendue de services adaptés aux besoins des entreprises, y compris des solutions spécifiques pour la VoIP comme Amazon Connect et Amazon Chime. AWS est reconnu pour sa fiabilité, sa scalabilité et ses options de résilience. Ses multiples zones de disponibilité et son réseau privé mondial offrent une latence faible, ce qui est essentiel pour garantir la qualité des appels VoIP. De plus, AWS propose des outils de gestion comme CloudFormation et Elastic Load Balancer, permettant de gérer facilement les services en fonction des besoins croissants d'une plateforme VoIP [46].

Cependant, bien qu'AWS soit robuste et flexible, ses coûts peuvent rapidement augmenter à mesure que l'utilisation des ressources croît, notamment avec des services à la demande. Il peut aussi être complexe à gérer pour des équipes non expertes, avec de nombreux services et outils à apprendre pour configurer et déployer une solution VoIP. Le modèle de tarification à l'usage peut également rendre difficile la gestion des coûts à long terme, bien qu'il soit possible de réduire ces coûts grâce à des instances réservées [46].

#### ❖ **Microsoft Azure**

Microsoft Azure s'intègre parfaitement avec l'écosystème Microsoft, ce qui le rend particulièrement adapté pour les entreprises déjà utilisatrices de technologies Microsoft, telles que Windows Server ou Active Directory. Pour la VoIP, Azure propose Azure Communication Services, qui permet de créer facilement des solutions de communication en temps réel, y compris les appels VoIP. Azure offre également une scalabilité impressionnante, ainsi qu'un environnement hybride qui permet de combiner des infrastructures sur site avec des ressources cloud, idéal pour les entreprises ayant besoin de flexibilité [47].

Cependant, Azure peut être un peu moins adapté pour des solutions VoIP prêtes à l'emploi comparées à AWS. Son modèle de tarification peut être complexe et difficile à prévoir, en particulier pour des services spécifiques comme la VoIP. Le suivi et la gestion des coûts peuvent donc être des défis, et l'intégration avec certains outils tiers peut être plus compliquée. Malgré

cela, Azure reste une plateforme solide pour les entreprises cherchant à adopter une approche hybride ou utilisant déjà des produits Microsoft [47].

#### ❖ Google Cloud Platform (GCP)

Google Cloud Platform (GCP) se distingue par ses performances réseau exceptionnelles et ses capacités en calcul haute performance, ce qui en fait un excellent choix pour les applications VoIP à faible latence. GCP propose également des API de communication et des outils d'analyse avancés qui peuvent être utilisés pour améliorer les services VoIP, comme la transcription en temps réel ou l'analyse des appels. De plus, l'intégration avec des technologies modernes comme Kubernetes et Google Kubernetes Engine permet de déployer des applications VoIP en utilisant des microservices et des containers, offrant une flexibilité maximale [48].

Cependant, GCP manque d'un écosystème VoIP prêt à l'emploi comme celui d'AWS ou Azure. Bien qu'il soit excellent pour les développeurs cherchant à créer des solutions personnalisées, il peut ne pas offrir autant de services ou d'outils spécifiquement adaptés à des déploiements VoIP traditionnels. De plus, le modèle de tarification à la demande peut devenir complexe à gérer au fur et à mesure que les besoins augmentent, et les outils comme Stackdriver pour le monitoring nécessitent une configuration minutieuse [48].

Le tableau (3.2) récapitulatif des principales solutions Coud.

Critère	AWS	Microsoft Azure	Google Cloud Platform (GCP)
<b>Nombre de régions (2024)</b>	33 régions	60 régions	40 régions
<b>Nombre de zones de disponibilité (AZ)</b>	105 AZ	116 AZ	121 zones
<b>Max. vCPU par VM</b>	224 vCPU (X1e)	384 vCPU (SAP HANA sur Azure Large Instances)	160 vCPU (types de machines optimisées pour la mémoire)
<b>Max. RAM par VM</b>	3904 GiB (~4 To)	8192 GiB (~8 To)	3844 Go (~3,8 To)
<b>Fréquence max. CPU</b>	Jusqu'à 3,1 GHz (Xeon Platinum), 2,6 GHz (Graviton3)	Jusqu'à 3,7 GHz (Xeon E-2176G), 2,7 GHz (Xeon Platinum 8168)	Jusqu'à 3,8 GHz (Xeon Skylake)

Critère	AWS	Microsoft Azure	Google Cloud Platform (GCP)
<b>Types d'instances</b>	Générales (M), calcul (C), mémoire (R), stockage (I), GPU (P, G), ARM (Graviton)	Générales (D, E), calcul (F), mémoire (M), GPU (NC, ND), SAP HANA	Générales (N1, N2), calcul (C2), mémoire (M2, M3), GPU (A2), ARM (Tau T2A)
<b>Systèmes d'exploitation supportés</b>	Windows Server, Amazon Linux, Ubuntu, RHEL, SUSE, Debian, CentOS, Oracle Linux, FreeBSD	Windows Server, Ubuntu, RHEL, SUSE, CentOS, Debian, Oracle Linux	Windows Server, Ubuntu, RHEL, SUSE, Debian, CentOS, CoreOS
<b>Services DevOps</b>	AWS CodePipeline, CodeBuild, CodeDeploy, CloudFormation	Azure DevOps, Azure Pipelines, Azure Resource Manager	Cloud Build, Cloud Deployment Manager, Cloud Code
<b>Équilibrage de charge</b>	Elastic Load Balancing (ALB, NLB, CLB)	Azure Load Balancer, Azure Application Gateway	Cloud Load Balancing (HTTP(S), TCP/UDP, SSL Proxy)
<b>Authentification &amp; autorisation</b>	IAM, Cognito, IAM Identity Center	Azure Active Directory (AD), Entra ID, Azure AD B2C	Identity Platform, Firebase Authentication, IAM
<b>PaaS</b>	AWS Elastic Beanstalk, AWS Lambda, AWS App Runner	Azure App Service, Azure Functions, Azure Spring Apps	App Engine, Cloud Functions, Cloud Run
<b>Chiffrement des données</b>	AWS KMS, CloudHSM, chiffrement au repos et en transit	Azure Key Vault, Azure Dedicated HSM, chiffrement au repos et en transit	Cloud KMS, Cloud HSM, chiffrement au repos et en transit
<b>Pare-feu &amp; sécurité réseau</b>	AWS WAF, AWS Firewall Manager, AWS Network Firewall	Azure Firewall, Azure WAF, Azure DDoS Protection	Cloud Armor, Cloud Next Generation Firewall (NGFW)
<b>Gestion identités</b>	IAM, Cognito, IAM Identity Center	Azure Active Directory, Microsoft Entra ID, Azure AD B2C	Identity Platform, Firebase Authentication, Cloud Identity

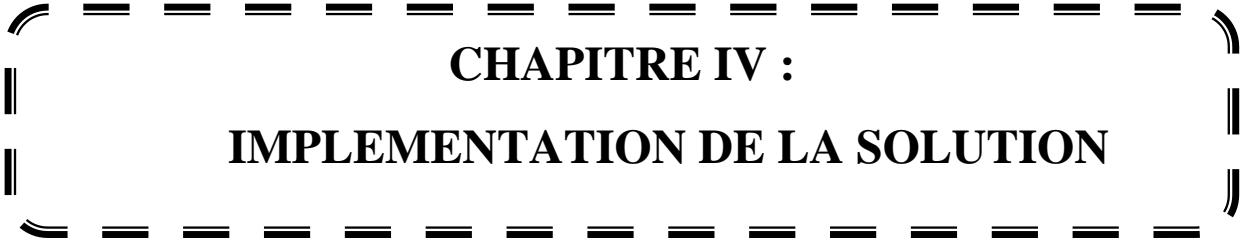
Tableau 3.2: Tableau récapitulatif des différents cloud

### **3.3. CHOIX DE SOLUTION : CAS AWS**

Le choix d'AWS (Amazon Web Services) pour la création de la plateforme VoIP (Voice over IP) présente plusieurs avantages stratégiques et techniques. Voici un argumentaire détaillé pour justifier ce choix :

- ☛ **Scalabilité et Elasticité** : L'un des principaux avantages d'AWS est sa scalabilité. Grâce à des solutions élastiques comme EC2, AWS permet d'ajuster les ressources en temps réel en fonction de la demande. Cela est particulièrement utile pour une plateforme VoIP, où les variations de trafic et d'utilisateurs peuvent être importantes, garantissant ainsi une qualité de service optimale sans surcharger l'infrastructure.
- ☛ **Haute Disponibilité (High Availability)** : Pour une plateforme VoIP, la disponibilité est essentielle. AWS propose des zones de disponibilité dans différentes régions géographiques, permettant de concevoir des architectures résilientes et tolérantes aux pannes. L'utilisation de plusieurs AZ (Availability Zones) et de load balancers (comme ELB) garantit que même en cas de défaillance d'un centre de données, les appels VoIP peuvent continuer sans interruption.
- ☛ **Sécurité et Conformité** : La sécurité est essentielle pour les plateformes VoIP, et AWS offre des outils robustes pour la protéger. Cela inclut IAM pour le contrôle d'accès, VPC pour l'isolement de l'infrastructure, ainsi que des mécanismes de chiffrement des données en transit et au repos. AWS respecte également plusieurs normes de sécurité internationales, assurant ainsi la confidentialité et l'intégrité des données des utilisateurs.
- ☛ **Performance et Fiabilité des Ressources** : Les instances EC2 d'AWS offrent une performance élevée adaptée à la VoIP, notamment pour le traitement en temps réel des paquets RTP, essentiel pour la qualité des appels. AWS propose des instances optimisées pour la mémoire, le processeur et le réseau, adaptées à la gestion du trafic VoIP. De plus, AWS garantit
- ☛ **Gestion Simplifiée avec Amazon Connect** : AWS propose également des services dédiés à la gestion des communications, comme Amazon Connect, qui est une solution cloud pour les centres de contact. Ce service permet de créer et gérer facilement des interactions vocales et des services VoIP à grande échelle, tout en s'intégrant de manière fluide avec les autres services AWS.

Cette étude comparative des différentes solutions de VoIP et de Cloud a permis de mettre en lumière les forces et les limites de chaque option, tant sur le plan technique qu'économique. Il en ressort que le choix d'une solution dépend étroitement des besoins spécifiques de l'organisation, notamment en termes de sécurité, de flexibilité, de coûts et de facilité de déploiement. Tandis que certaines solutions offrent une grande évolutivité via le Cloud, d'autres privilégient le contrôle et la personnalisation via des installations sur site. Une analyse rigoureuse des exigences métiers est donc essentielle pour adopter la solution la plus adaptée.



**CHAPITRE IV :**  
**IMPLEMENTATION DE LA SOLUTION**

Après une étude approfondie des solutions cloud et open source, le choix s'est porté sur AWS et Astérisk pour leur performance et leur pertinence dans le projet. Cette décision repose sur la solidité de ces solutions pour le déploiement d'une plateforme VoIP.

#### **4.3. LES ACTEURS INTERVENANT DU SYSTEME**

Les différents acteurs d'une solution VoIP, tels que l'administrateur cloud, l'administrateur d'entreprise et les utilisateurs finaux, jouent des rôles essentiels dans la gestion, la configuration et l'utilisation quotidienne du système, chacun contribuant à son bon fonctionnement et à son efficacité globale.

##### **4.1.4. Administrateur cloud**

Les administrateurs Cloud sont responsables de la conception, du déploiement, de la gestion et de la surveillance des ressources cloud d'une entreprise. Ils optimisent les performances, renforcent la sécurité des données et réduisent les coûts d'exploitation, assurant ainsi le bon fonctionnement des applications et services dans le Cloud, tout en contribuant à la compétitivité de l'entreprise.



##### **Ceci implique notamment :**

- ✚ provisionnement de ressources de stockage,
- ✚ la gestion des droits d'accès au service,
- ✚ la résolution des problèmes techniques.
- ✚ Ajouter de nouveaux administrateurs.
- ✚ Supprimer des administrateurs.
- ✚ Ajouter de nouveaux utilisateurs.
- ✚ Créer un projet.
- ✚ Créer de nouvelles machines virtuelles.
- ✚ Gérer et créer un réseau.
- ✚ Ajouter de nouveaux serveurs stockage.

##### **4.1.5. Administrateur entreprise**

L'administrateur de l'entreprise est toute personne ayant reçu un compte d'accès lui permettant d'ajouter de nouveaux clients, d'activer de nouvelles fonctionnalités, de gérer les utilisateurs, de configurer les paramètres du système, de surveiller les activités, et d'effectuer d'autres modifications pour assurer le bon fonctionnement.

#### 4.1.6. Utilisateur de VoIP

Les utilisateurs finaux disposent de droits d'accès limités, leur permettant d'utiliser les fonctionnalités essentielles telles que passer des appels, recevoir des messages et participer à des réunions, sans pouvoir modifier les paramètres système.

### 4.2. ARCHITECTURE DE LA SOLUTION PROPOSEE

Cette architecture est spécifiquement conçue pour une plateforme VoIP déployée sur AWS, avec pour objectif principal d'assurer une communication fiable, sécurisée et hautement disponible. Elle permet de protéger les serveurs VoIP (comme Asterisk), de gérer efficacement les pics de trafic d'appels, de répartir intelligemment les charges, tout en assurant la résilience du système face aux pannes.

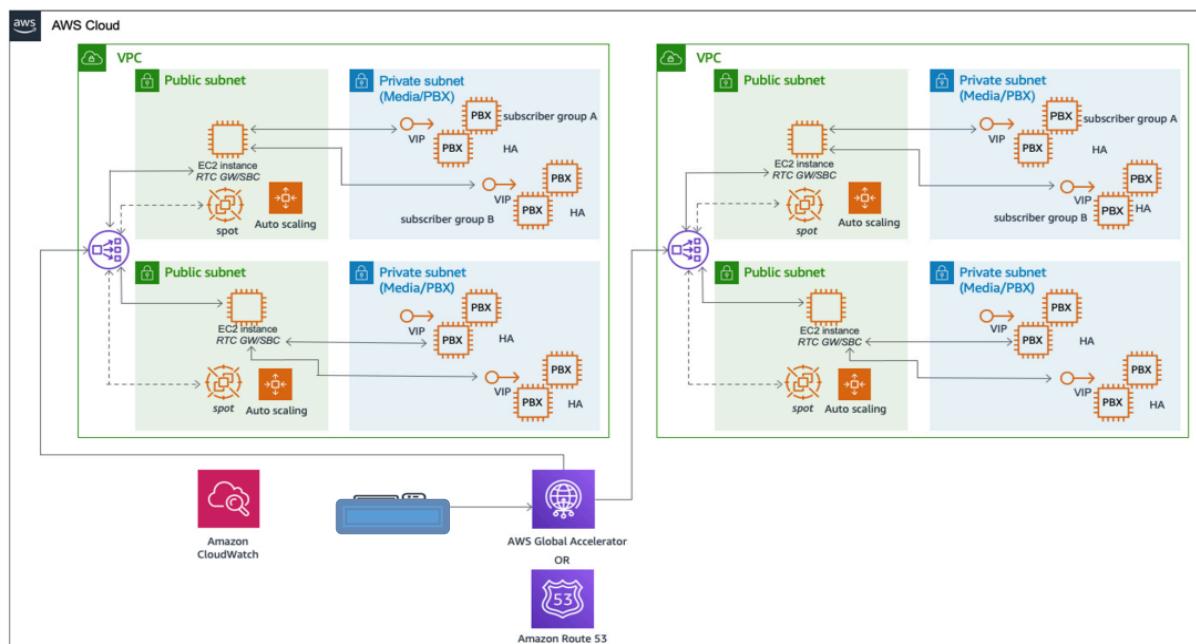


Figure 4.1: Architecture générale

#### ❖ Les services AWS dans l'architecture ☞ Mise en place des VPC (Virtual Private Cloud)

J'ai commencé par créer deux VPC distincts pour répartir la charge et garantir la tolérance aux pannes. Chaque VPC peut être située dans une zone de disponibilité (AZ) différente, ou même dans une autre région AWS si besoin.

Cette séparation géographique permet d'assurer que même si une zone tombe en panne, l'autre peut continuer à traiter les appels normalement.

Chaque VPC est structurée avec :

Des **sous-réseaux publics** pour les composants exposés à Internet.

Des **sous-réseaux privés** pour les composants sensibles comme les serveurs PBX.



### **Passerelles VoIP dans les sous-réseaux publics (RTC Gateway / SBC)**

Dans les sous-réseaux publics, j'ai déployé des instances EC2 configurées comme passerelles VoIP ou Session Border Controllers (SBC).

Ces instances :

Gèrent les flux SIP et médias.

Protègent le cœur du système contre les attaques.

Font office de barrière sécurisée entre Internet et les serveurs internes.

J'ai activé l'auto scaling pour qu'AWS ajuste automatiquement le nombre d'instances selon la charge, et j'utilise des instances Spot pour réduire les coûts.



### **Serveurs PBX dans les sous-réseaux privés**

Dans les sous-réseaux privés, j'ai installé les serveurs PBX responsables du traitement des appels.

Je les ai organisés en deux groupes de cluster (A et B), chacun étant composé de :

Deux PBX en haute disponibilité (actif/passif).

Une adresse IP virtuelle (VIP) qui reste toujours active, même en cas de panne d'un PBX.



### **Haute disponibilité des PBX (HA)**

La haute disponibilité est au cœur de mon architecture. Chaque paire de PBX utilise un système de surveillance mutuelle (heartbeat). Si le PBX principal tombe, le secondaire prend le relais automatiquement en reprenant l'adresse VIP.

Cela garantit une continuité de service sans interruption pour les utilisateurs finaux.

**Répartition de charge (Load Balancing)** : Pour assurer la performance et la disponibilité, j'ai intégré le network load balancing entre les deux sous réseaux pour en cas de perte d'information ou de panne survenant dans un SBC le trafic soit automatiquement rediriger vers l'autre sous réseau publique.

**AWS Global Accelerator** : il optimise les trajets réseau et dirige les utilisateurs vers la zone la plus performante. Il détecte les pannes et bascule automatiquement le trafic vers une autre région ou un autre VPC.

**Amazon Route 53** : c'est un DNS intelligent qui distribue le trafic selon la localisation, la latence ou l'état de santé des composants. En cas de panne d'un VPC, il redirige les appels vers un autre site sans intervention manuelle.



## Supervision en temps réel avec Amazon CloudWatch

Pour que l'ensemble fonctionne de manière fiable, j'ai mis en place une surveillance complète avec Amazon CloudWatch.

Grâce à CloudWatch, je peux :

Surveiller l'état des serveurs (CPU, mémoire, bande passante).

Suivre la qualité des appels (jitter, latency, pertes de paquets).

Recevoir des alertes instantanées en cas de panne ou de comportement anormal.

Déclencher des actions automatiques, comme redémarrer un serveur ou déclencher un scaling.

## ❖ interopérabilité dans l'architecture

L'interopérabilité entre les services AWS est un élément fondamental dans une architecture VoIP, car elle permet une réponse automatique et coordonnée en cas de panne.

Lorsqu'une instance RTC Gateway / SBC ou devient défaillante, le Load Balancer (équilibrEUR de charge) détecte le problème via les sondes de santé et redirige immédiatement le trafic vers d'autres instances encore saines. En parallèle, cette défaillance est signalée au groupe Auto Scaling, qui identifie que l'instance ne fonctionne plus correctement.

Grâce à cette information, Auto Scaling déclenche automatiquement le remplacement de l'instance défaillante par une nouvelle, sans aucune intervention de l'administrateur système. Cela garantit que le nombre de serveurs disponibles reste toujours optimal pour maintenir le service.

De son côté, Amazon CloudWatch, le service de surveillance de l'infrastructure, joue aussi un rôle essentiel. Il permet de détecter en temps réel des anomalies de performance comme une surconsommation de RAM, un CPU trop élevé, ou un volume disque saturé. Lorsqu'un de ces problèmes est repéré, CloudWatch peut déclencher des alertes ou automatiser des actions correctives, comme le redémarrage d'une instance ou l'augmentation de ressources.

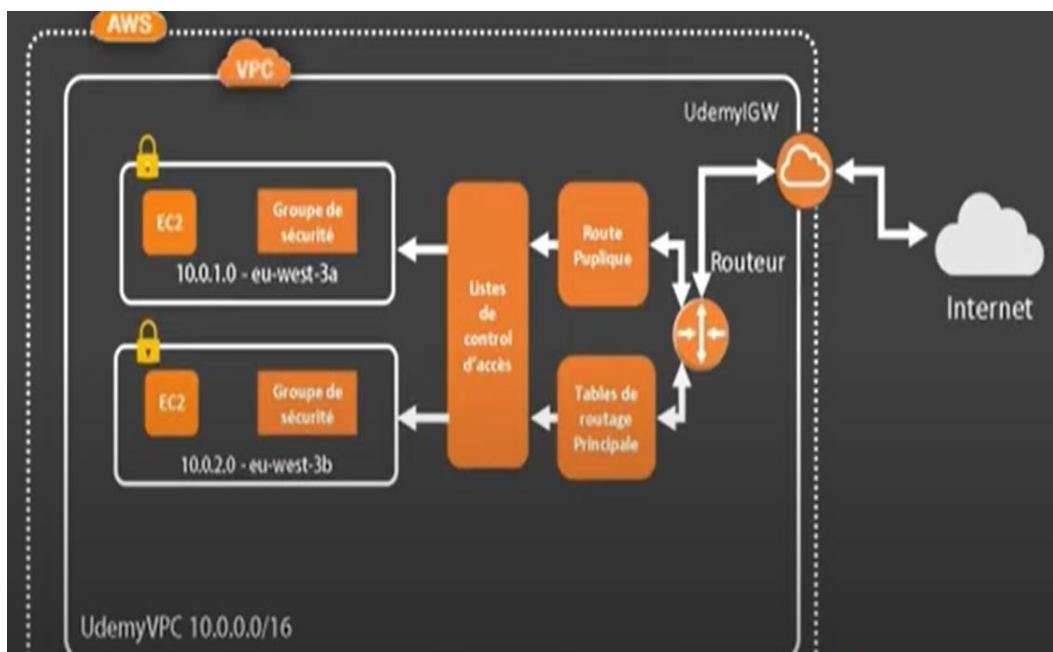
### 4.3.Implémentation avec aws

Cette section décrit l'implémentation d'une solution VoIP open source sur AWS, en combinant l'évolutivité et la fiabilité du cloud avec les technologies open source. Elle couvre la configuration de l'infrastructure, les bonnes pratiques d'intégration, et la gestion des ressources pour assurer une solution performante, sécurisée et évolutive, tout en optimisant les coûts et garantissant une haute disponibilité.

Compte tenu du fait que seuls les services gratuits d'AWS (Free Tier) peuvent être utilisés, il est essentiel de concevoir une architecture qui reste légère et compatible avec les limites de ce matériel. Cette approche vise à garantir que notre machine puisse répondre aux besoins de base tout en déléguant les charges les plus exigeantes à des ressources cloud gratuites et adaptées.

### ❖ Maquette de teste

Cette maquette de test présente une architecture réseau sur AWS, mettant en avant la haute disponibilité, l'évolutivité et la résilience. Elle repose sur une infrastructure VPC avec deux instances EC2 réparties sur deux zones de disponibilité, garantissant tolérance aux pannes et répartition de charge. Chaque instance est associée à un volume EBS pour la persistance des données, assurant ainsi performance et disponibilité. Cette configuration sert de base pour tester une solution fiable et évolutive dans un environnement cloud AWS, en suivant les bonnes pratiques.



*Figure 4.2: Maquette de teste*

- ☞ **Virtual Private Cloud (VPC)** : Une infrastructure réseau privée avec un CIDR de **10.0.0.0/16**, assurant l'isolation et la gestion des ressources AWS.
- ☞ **Instances EC2 (serveurs)** : Deux instances EC2 déployées dans des deux différentes zones de disponibilité renforçant la haute disponibilité et la tolérance aux pannes.
- ☞ **Elastic Block Store (EBS)** : Chaque instance EC2 sera rattachée à un volume de stockage Elastic Block Store (EBS) pour répondre aux besoins de persistance des données.

- 👉 **Load Balancer** : sera intégré pour répartir de manière équilibrée le trafic réseau entre les deux instances EC2, assurant une gestion efficace des requêtes.
- 👉 **Groupes de sécurité** : Associés à chaque instance EC2, ils gèrent les règles d'accès réseau pour garantir une sécurité granulaire.
- 👉 **Listes de contrôle d'accès (ACL)** : Fournit une sécurité supplémentaire au niveau du sous-réseau
- 👉 **Internet Gateway**: Permet aux instances de communiquer avec Internet via des routes configurées dans les tables de routage.
- 👉 **Table d'acheminement principale et route publique** : Diriger le trafic interne et externe en fonction des besoins.

#### 4.3.1. Préparation de l'environnement AWS (Amazon Web Service)

Ce projet consiste à préparer un environnement AWS pour explorer les services cloud et déployer une machine virtuelle Ubuntu. Le guide couvre les étapes clés, de la création du compte AWS à l'installation et la configuration des instances Ubuntu sur Amazon EC2. Il est essentiel de comprendre les services AWS, notamment le calcul, le stockage, la mise en réseau et la sécurité, pour concevoir une architecture cloud robuste, évolutive et adaptée aux besoins du projet.

##### ❖ Crédit du compte et accès à la console Amazon Web Service (AWS)

Pour créer un compte AWS, j'ai utilisé l'adresse e-mail djibysene707@gmail.com et une carte Visa Ecobank. Une fois le compte activé, je me suis connecté pour la première fois en tant qu'utilisateur racine via le portail principal : <https://aws.amazon.com>. Ce portail permet de gérer les ressources, d'explorer les services et de configurer l'environnement AWS

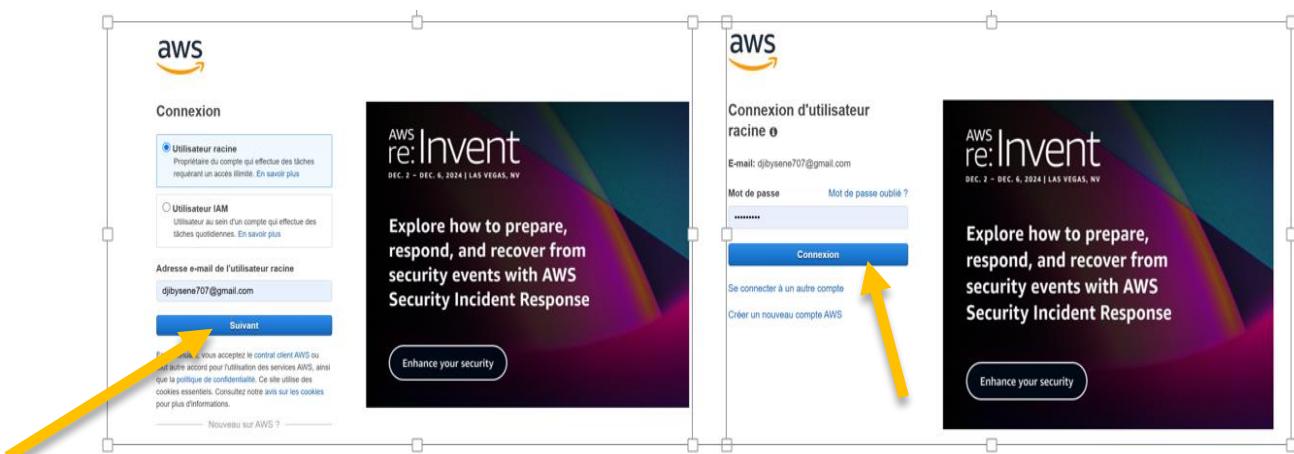
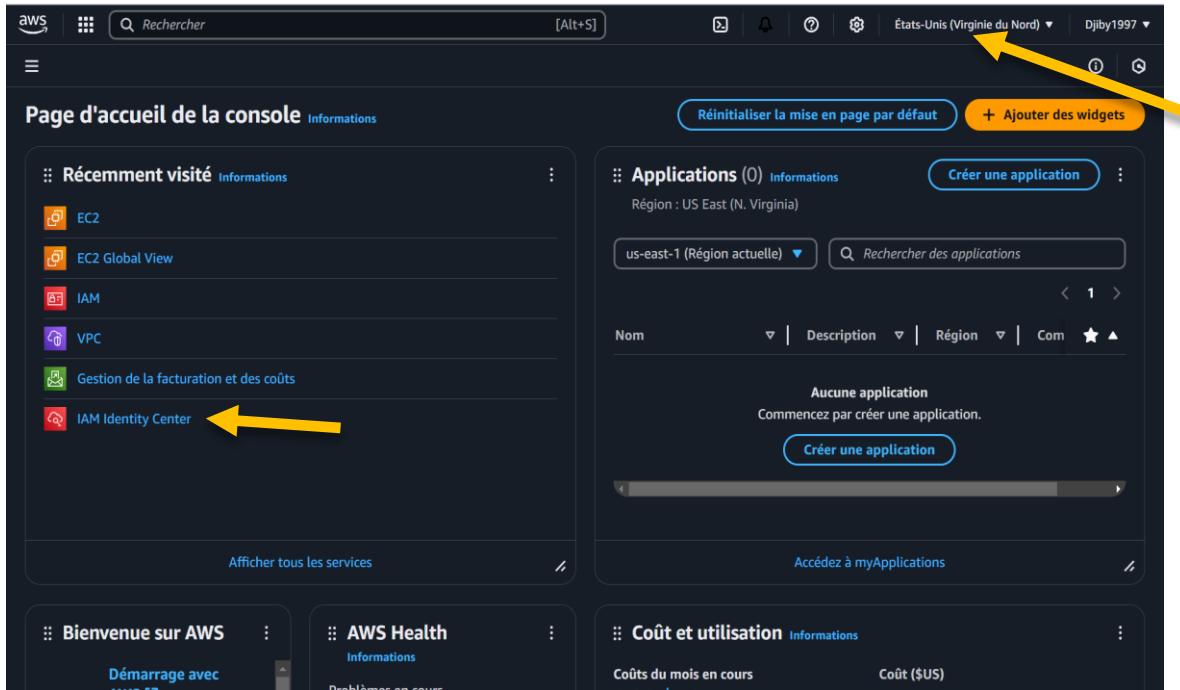


Figure 4.3 : création compte et accès à AWS

Après connexion à la console AWS, j'ai exploré les services nécessaires pour créer mon réseau virtuel (VPC) et choisi la région correspondante. J'ai également personnalisé l'interface en modifiant la couleur pour améliorer l'expérience utilisateur.



*Figure 4.4 : Console d'Amazon Web Service (AWS)*

#### ❖ Sécurité, identité et conformité de la console AWS

Le premier service que j'ai exploré est IAM (Identity and Access Management), accessible dans la catégorie Sécurité, identité et conformité de la console AWS. Ce service est crucial pour la gestion de la sécurité du compte AWS. Il permet de définir des politiques d'accès, de gérer les identités (utilisateurs, groupes et rôles) et de contrôler l'accès aux ressources AWS. Toutefois, étant le seul utilisateur de ce compte, je me connecte toujours directement en tant qu'utilisateur racine (root), ce qui me donne un accès complet à tous les services AWS sans nécessiter la création d'utilisateurs supplémentaires.



Figure 4.5:Sécurité, identité et conformité de la console AWS

#### ❖ Crédit d'un VPC (Virtual Private Cloud)

La création d'un VPC (Virtual Private Cloud) sur AWS permet de définir un réseau virtuel privé isolé dans le cloud, offrant un contrôle complet sur l'adressage IP, les sous-réseaux, les tables de routage et les passerelles. Le VPC est essentielle pour organiser les ressources réseau et assurer une sécurité optimale. Il est important de choisir une région AWS pour le déploiement, car cela affecte la latence, la disponibilité et la conformité des services.

 **Choisir une région AWS :** Le choix de la région pour la création d'un VPC est déterminant, car il influence sur la latence et la performance des services. Bien que le Sénégal soit la région cible pour ce projet, j'ai choisi la Virginie du Nord (North Virginia) region AWS. En effet, cette région est géographiquement plus proche du Sénégal, ce qui permet d'optimiser la latence et d'assurer une meilleure performance pour les utilisateurs finaux dans cette zone géographique.

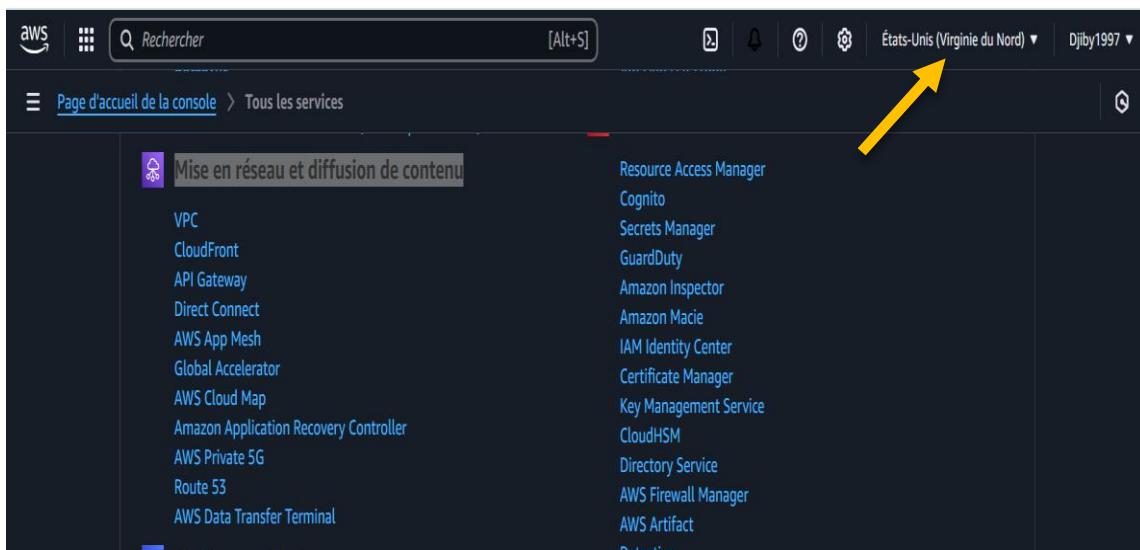


Figure 4 .6 : Choisir une région AWS pour un VPC

**Création d'un VPC :** Pour ce faire, nous accèderons à l'option "Créer un VPC" dans la console AWS, où nous définirons une balise de nom pour identifier facilement notre VPC. Nous spécifierons également un bloc d'adressage CIDR en IPv4. Concernant l'allocation, nous opterons pour l'option par défaut, car l'allocation dédiée est facturée, ce qui ne correspond pas à notre objectif d'utiliser exclusivement les services gratuits proposés par AWS pour cette phase de déploiement.

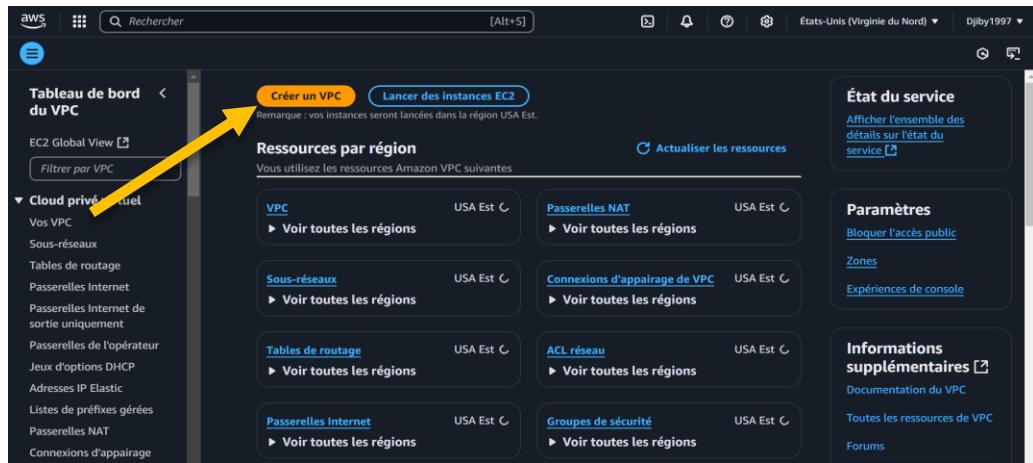
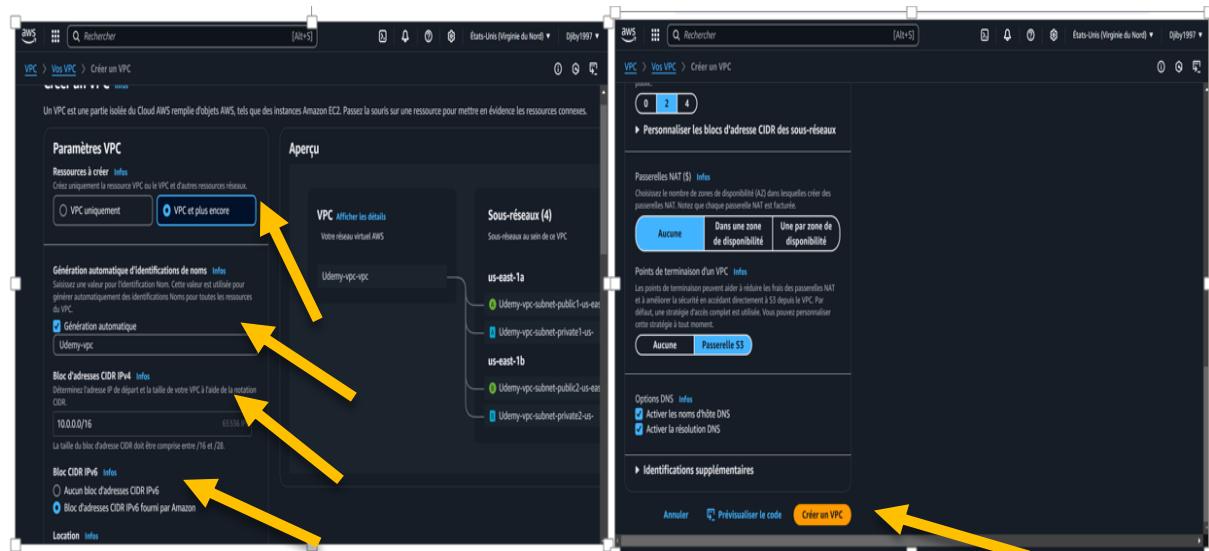


Figure 4.7 : Création d'un VPC

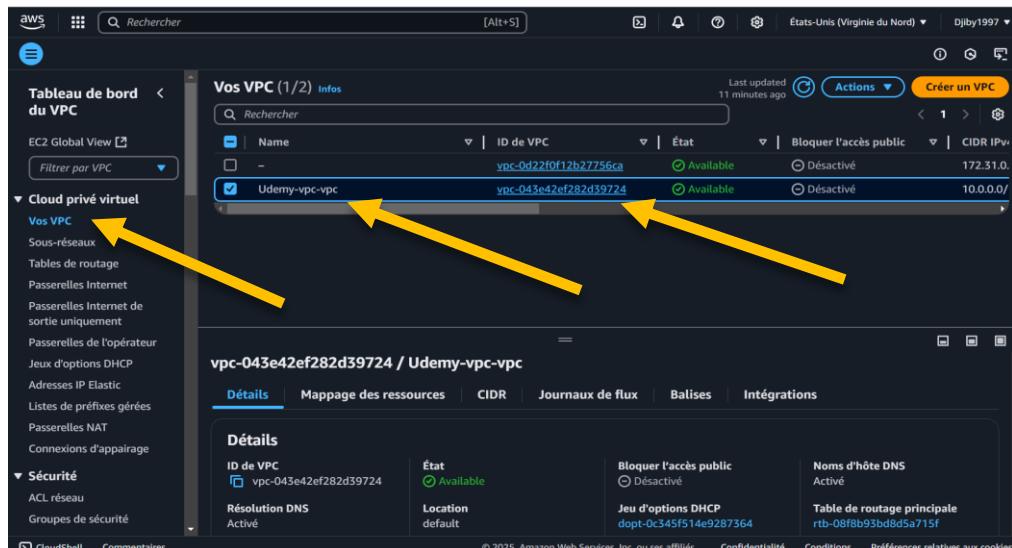
Pour le bloc d'adressage CIDR IPv4, nous avons choisi 10.0.0.0/16 afin de disposer d'un maximum d'adresses IP dans notre sous-réseau. En ce qui concerne l'adressage IPv6, nous avons opté pour un bloc fourni par Amazon. Cette configuration permet d'exploiter pleinement les adresses IPv6, qui sont devenues incontournables dans le monde informatique actuel en raison de leur capacité à répondre à la pénurie d'adresses IPv4.



*Figure 4 .7: Cr eation un VPC*

Une fois que nous cliquons sur "Cr eer un VPC", la console AWS affiche automatiquement tous les  l ments n cessaires   la mise en r seau. Parmi ces  l ments, on retrouve notamment les passerelles, les tables de routage, les points de terminaison, les sous-r seaux, ainsi que d'autres composants. Ces  l ments sont cr 『es et disponibles, mais en attente de leur configuration.

Certains de ces  l ments disposent de leur propre ID unique et le VPC aussi auquel ils sont associ s. Par exemple, dans notre cas, l'ID du VPC est **vpc-043e42ef282d39724**, ce qui confirme que le VPC a  t  cr 『e avec succ s. Pour obtenir une confirmation suppl mentaire de la cr 『ation du VPC, nous pouvons acc der   l'onglet "Vos VPC" dans la console AWS. Cet onglet r pertorie les VPC disponibles, notamment le VPC par d faut cr 『e par Amazon. Nous y retrouvons  galement notre nouveau VPC, nomm  Udemy-vpc, avec son ID unique : **vpc-043e42ef282d39724**. Ce d tail confirme que le VPC a bien  t  r cemment cr 『e avec succ s et est pr t    tre utilis  dans nos configurations.



*Figure 4 .7: Cr eation un VPC*

Une fois le VPC cr 『, AWS lui attribue automatiquement une table de routage principale, une liste de contr le d'acc s (ACL) et un groupe de s curit . Ces  l ments sont essentiels pour la gestion du r seau et la s curit  du VPC. Nous pouvons v rifier et confirmer l'existence de ces ressources soit en utilisant le nom du VPC Udemy-vpc soit en recherchant son ID unique **vpc-043e42ef282d39724** dans la console AWS. Cette m thode est particuli rement utile pour

valider les informations associées, notamment dans le cas du groupe de sécurité qui est directement lié au VPC.



**Table de routage :** Les tables de routage dans le cloud AWS réside dans leur rôle essentiel pour gérer le trafic réseau entre les ressources dans un Amazon VPC (Virtual Private Cloud). AWS crée automatiquement une table de routage principale pour les deux sous réseaux

The screenshot shows the AWS VPC Tables of Routing interface. On the left, there is a navigation sidebar with various options like 'Tableau de bord du VPC', 'Vos VPC', 'Tables de routage' (which is selected and highlighted in blue), and 'Passerelles Internet'. The main content area displays two tables of routing. The top table, titled 'Tables de routage (1/2) Infos', lists two entries: 'vpc-0d22f0f12b27756ca' and 'vpc-043e42ef282d39724 | Udemy-vpc-vpc'. The bottom table, titled 'Routes (2)', lists two routes: '2600:1f18:6abe:4b00::/56' with a 'local' target and '10.0.0.0/16' with a 'local' target, both marked as 'Actif' (Active) and 'Non' for propagation. A yellow arrow points from the sidebar's 'Tables de routage' option to the corresponding table in the main view.

Figure 4 .8 : Table de routage principale du VPC



**Création d'une table de routage spécifique :** Bien qu'AWS crée automatiquement une table de routage principale pour notre VPC, il est nécessaire de créer une table de routage spécifique pour notre sous-réseau public. Cette table de routage doit être configurée pour inclure une route vers l'extérieur, avec comme cible la passerelle Internet (Internet Gateway) attachée à notre VPC.

En associant cette table de routage au sous-réseau public, nous permettons à ce dernier d'être accessible depuis l'extérieur via Internet.

Figure 4 .9 : Table de routage public du VPC

**Liste de Contrôle d'accès (ACL) :** Les ACL (Access Control Lists) dans AWS jouent un rôle essentiel dans la gestion de la sécurité au niveau du réseau. Elles servent à contrôler l'accès au trafic entrant et sortant des sous-réseaux dans un Amazon VPC (Virtual Private Cloud).

Figure 4 .10 : Liste de contrôle d'accès du VPC

 **Groupe de sécurité :** Les groupes de sécurité (Security Groups) dans AWS sont des composants essentiels pour gérer la sécurité des ressources dans un Amazon VPC (Virtual Private Cloud). Ils fonctionnent comme des pare-feu virtuels stateful qui contrôlent le trafic entrant et sortant des instances au niveau de l'interface réseau.

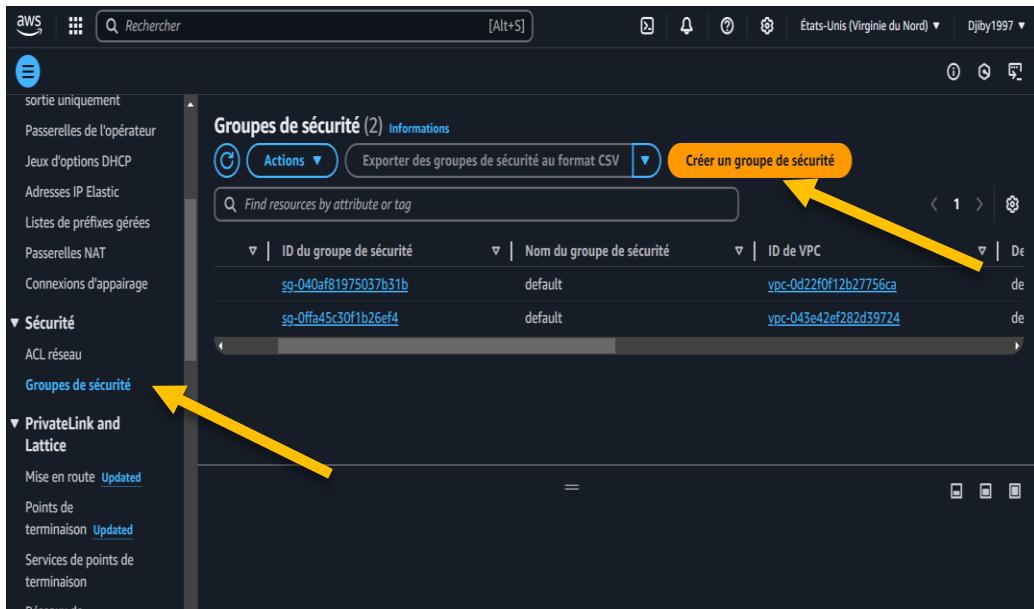


Figure 4 .11: Groupe de sécurité du VPC

 **Création des sous-réseaux :** Ensuite, nous allons créer deux sous-réseaux dans notre VPC, chacun délimité par une zone de disponibilité différente afin d'assurer une haute disponibilité. Chaque sous-réseau recevra un nom distinct et une plage d'adresses IPv4 spécifique.

Le premier sous-réseau aura la plage **10.0.1.0/24**.

Le second sous-réseau aura la plage **10.0.2.0/24**.

Ces sous-réseaux permettront de mieux organiser les ressources et de tirer parti des différentes zones de disponibilité pour améliorer la résilience de notre architecture.

Sous-réseaux (10) infos

Last updated 22 minutes ago

**Actions** ▾ **Créer un sous-réseau**

	Name	ID de sous-réseau	État	VPC
<input type="checkbox"/>	-	subnet-049adf40516564eb1	Available	vpc-0d22f0f12b27756ca
<input type="checkbox"/>	-	subnet-0e4cba1af6bf797a5	Available	vpc-0d22f0f12b27756ca
<input type="checkbox"/>	-	subnet-00a828da4e3fa4923	Available	vpc-0d22f0f12b27756ca
<input type="checkbox"/>	-	subnet-019b04445834d1705	Available	vpc-0d22f0f12b27756ca
<input type="checkbox"/>	-	subnet-074119eb53d1d4e57	Available	vpc-0d22f0f12b27756ca
<input type="checkbox"/>	-	subnet-07fc1ddfc0b08231e	Available	vpc-0d22f0f12b27756ca
<input type="checkbox"/>	Udemy-vpc-subnet-public2-us-east-1b	subnet-037363229191f4786	Available	vpc-043e42ef282d39724   Ude...

Figure 4 .12 : Création des sous-réseaux du VPC

💡 Le sous-réseau dont le nom est : 10.0.1.0-us-east-1a

**Créer un sous-réseau** infos

**VPC**

**ID de VPC**  
Crée des sous-réseaux dans ce VPC.

**CIDR IPv4** 10.0.0.0/16    **CIDR IPv6** 2602:1f18:5abe:4000::/56 (us-east-1)

**Paramètres du sous-réseau**  
Précisez les blocs d'adresse CIDR et la zone de disponibilité pour le sous-réseau.

**Zone de disponibilité** infos  
Choisissez la zone dans laquelle votre sous-réseau résidera ou laissez Amazon en choisir une pour vous.

**Nom du sous-réseau (subnet)**  
Créez une balise avec une clé « Name » et une valeur à spécifier.  
10.0.1.0-us-east-1a

**Zone de disponibilité** infos  
Choisissez la zone dans laquelle votre sous-réseau résidera ou laissez Amazon en choisir une pour vous.

**Zone de disponibilité** infos  
Choisissez la zone dans laquelle votre sous-réseau résidera ou laissez Amazon en choisir une pour vous.

**Balises - facultatif**

Figure 4 .13: Création du sous-réseau (10.0.1.0-us-east-1a) du VPC

💡 Le sous-réseau dont le nom est : 10.0.2.0-us-east-1b

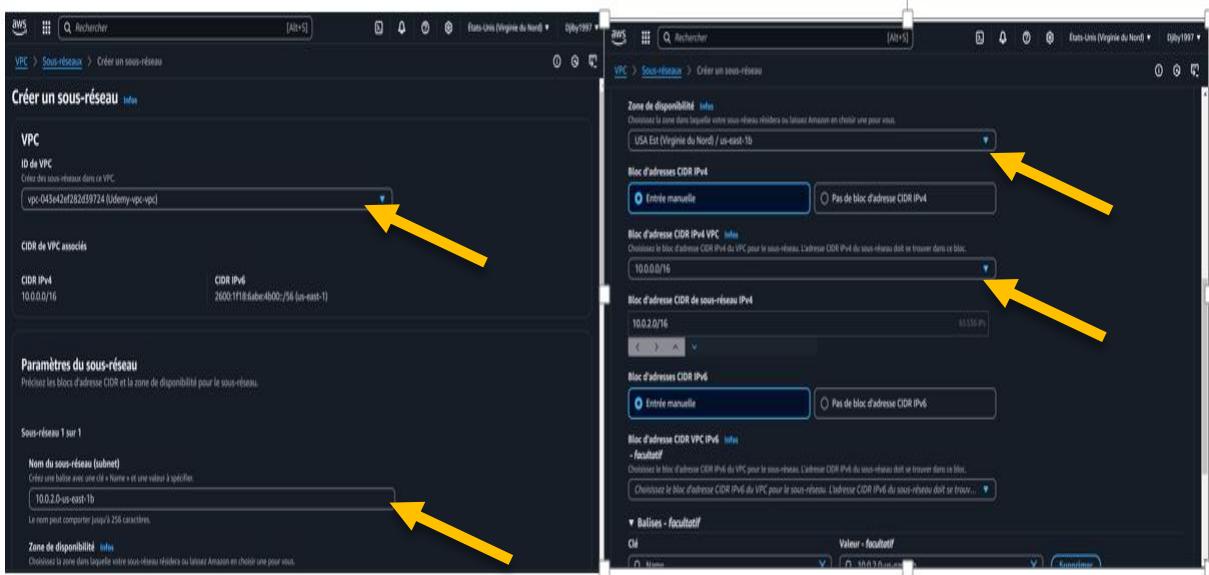


Figure 4 .14: Création du sous-réseau (10.0.2.0-us-east-1b) du VPC

La figure 4.15 montre que les deux sous-réseaux sont bien créées leurs noms et leurs ID

Tableau de bord du VPC					
Cloud privé virtuel		Sous-réseaux (5)			
Vos VPC					Actions ▾
<b>Sous-réseaux</b>					<a href="#">Créer un sous-réseau</a>
Tables de routage		Name	ID de sous-réseau	État	VPC
Passerelles Internet		-	<a href="#">subnet-0e4cba1af6bf797a5</a>	<span>Available</span>	<a href="#">vpc-0d22f0f12b27756ca</a>
Passerelles Internet de sortie uniquement		-	<a href="#">subnet-00a828da4e5fa4923</a>	<span>Available</span>	<a href="#">vpc-0d22f0f12b27756ca</a>
Passerelles de l'opérateur		-	<a href="#">subnet-019b04445834d1705</a>	<span>Available</span>	<a href="#">vpc-0d22f0f12b27756ca</a>
Jeux d'options DHCP		10.0.1.0-us-east-1a	<a href="#">subnet-0f09ac74a6e18e481</a>	<span>Available</span>	<a href="#">vpc-043e42ef282d39724   Udemy-vpc-vpc</a>
Adresses IP Elastic		10.0.2.0-us-east-1b	<a href="#">subnet-08359b215cb5bb765</a>	<span>Available</span>	<a href="#">vpc-043e42ef282d39724   Udemy-vpc-vpc</a>
Listes de prefixes gérées		Sélectionner un sous-réseau			
Passerelles NAT					

Figure 4 .15 : Création du sous-réseau VPC

- + **Configuration du sous réseau public :** Notre objectif est de configurer un sous-réseau public et un sous-réseau privé dans notre VPC. Dans notre cas : Nous choisissons le sous-réseau **10.0.1.0/24** comme sous-réseau public. Cela signifie que toute instance EC2 déployée dans ce sous-réseau pourra automatiquement se voir attribuer une adresse IP publique. Le sous-réseau **10.0.2.0/24**, quant à lui, sera configuré comme sous-réseau privé.

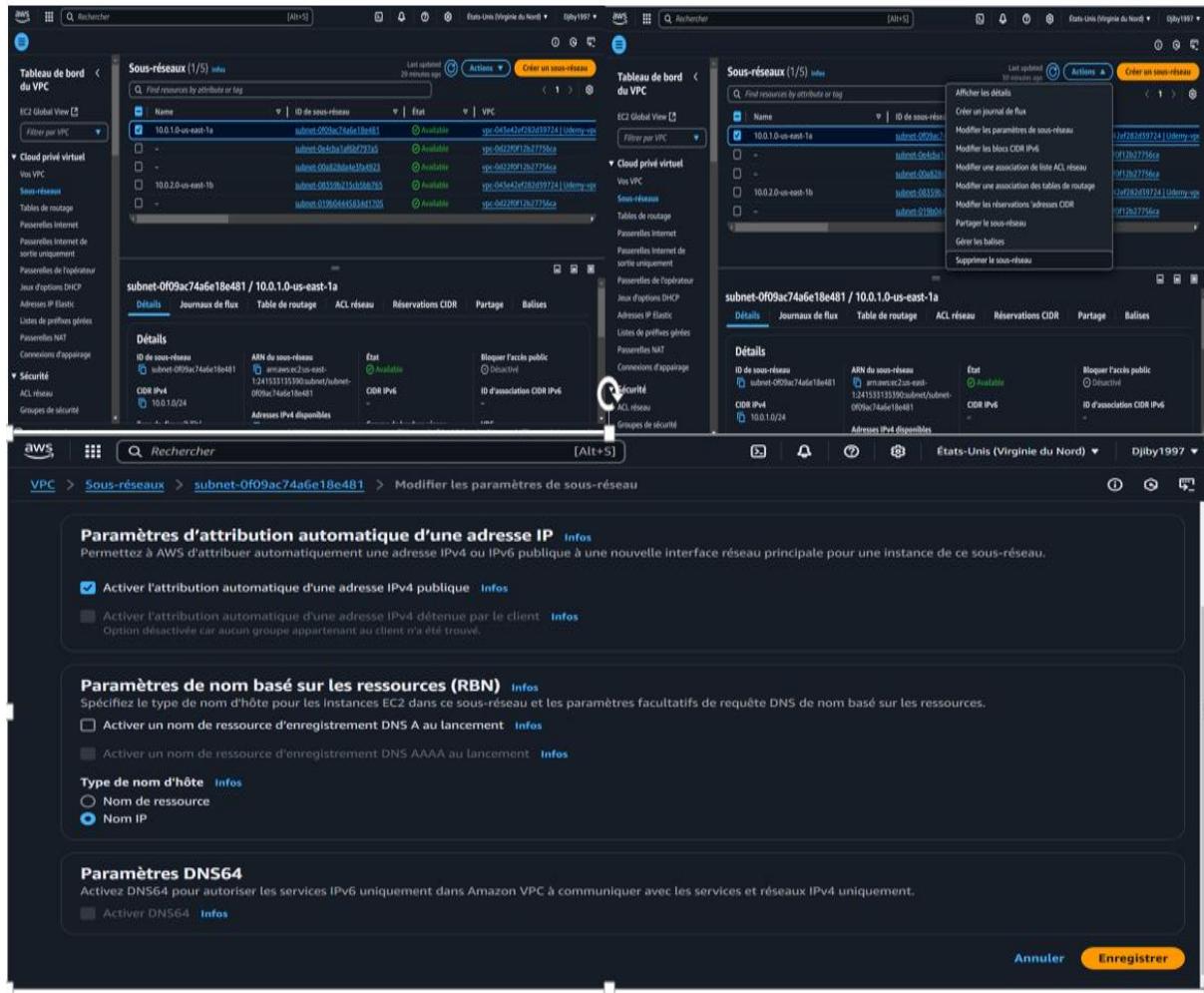
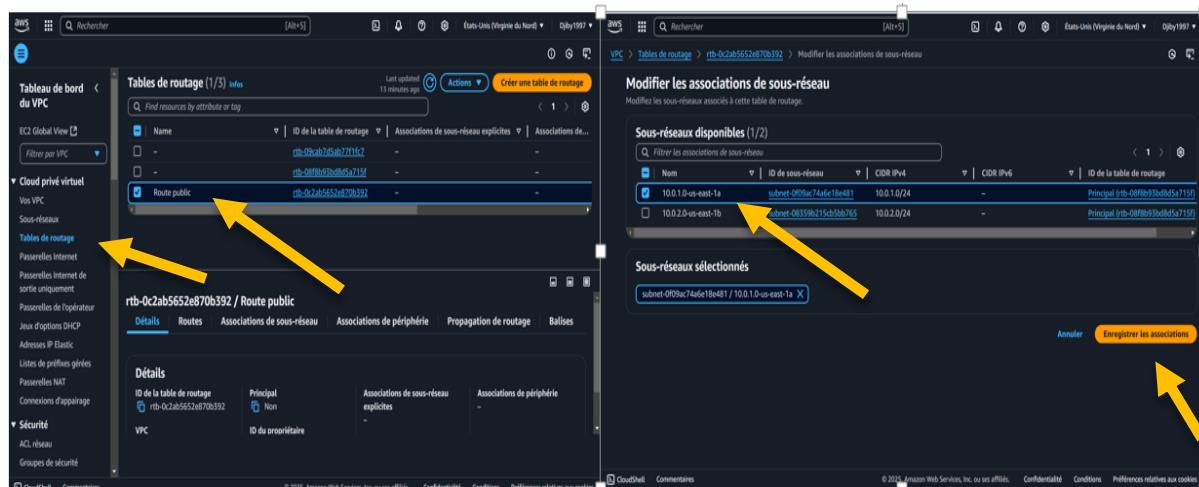


Figure 4 .16: Configuration du sous réseau public

**💡 Association de façon explicite le sous-réseau public à la table de routage publique :** Lorsque la route principale **10.0.0.0/16** est créée, les deux sous réseaux peuvent communiquer automatiquement entre eux via cette route.

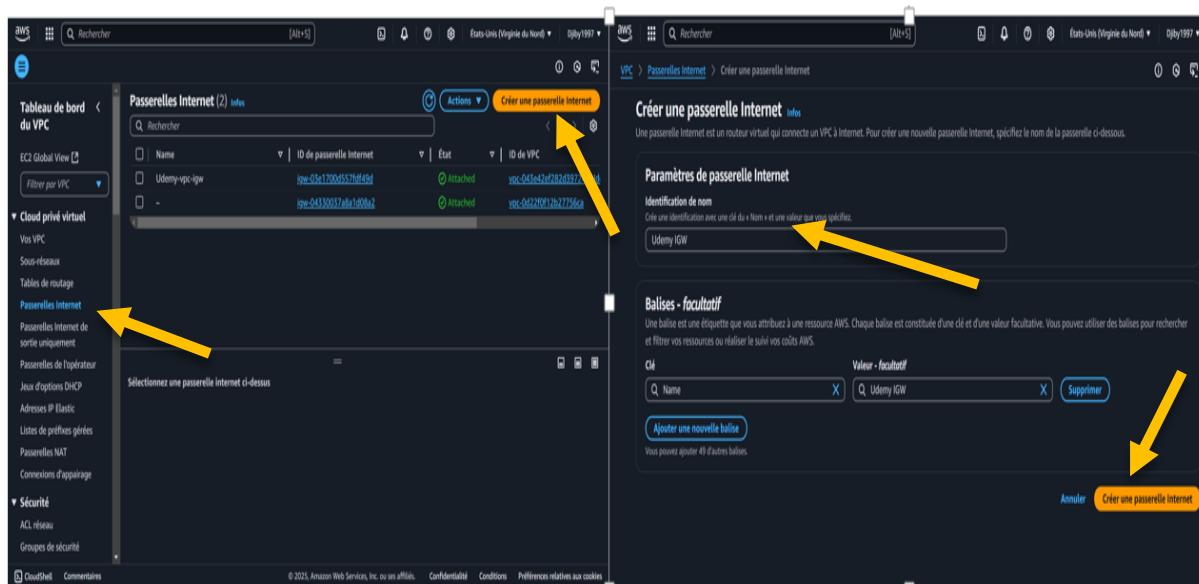
Cependant, cela permet également un accès potentiel à l'extérieur, ce qui constitue une faille de sécurité.

Pour pallier cette situation, nous procédons comme suit : Nous créons une table de routage public; Nous y ajoutons une route vers internet **0.0.0.0/0** avec comme cible la passerelle Internet (Internet Gateway) attachée au VPC ; Nous associons explicitons cette table de routage au sous-réseau public (**10.0.1.0/24**).



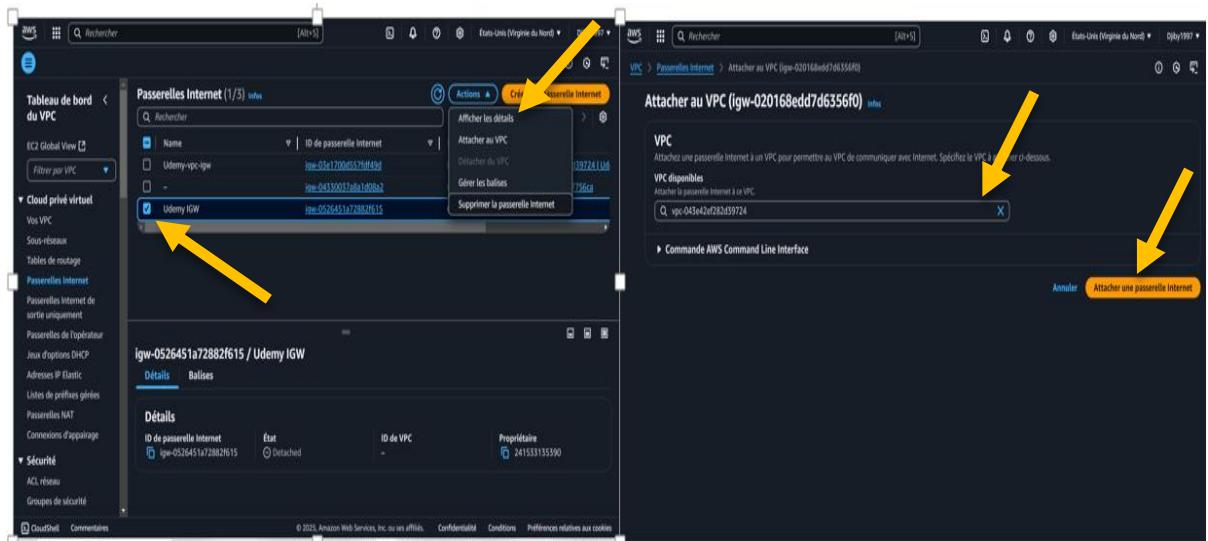
*Figure 4 .17: Association de façon explicite le sous-réseau public à la table de routage publique*

**Passerelle Internet :** est un composant essentiel qui sert de point d'accès pour connecter notre sous-réseau public à Internet. Elle sera configurée et attachée à notre VPC afin de permettre une communication bidirectionnelle entre les ressources de notre sous-réseau public et Internet.



*Figure 4 .19: passerelle internet*

Cette configuration garantit que les instances EC2 dans le sous-réseau public pourront envoyer et recevoir du trafic depuis l'extérieur, tout en maintenant une architecture réseau sécurisée et bien définie.



*Figure 4 .21: Configuration de la passerelle internet*

**Instance EC2 dans le sous-réseau public :** Pour configurer mon instance EC2 dans un sous-réseau public, je commence par accéder à la console AWS. Une fois dans le menu des instances, je crée une nouvelle instance et lui donne un nom. Comme je travaille avec un sous-réseau public, je choisis une clé SSH pour garantir un accès sécurisé à mon serveur. Ensuite, je sélectionne mon VPC, que j'identifie grâce à son nom et son ID, et je spécifie le sous-réseau 10.0.1.0/24.

Pour sécuriser une plateforme VoIP sur AWS, il est recommandé d'utiliser une clé SSH distincte de celle du sous-réseau public et de configurer un groupe de sécurité spécifique pour le sous-réseau privé. Par défaut, seul le port SSH (22) est ouvert. Pour permettre le bon fonctionnement de la VoIP, il faut ajouter des règles pour autoriser le trafic UDP sur le port 5060 (SIP) ainsi que les ports 10000-20000 pour le RTP et le RTCP, qui assurent la transmission de la voix et la gestion de la qualité en temps réel.

Enfin, pour le système d'exploitation, je choisis une AMI de type Ubuntu, comme Ubuntu Server 24.04 LTS. Avec cette configuration, mon serveur est prêt, sécurisé, et adapté à mes besoins réseau.

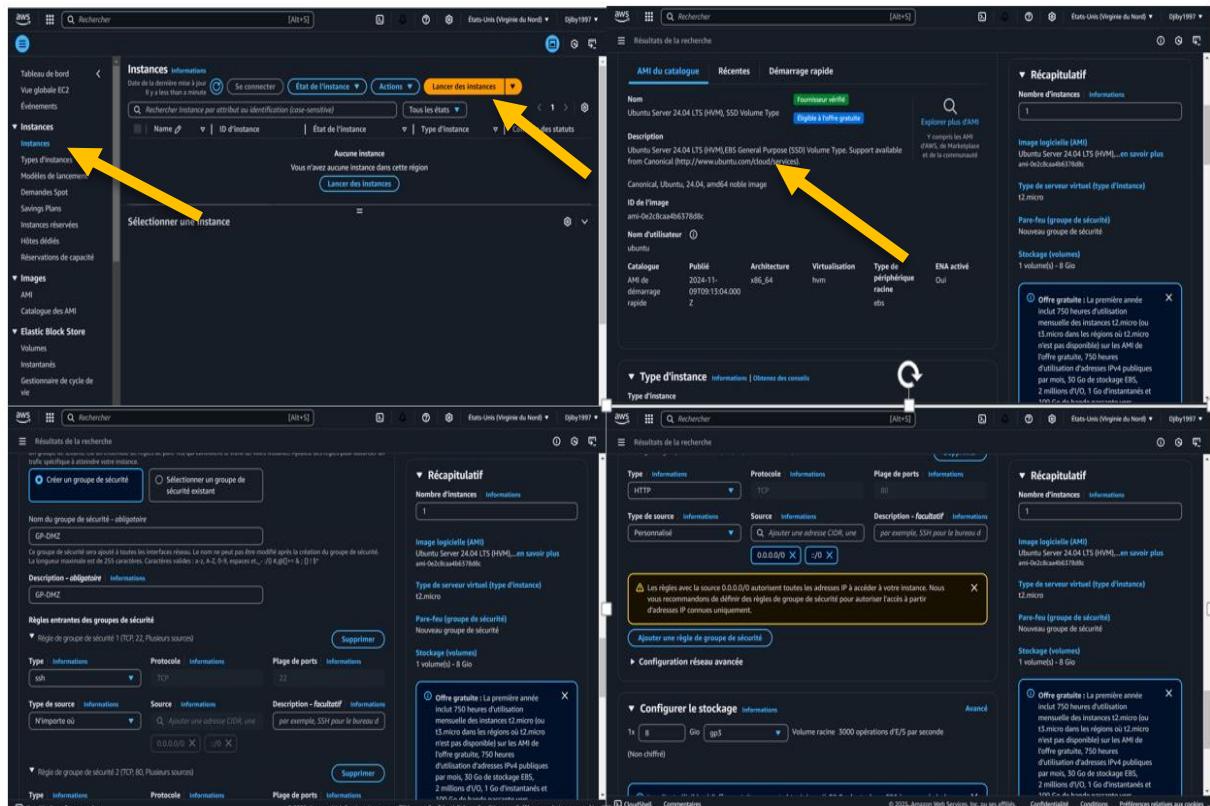


Figure 4.23 : Instance EC2 dans le sous-réseau public



Figure 4.26 : Instance EC2 dans le sous-réseau public

**Instance EC2 sous-réseau privé :** Pour configurer une instance EC2 dans un sous-réseau privé sur AWS, il faut utiliser le réseau 10.0.2.0/24, choisir une clé SSH distincte, et créer un groupe de sécurité adapté au trafic VoIP (UDP 5060, ports 10000–20000 pour RTP/RTCP). L’instance, basée sur Ubuntu Server 22.04 LTS, reçoit une adresse IP privée

sans IP publique, assurant une configuration sécurisée et adaptée aux besoins du projet.

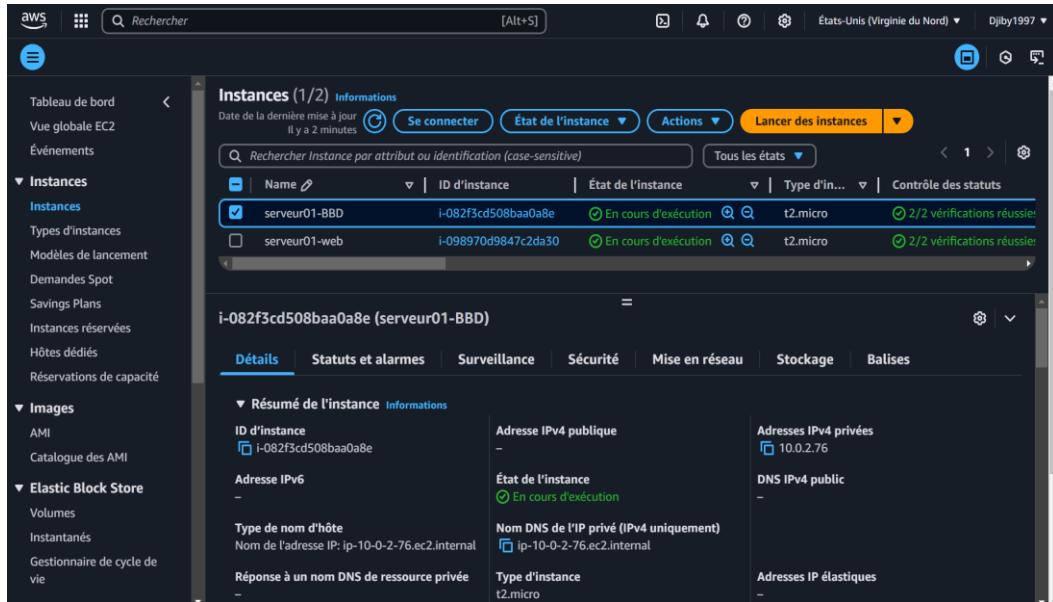


Figure 4.27 : Instance EC2 dans le sous-réseau privé

### 4.3.2. Installation du serveur asterisk

Après avoir préparé notre environnement AWS et déployé une instance EC2 à partir d'une AMI de type ubuntu dans un sous-réseau configuré, il est temps d'installer Asterisk, notre solution open source. Grâce à ses nombreuses fonctionnalités, Asterisk permettra de mettre en place un système de communication téléphonique robuste, incluant la gestion des appels, la messagerie vocale, les conférences et bien plus encore.

Nous avons utilisé une AMI dans AWS pour déployer un système d'exploitation Ubuntu 24.04, configuré avec au moins 4 Go de RAM et 16 Go d'espace disque. Cette configuration garantit une performance optimale et une capacité suffisante pour supporter Asterisk et ses fonctionnalités avancées.



#### Mise à jour du système :

```
sudo apt-get update
```

```
sudo apt-get upgrade
```



#### Installation des paquets nécessaires :

```
sudo apt-get install build-essential
```

```
sudo apt-get install git-core subversion wget libjansson-dev sqlite3 autoconf automake libxml2-dev libncurses5-dev libtool
```



#### Téléchargement de la source Asterisk :

Aller dans le répertoire des sources

```
cd /usr/src/
```

```
sudo wget http://downloads.asterisk.org/pub/telephony/asterisk/asterisk-20-current.tar.gz  
sudo tar zxf asterisk-20-current.tar.gz  
cd asterisk-*/
```



### Installation des dépendances spécifiques à Asterisk :

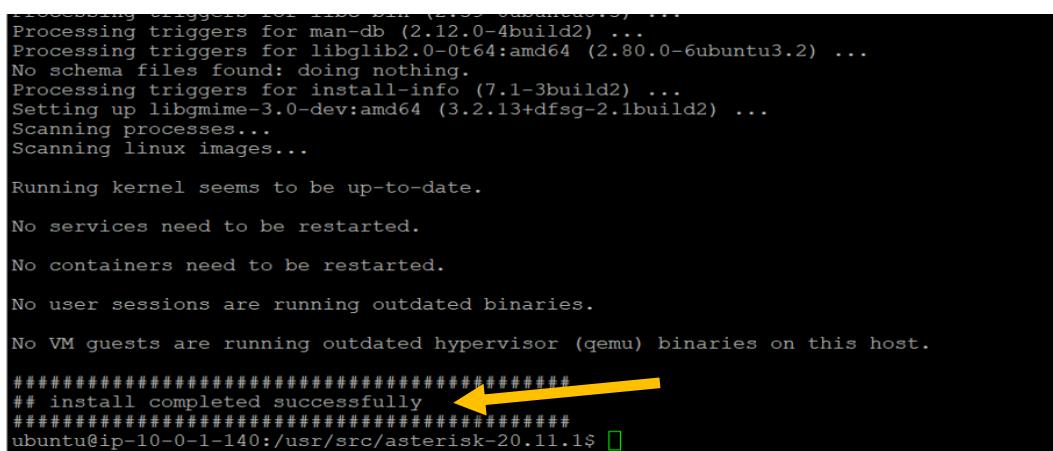
#### ⊕ Récupérer les sources MP3 :

```
sudo contrib/scripts/get_mp3_source.sh
```

#### ⊕ Installer les prérequis :

```
sudo contrib/scripts/install_prereq install
```

Après avoir exécuté avec succès le script ci-dessus, tous les packages requis seront installés.



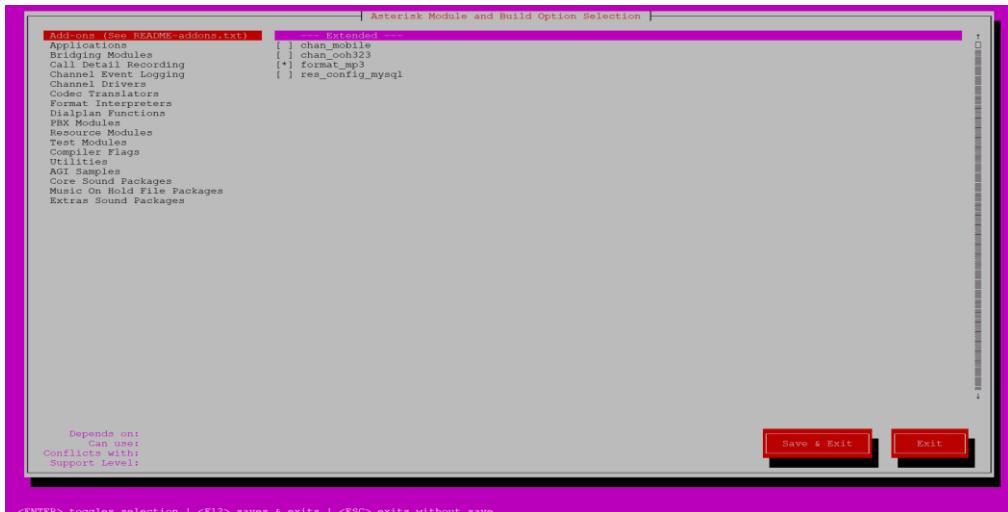
```
Processing triggers for man-db (2.12.0-4build2) ...  
Processing triggers for libglib2.0-0t64:amd64 (2.80.0-6ubuntu3.2) ...  
No schema files found: doing nothing.  
Processing triggers for install-info (7.1-3build2) ...  
Setting up libgmime-3.0-dev:amd64 (3.2.13+dfsg-2.1build2) ...  
Scanning processes...  
Scanning linux images...  
  
Running kernel seems to be up-to-date.  
  
No services need to be restarted.  
  
No containers need to be restarted.  
  
No user sessions are running outdated binaries.  
  
No VM guests are running outdated hypervisor (qemu) binaries on this host.  
  
#####  
## install completed successfully  
#####  
ubuntu@ip-10-0-1-140:/usr/src/asterisk-20.11.1$
```

Figure 4.28 : L'installation d'asterisk ubuntu dans AWS

### 4.3.3. Configuration du serveur asterisk

La configuration d'Asterisk repose sur un ensemble de fichiers texte qui permettent de définir les fonctionnalités et les comportements du système. Ces fichiers incluent la gestion des extensions, des utilisateurs, des protocoles VoIP (comme SIP ou PJSIP), et des règles de routage des appels. Une bonne compréhension de ces configurations est essentielle pour déployer et personnaliser un système de communication performant.

On sélectionne les modules que nous souhaitons compiler et installer. Accédez ensuite au menu système de sélection en tapant la commande suivante : **sudo make menuselect**.



*Figure 4.29: Configuration du serveur asterisk dans AWS*

Lorsque Nous avons terminé, appuyons sur F12pour enregistrer et quitter ou passez au Save and Exitbouton. Ensuite, appuyez sur Enter. Nous pourrons démarrer le processus de compilation en utilisant la makecommande, en :**sudo make -j2**

Nous verrons le message suivant :

```
prometheus.so
[LD] res_realtime.o -> res_realtime.so
[LD] res_resolver_unbound.o -> res_resolver_unbound.so
[LD] res_rtp_asterisk.o -> res_rtp_asterisk.so
[LD] res_rtp_multicast.o -> res_rtp_multicast.so
[LD] res_security_log.o -> res_security_log.so
[LD] res_smidi.o -> res_smidi.so
[LD] res_snmp.o -> res_snmp.so
[LD] res_snmp_agent.o -> res_snmp.so
[LD] res_snmp_disco.o -> res_snmp_disco.so
[LD] res_sorcery_config.o -> res_sorcery_config.so
[LD] res_sorcery_memory.o -> res_sorcery_memory.so
[LD] res_sorcery_memory_cache.o -> res_sorcery_memory_cache.so
[LD] res_sorcery_realtime.o -> res_sorcery_realtime.so
[LD] res_speech.o -> res_speech.so
[LD] res_speech_aap.o -> res_speech_aap.so
[LD] res_stp.o -> res_stp.so
[LD] res_stasis.o stasis/app.o stasis/commando.o stasis/control.o stasis/messaging.o stasis/stasis_bridge.o -> res_stasis.so
[LD] res_stasis_answer.o -> res_stasis_answer.so
[LD] res_stasis_device_state.o -> res_stasis_device_state.so
[LD] res_stasis_playback.o -> res_stasis_playback.so
[LD] res_stasis_recording.o stasis/recording/stored.o -> res_stasis_recording.so
[LD] res_stasis_snoop.o -> res_stasis_snoop.so
[LD] res_statsd.o -> res_statsd.so
[LD] res_stun_monitor.o -> res_stun_monitor.so
[LD] res_timing_timerfd.o -> res_timing_timerfd.so
[LD] res_timing_pthread.o -> res_timing_pthread.so
[LD] res_tonedetect.o -> res_tonedetect.so
[LD] res_xmp.o -> res_xmp.so
[CC] format_mp3.c -> format_mp3.o
[CC] mp3/common.c -> mp3/common.o
[CC] mp3/dct64_i386.o -> mp3/dct64_i386.o
[CC] mp3/decode_ntom.c -> mp3/decode_ntom.o
[CC] mp3/lameif3.c -> mp3/lameif3.o
[CC] mp3/tabinit.o -> mp3/tabinit.o
[CC] mp3/interface.c -> mp3/interface.o
[LD] format_mp3.o mp3/common.o mp3/dct64_i386.o mp3/decode_ntom.o mp3/layer3.o mp3/tabinit.o mp3/interface.o -> format_mp3.so
Building Documentation For: channels bpl apps codecs formats cdr cel bridges funcs tests main res addons
+----- Asterisk Build Complete -----+
+ Asterisk has successfully been built, and +
+ can be installed by running:
+     ./install
+         make install
+-----+
ubuntu@ip-10-0-1-140:/usr/src/asterisk-20.11.15
```

*Figure 4.29: Configuration du serveur asterisk dans AWS*

L'étape suivante consiste à installer Asterisk et ses modules en exécutant la commande suivante : **sudo make install**

Une fois l'installation terminée, le script affichera le message ci-dessous :

```

asterisk-moh-opsound-siren14-2.03.tar.gz 100%[=====] 6.22M 5.47MB/s in 1.s
2025-01-19 06:41:08 (5.47 MB/s) - 'asterisk-moh-opsound-siren14-2.03.tar.gz' saved [6518720/6518720]

make[1]: Leaving directory '/usr/src/asterisk-20.11.1/sounds'
find rest-api -name "*.json" | while read x; do \
/usr/bin/install -c -m 644 $x "/var/lib/asterisk/rest-api" ; \
done
+--- Asterisk Installation Complete -----
+
+ YOU MUST READ THE SECURITY DOCUMENT +
+
+ Asterisk has successfully been installed. +
+ If you would like to install the sample +
+ configuration files (overwriting any +
+ existing config files), run:
+
+ For generic reference documentation:
+ make samples
+
+ For a sample basic PBX:
+ make basic-pbx
+
+
+----- or -----
+
+ You can go ahead and install the asterisk +
+ program documentation now or later run:
+
+ make progdocs
+
+ **Note** This requires that you have +
+ doxygen installed on your local system +
+-----+
ubuntu@ip-10-0-1-140:/usr/src/asterisk-20.11.1$ 

```

*Figure 4.29: Configuration du serveur asterisk dans AWS*



**installation des exemples de fichiers de configuration.**

Installons les fichiers de configuration génériques, avec la documentation de référence en :

**sudo make samples**

Ou installons les fichiers de configuration de base du PBX, en : **sudo make basic-pbx**

Installons le initscript Asterisk en utilisant la commande suivante : **sudo make config**

Il est bon de mettre à jour le cache des bibliothèques partagées avec: **sudo ldconfig**



### **Créons un utilisateur Asterisk**

Par défaut, Asterisk s'exécute en tant qu'utilisateur root. Pour des raisons de sécurité, nous devrons créer un nouvel utilisateur système et configurer Asterisk pour qu'il s'exécute en tant que nouvel utilisateur. Pour créer un nouvel utilisateur système, exécutez la commande ci-dessous : **sudo adduser --system --group --home /var/lib/asterisk --no-create-home --gecos "Asterisk PBX" asterisk**

Pour configurer Asterisk pour qu'il s'exécute en tant asterisk qu'utilisateur, ouvrons **/etc/default/asterisk** le fichier. Ensuite, supprimez le commentaire des 2 lignes ci-dessous : **AST\_USER="asterisk"** et **AST\_GROUP="asterisk"**

```

GNU nano 2.2
# Startup configuration for the Asterisk daemon
#
# Uncomment the following and set them to the user/groups that you
# want to run Asterisk as. NOTE: this requires substantial work to
# be sure that Asterisk's environment has permission to write the
# files required for its operation, including logs, its comm-
# socket, the asterisk database, etc.
#AST_USER="asterisk" AST_USER='asterisk'
# If you DON'T want Asterisk to start up with terminal colors, comment
# this out.
#COLORS=yes
# If you want Asterisk to run with a non-default configuration file,
# uncomment the following option, and set the value appropriately.
#ALTCONF=/etc/asterisk/asterisk.conf
# In the case of a crash, Asterisk may create a core file. Uncomment
# if you want this behavior.
#COREDUMP=yes
# Asterisk may establish a maximum load average for the system. This
# may be useful to prevent a flood of calls from taking down the system.
#MAXLOAD=4
# Or, if you'd prefer, you can limit the maximum number of calls.
#MAXCALLS=1000
# Default console verbosity. This may be raised or lowered on the console.
# Note this is analogous to the -v command line switch, which by default
# will cause Asterisk to start in console mode and run in the foreground,
# unless the always fork (-F) option is also provided.
#VERBOSITY=0
# Enable internal timing if the DAHDI timer is available. The default
# behaviour is that outbound packets are phase locked to inbound packets.
#Enabling this option causes them to be locked to the internal DAHDI
# timer instead.
#INTERNALTIMING=yes
# Start all recordings into a temporary directory, before moving them to
# their final location.
#TEMPRECORDINGLOCATION=yes

```

*Figure 4.29: Configuration du serveur asterisk dans AWS*

Ensute, ajoutons asteriskl'utilisateur dialoutainsi que audioles groupes, en utilisant la commande suivante : **sudo usermod -a -G dialout,audio asterisk**

Nous devrons également modifier la propriété et les autorisations de tous les fichiers et répertoires Asterisk, après quoi Asterisk pourra accéder à ces fichiers :

**sudo chown -R asterisk: /var/{lib,log,run,spool}/asterisk /usr/lib/asterisk /etc/asterisk et sudo chmod -R 750 /var/{lib,log,run,spool}/asterisk /usr/lib/asterisk /etc/asterisk**



#### Démarrer Asterisk

Nous pouvons maintenant démarrer le service Asterisk. Faites-le avec la commande suivante : **sudo systemctl start asterisk**.

L'étape suivante consiste à activer le service Asterisk pour qu'il démarre au démarrage à l'aide de la commande suivante : **sudo systemctl enable asterisk**



#### Configurer le pare-feu

Le pare-feu protégera notre serveur du trafic indésirable. Par défaut, SIP utilise le port UDP 5060, pour ouvrir le port, exécutez : **sudo ufw allow 5060/udp**

Nous souhaitons activer le protocole en temps réel, nous devrions également ouvrir la plage de ports en procédant comme suit : **sudo ufw allow 10000:20000/udp**

#### **4.3.4. Configuration des services téléphonique (fonctionnalités) dans Asterisk**

Après avoir installé les modules d'Asterisk, nous avons désactivé SIP s'il était présent pour activer uniquement PJSIP. Les fichiers de configuration par défaut (pjsip.conf, extensions.conf, users.conf) ont été renommés avec l'extension .origin, puis recréés vides afin de repartir sur une configuration propre et adaptée au projet.

##### **❖ Appel téléphonique**

Un appel téléphonique est l'action de connecter une personne utilisant un téléphone à son ou ses destinataires. Après avoir composé le numéro du destinataire sur le clavier de l'appareil émetteur, une sonnerie retenue chez ce dernier jusqu'à ce qu'il accepte l'appel. La conversation téléphonique débute alors, généralement par le mot « allô ». Sa configuration se fait principalement dans les fichiers **pjsip.conf** et **extensions.conf**



##### **Création des comptes utilisateurs :**

Pour créer des utilisateurs dans Asterisk, nous devons effectivement configurer le fichier **pjsip.conf**. Voici le contenu de ce fichier pour créer trois utilisateurs, avec les numéros respectifs 1000, 1001 et 1002, et un mot de passe (secret) passer.

```

root@ip-10-0-1-152: /etc/asterisk
GNU nano 7.2                                     pjsip.conf

[global]
type=global
user_agent=Asterisk
[presence]
type=publishation
event=presence
[transport-udp]
type=transport
protocol=udp
bind=0.0.0.0:5060
local_net=10.0.1.0/24
external_media_address=3.95.250.115
external_signaling_address=3.95.250.115

[transport-wss]
protocol=wss
bind=0.0.0.0

; =====
; ◊ Configuration de l'endpoint 1000
; =====
[1000]
type=endpoint
context=rtn
dissallow=all
allow-ulaw,alaw,g722,g726,h264,h265,vp8
asymmetric_rtp_codec=yes ;----- Permet de gérer des codecs asymétriques (ex: vidéo et audio différents)
transport=transport-udp
auth=1000
aors=1000
aors=1000
language=fr
callerid="Djiby" <1000>
allow_transfer=yes ;----- Autorise le transfert d'appel
call_group=1 ;----- Groupe d'appel SIP
max_video_streams=2 ;----- Nombre maximal de flux vidéo
; -----
; ◊ Gestion du NAT et RTP
; -----
disable_direct_media_on_nat=true ;----- Désactive le media direct si NAT détecté
direct_media=false ;----- Force le passage par Asterisk pour les flux RTP
media_address=3.95.250.115 ;----- Adresse publique du serveur pour les médias
rtp_symmetric=yes ;----- Permet d'assurer que les flux RTP reviennent à la même adresse
rewrite_contact=yes ;----- Réécrit l'adresse contact pour éviter les problèmes NAT
force_rport=yes ;----- Force l'utilisation du port source pour RTP
rtcp_mux=true ;----- Regroupe RTP et RTCP sur un seul port (utile pour WebRTC)

; =====
; ◊ Authentification de l'utilisateur 1000
; =====

; =====
; ◊ Authentification de l'utilisateur 1000
; =====
[1000]
type=auth
auth_type=userpass
username=1000
password=passer

; =====
; ◊ Gestion des AORs (Contacts SIP)
; =====
[1000]
type=aor
max_contacts=30 ;----- Permet à 30 appareils de s'enregistrer sur cet identifiant
qualify_frequency=30 ;----- Vérifie la connexion toutes les 30 secondes

; =====
; ◊ Configuration de l'endpoint 1001
; =====

```

Ceci est réalisé pour la création des comptes 1001 et 1002 dans le même fichier pjsip.conf.

 **Visualiser les utilisateurs :** Pour vérifier si les comptes utilisateurs sont bien enregistrés, nous lançons la console avec la commande **asterisk -rvvv** puis **pjsip list aors** et voici le résultat prouvant qu'ils sont bien enregistrés.

```

core debug is still on.
ip-10-0-1-152*CLI> pjsip list aors

    Aor: <Aor.....> <MaxContact>
=====
=====

    Aor: 1000                      30
    Aor: 1001                      30
    Aor: 1002                      30

Objects found: 3
ip-10-0-1-152*CLI> 

```

 **Configuration du Dialplan :** Nous allons donc configurer Asterisk de telle sorte que l'utilisateur 1000 puisse appeler le numéro 1001. Voici donc mon fichier **extensions.conf**

afficher avec de la commande :**sudo nano/etc/asterisk/extension.conf**

```

root@ip-10-0-1-152: ~
GNU nano 7.2                               /etc/asterisk/extensions.conf *
[general]
static=yes          ; Empêche la réécriture automatique du fichier extensions.conf
writeprotect=no    ; Autorise l'enregistrement des modifications via la CLI avec "dialplan save"
autofallthrough=yes ; Assure que les appels sont terminés proprement à la fin d'une extension
extenpatternmatchnew=yes ; Active l'algorithme optimisé pour la recherche rapide des extensions

[rtn]
exten => _1XXX,1,Dial(PJSIP/${EXTEN},30,tr)
exten => _1XXX,n,NoOp(DIALSTATUS=${DIALSTATUS} HANGUPCAUSE=${HANGUPCAUSE})

```

### ❖ Messagerie vocale (boite vocal)

La messagerie vocale est un répondeur/enregistreur permettant à un utilisateur interne du système Asterisk de disposer d'une boîte vocale. L'utilisateur peut la paramétrier selon ses préférences et l'activer en cas d'absence. Ainsi, l'appelant pourra laisser un message vocal.

Fichiers concernés : **voicemail.conf** : Configuration de la messagerie vocale et **extensions.conf** : Gestion du plan de numérotation et des règles d'acheminement des appels.

 **voicemail.conf** ; Dans le fichier voicemail.conf, nous avons d'abord défini, dans le contexte [general], le format d'enregistrement des messages vocaux ainsi que le codec utilisé. Ensuite, dans le contexte [default], nous avons spécifié les numéros de téléphone correspondant à chaque utilisateur, associés à leur nom respectif et au mot de passe qu'ils doivent composer pour écouter leurs messages sur leur boîte vocale.

```

GNU nano 7.2
/etc/asterisk/voicemail.conf

[general]
format=wav49|gsm|wav

[default]
1000 => 1234, Djiby
1001 => 1234, Seynabou
1002 => 1234, Amy
1003 => 1234, Moustapha

```

**extensions.conf** ; Dans le fichier extensions.conf, nous nous intéressons au contexte [rtn], où nous avons défini l'extension du plan de numérotation pour la boîte vocale. Nous y avons également spécifié le numéro à composer afin d'accéder à la messagerie vocale et d'écouter les messages enregistrés.

```

GNU nano 7.2
/etc/asterisk/extensions.conf

[general]
static=yes ; Empêche la réécriture automatique du fichier extcsions.conf
writeprotect=no ; Autorise l'enregistrement des modifications via la CLI avec "dialplan save"
Clear global vars=no ; Assure que les appels sont terminés proprement à la fin d'une extension
autofallthrough=yes ; Active l'algorithme optimisé pour la recherche rapide des extensions
extenpatternmatchnew=yes

[rtn]

exten => _1xxx,n,Dial(PJSIP/${EXTEN},20(${CALLERID(name)}))
exten => _1XXX,n,NoOp(DIALSTATUS=${DIALSTATUS} HANGUPCAUSE=${HANGUPCAUSE})
same => n,voicemail(${EXTEN}@default) ←
exten =>123,1,VoiceMailMain() ←

```

### ❖ Interception d'appels

L'interception d'appels permet de répondre à un appel destiné à un autre poste en appuyant sur \*8. User1 à un **call\_group** défini, et user2 à un **pickup\_group** correspondant. Si user1 ne répond pas, user2 peut composer l'extension **pickupexten** (définie dans **feature.conf**) pour intercepter l'appel. Cette fonctionnalité est configurée dans **Features.conf** et **pjsip.conf** et est utile en environnement de bureau.

**features.conf** ; on édite le fichier ; On édite le fichier features.conf et supprime le point-virgule à la fin de la ligne pickupexten=\*8 afin qu'Asterisk puisse charger cette configuration au démarrage.

Le numéro **\*8** pourra alors être composé pour intercepter un appel.

```

GNU nano 7.2                                         features.conf *
;
; Sample Call Features (transfer, monitor/mixmonitor, etc) configuration
;

; Note: From Asterisk 12 - All parking lot configuration is now done in res_parking.conf

[general]
;transferdigittimeout => 3      ; Number of seconds to wait between digits when transferring a call
;                               ; (default is 3 seconds). If the TRANSFER_EXTEN dialplan variable has been set
;                               ; on the channel of the user that is invoking the transfer feature, then
;                               ; this option is not used as the user is transferred directly to the extension
;                               ; specified by TRANSFER_EXTEN (the transfer context remains the context specified
;                               ; by TRANSFER_CONTEXT, if set, and otherwise the default context).
;                               ; to indicate an attended transfer is complete
;xfersound = beep          ; to indicate a failed transfer
;xferfailsound = beeperr    ; Configure the pickup extension. (default is *8)
pickupexten = *8           ; to indicate a successful pickup (default: no sound)
;pickupfailsound = beeperr  ; to indicate that the pickup failed (default: no sound)
;featuredigittimeout = 1000   ; Max time (ms) between digits for
;                               ; feature activation (default is 1000 ms)
;recordingfailsound = beeperr ; indicates that a one-touch monitor or one-touch mixmonitor feature failed
;                               ; to be applied to the call. (default: no sound)
;atxfernoanswertimeout = 15   ; Timeout for answer on attended transfer default is 15 seconds.
;atxferdropcall = no        ; If someone does an attended transfer, then hangs up before the transfer
;                               ; target answers, then by default, the system will try to call back the
;                               ; person that did the transfer. If this is set to "yes", the ringing
;                               ; transfer target is immediately transferred to the transferer

```

**pjsip.conf**: Dans le fichier pjsip.conf, les fonctions **call\_group** et **pickup\_group** permettent d'intercepter un appel destiné à un autre utilisateur lorsque son téléphone sonne.

```

allow_transfer=yes          ;----- Autorise le transfert d'appel
call_group=1                ;----- Groupe d'appel SIP
max_video_streams=2         ;----- Nombre maximal de flux vidéo
;
; ----- Gestion du NAT et RTP -----
;----- Désactive le media direct si NAT détecté
;----- Force le passage par Asterisk pour les flux RTP
;----- Adresse publique du serveur pour les médias
;----- Permet d'assurer que les flux RTP reviennent à la même adresse
;----- Réécrit l'adresse contact pour éviter les problèmes NAT
;----- Force l'utilisation du port source pour RTP
;----- Regroupe RTP et RTCP sur un seul port (utile pour WebRTC)

; ----- Authentification de l'utilisateur 1000 -----
; ----- Gestion des AORs (Contacts SIP) -----
[1000]
type=auth
auth_type=userpass
username=1000
password=passer

; ----- Configuration de l'endpoint 1001 -----
[1001]
type=endpoint
context=rtn
allow_subscribe=yes          ;----- Autorise l'abonnement aux événements (présence, BLF)
disallow=all
allow=ulaw,alaw,g722,g726,h264,h265,vp8
asymmetric_rtp_codec=yes     ;----- Permet de gérer des codecs asymétriques (ex: vidéo et audio différents)
transport=transport-udp
auth=1001
aors=1001
language=fr
callerid="Utilisateur 1001" <1001> ;----- Affiche "Utilisateur 1001" comme identifiant d'appel
allow_transfer=yes            ;----- Autorise le transfert d'appel
pickup_group=1                ;----- Groupe de récupération des appels (pickup group)
max_video_streams=2           ;----- Nombre maximal de flux vidéo

```

## ❖ Gestion de conférence téléphonique et vidéo conférence

Les conférences téléphoniques permettent à plusieurs participants de rejoindre une salle en composant son numéro. Tous peuvent échanger librement et entrer ou sortir à leur convenance. **ConfBridge** a remplacé l'ancien module **MeetMe** et offre des fonctionnalités avancées. La configuration se fait via **extensions.conf** et **confbridge.conf**.



**confbridge.conf** : Pour configurer une conférence téléphonique dans Asterisk avec **ConfBridge**, on doit éditer le fichier **/etc/asterisk/confbridge.conf**. Ce fichier permet de définir différents profils pour gérer les conférences et les utilisateurs qui y participent.

La configuration de ConfBridge repose sur trois types de profils définis dans **confbridge.conf**. Le profil de salle (type *bridge*) permet de fixer des paramètres comme le nombre maximal de participants. Le profil utilisateur (type *user*) attribue des options spécifiques à chaque participant, comme un mot de passe (pin) pour sécuriser l'accès ou le statut d'administrateur (admin yes/no) donnant des priviléges supplémentaires (expulsion de participants, gestion de la conférence). Il permet également d'activer une musique d'attente avant l'entrée dans la salle. Enfin, le profil de menu (type *menu*) configure les actions accessibles via les touches DTMF, comme la mise en sourdine, l'expulsion ou le réglage du volume des participants. Ces éléments permettent une gestion flexible et efficace des conférences téléphoniques dans Asterisk.

```

[general]
[admin_user]
type=user
pin=123
marked=yes
admin=yes
music_on_hold_when_empty=yes
announce_user_count=yes

[default_user]
type=user
pin=1234
wait_marked=yes
end_marked=yes
music_on_hold_when_empty=yes
announce_user_count=yes

[default_bridge]
type=bridge
max_members=10
language=fr

[sample_user_menu]
type=menu
*=playback_and_continue(conf-usermenu)
*1=toggle_mute
1=toggle_mute
*4=decrease_listening_volume
4=decrease_listening_volume
*6=increase_listening_volume
6=increase_listening_volume
*7=decrease_talking_volume
7=decrease_talking_volume
*8=leave_conference
8=leave_Conference
*9=increase_talking_volume
9=increase_talking_volume

[sample_admin_menu]
type=menu
*=playback_and_continue(conf-adminmenu)
*1=toggle_mute
1=toggle_mute
*2=admin_toggle_conference_lock ; only applied to admin users
2=admin_toggle_conference_lock ; only applied to admin users
*3=admin_kick_last ; only applied to admin users
3=admin_kick_last ; only applied to admin users
*4=decrease_listening_volume
4=decrease_listening_volume
*6=increase_listening_volume
6=increase_listening_volume
*7=decrease_talking_volume
7=decrease_talking_volume
*8=no_op
8=no_op
*9=increase_talking_volume
9=increase_talking_volume

```

□

 **extensions.conf:** Dans le fichier extensions.conf, nous allons définir le numéro à composer, soit pour Amin, soit pour un utilisateur, afin d'accéder à la conférence

Voici le fichier de configuration

```

-----conference-----
-----CONFERENCE - GUEST-----
exten => 666,1,Progress()                                     ; Donne une tonalité avant de commencer
exten => 666,2,Wait(1)                                       ; Attends 1 seconde
exten => 666,3,ConfBridge(1,default_bridge,default_user)    ; Rejoint la conférence en tant qu'invité

-----CONFERENCE - ADMIN-----
exten => 777,1,Progress()                                     ; Donne une tonalité avant de commencer
exten => 777,2,Wait(1)                                       ; Attends 1 seconde
exten => 777,3,ConfBridge(1,default_bridge,admin_user)       ; Rejoint la conférence en tant qu'administrateur

```

### ❖ Transfert d'appel aveugle et supervisé

Dans Asterisk, le transfert d'appel est configuré dans **Features.conf** et nécessite l'option **Tt** dans **extensions.conf**. Deux types de transfert sont possibles : le transfert aveugle (immédiat) en composant # suivi du numéro de destination, et le transfert supervisé, où l'appelant est mis en attente avec une musique en appuyant sur \* avant de prévenir le destinataire.

```
GNU nano 7.2                                         features.conf
[general]
pickupexten = *8 ; Permet de récupérer un appel en cours sur une autre extension
transferdigittimeout = 7 ; Temps d'attente pour saisir le numéro
[featuremap]
blindxfer => #    ; Transfert aveugle : transfère l'appel immédiatement à une autre extension
atxfer => *      ; Transfert assisté : permet de parler avant de transférer l'appel
```

### ❖ Parking-d'appels

Le parking d'appel permet de mettre un appel en attente et de le récupérer depuis un autre poste en composant un numéro spécifique. Dans **res\_parking.conf**, **parkext (700)** définit le numéro pour placer un appel en attente après un transfert, tandis que **parkpos (701-720)** désigne les numéros permettant de récupérer l'appel. Le contexte (parkedcalls) doit être inclus dans **extensions.conf** pour intégrer la gestion du stationnement, et le temps de stationnement (**70s**) fixe la durée maximale avant renvoi à l'appelant initial. Cette configuration optimise la gestion des appels en attente dans Asterisk.

```
[default]
parkext => 700          ; Default Parking Lot
; What extension to dial to park. (optional; if
; specified, extensions will be created for parkext and
; the whole range of parkpos)
;
; Note: Generated parking extensions cannot overlap.
; The only exception is if neither overlapping parkext
; is exclusive.

;parkext_exclusive=yes   ; Specify that the parkext created for this parking lot
; will only access this parking lot. (default is no)

parkpos => 701-720       ; What range of parking spaces to use - must be numeric
; Creates these spaces as extensions if parkext is set.
; Since this value is interpreted numerically, leading 0's
; will be ignored (so expect 00700-00720 to map to 700-720)

context => parkedcalls   ; Which context parked calls and the default park

;parkinghints = no        ; Add hints priorities automatically for parkpos
; extensions if parkext is set

parkingtime => 70          ; Number of seconds a call can be parked before returning

;comebacktoorigin = yes    ; Setting this option configures the behavior of call parking when the
; parked call times out (See the parkingtime option). The default value is 'yes'.
;
; 'yes' - When the parked call times out, attempt to send the call back to the peer
; that parked this call. This is done by saving off the name of the channel
; that parked the call. The call will return to the context 'park-dial' and
; an extension created based on the name of the channel that originally parked
; the call. This extension will be created automatically to do a Dial() to the
; device that originally parked the call for comebacktodialtime seconds. If the
; call is not answered, the call will proceed to the next priority (usually none
; unless you deliberately set up a catch-all second priority in the park-call
; context) in the dialplan for extension matching the peer name (same as how
; peer names are flattened into extensions when comebacktoorigin is 'no').
```

### **4.3.5. Test des fonctionnalités avec le serveur Asterisk**

Une fois que tous les fichiers de configuration pour les fonctionnalités téléphoniques d'Asterisk sont terminés, je procéderai immédiatement aux tests des fonctionnalités.

#### **❖ Appels téléphoniques**

Les logs dans la console Asterisk montrent que l'appel, initié par le contact 1000 vers le contact 1001, se déroule correctement, sans aucune erreur détectée. Asterisk détecte l'appel et, une fois le contact 1001 décroché, il établit la connexion en verrouillant les deux ponts dans le "simple bridge". Les communications vocales se passent sans problème, sans aucune anomalie relevée durant l'échange. Cela confirme que le système fonctionne normalement, assurant ainsi une qualité d'appel stable.

```
running under group asterisk
Connected to Asterisk 20.11.1 currently running on ip-10-0-1-152 (pid = 12921)
Parsing /etc/asterisk/logger.conf
Core debug is still 2.
-- Executing [1001@rtn:1] Answer("PJSIP/1000-00000000", "") in new stack
  > 0x72217802a490 -- Strict RTP learning after remote address set to: 41.82.170.44:6982
  > 0x72217802a490 -- Strict RTP qualifying stream type: audio
  > 0x72217802a490 -- Strict RTP switching source address to 41.82.170.44:64125
-- Executing [1001@rtn:2] Set("PJSIP/1000-00000000", "CALLERID(name)=Djiby") in new stack
-- Executing [1001@rtn:3] Dial("PJSIP/1000-00000000", "PJSIP/1001,60,tT") in new stack
-- Called PJSIP/1001
-- PJSIP/1001-00000001 is ringing
  > 0x72217802a490 -- Strict RTP learning complete - Locking on source address 41.82.170.44:64125
  > 0x72217806b620 -- Strict RTP learning after remote address set to: 192.168.215.134:54769
-- PJSIP/1001-00000001 answered PJSIP/1000-00000000
-- Channel PJSIP/1001-00000001 joined 'simple_bridge' basic-bridge <dd260864-ba22-4e80-aa9b-e780f3alf2a1>
  > 0x72217806b620 -- Strict RTP qualifying stream type: audio
-- Channel PJSIP/1000-00000000 joined 'simple_bridge' basic-bridge <dd260864-ba22-4e80-aa9b-e780f3alf2a1>
  > 0x72217806b620 -- Strict RTP switching source address to 41.82.170.44:60671
  > 0x72217806b620 -- Strict RTP learning complete - Locking on source address 41.82.170.44:60671
-- Channel PJSIP/1001-00000001 left 'simple_bridge' basic-bridge <dd260864-ba22-4e80-aa9b-e780f3alf2a1>
-- Channel PJSIP/1000-00000000 left 'simple_bridge' basic-bridge <dd260864-ba22-4e80-aa9b-e780f3alf2a1>
== Spawn extension (rtn, 1001, 3) exited non-zero on 'PJSIP/1000-00000000'
ip-10-0-1-152*CLI>
Disconnected from Asterisk server
```

#### **❖ Appel vidéo**

Les logs dans la console Asterisk montrent que l'appel, initié par le contact 1001 vers le contact 1003, se déroule parfaitement, sans aucune erreur détectée. Asterisk identifie rapidement l'appel et, une fois le contact 1003 décroché, il établit la connexion en verrouillant les deux ponts dans le "simple bridge". Les communications vocales et vidéo se déroulent sans accroc, offrant une qualité fluide et stable, sans aucune anomalie relevée tout au long de l'échange. Cela confirme que le système fonctionne de manière optimale, garantissant ainsi une expérience de communication claire et fiable.

```

-- Channel PJSIP/1001-00000007 left 'native_rtp' basic-bridge <bf6c15ab-5562-4791-a6eb-346006c34313>
== Spawn extension (rtn, 1001, 1) exited non-zero on 'PJSIP/1000-00000006'
-- Executing [1001@rtn:1] Dial("PJSIP/1003-00000008", "PJSIP/1001,20(Moustapha") in new stack
-- Called PJSIP/1001
-- PJSIP/1001-00000009 is ringing
> 0x716314074cf0 -- Strict RTP learning after remote address set to: 192.168.96.134:58838
-- PJSIP/1001-00000009 answered PJSIP/1003-00000008
> 0x716314018d50 -- Strict RTP learning after remote address set to: 192.168.96.134:4000
> 0x71631405e5e0 -- Strict RTP learning after remote address set to: 192.168.96.134:4002
-- Channel PJSIP/1001-00000009 joined 'simple_bridge' basic-bridge <dcbeb9a6-2201-43a7-a5a3-ea414edf56fa>
-- Channel PJSIP/1003-00000008 joined 'simple_bridge' basic-bridge <dcbeb9a6-2201-43a7-a5a3-ea414edf56fa>
> 0x716314074cf0 -- Strict RTP qualifying stream type: audio
> 0x716314074cf0 -- Strict RTP switching source address to 41.82.170.4:32391
> 0x716314018d50 -- Strict RTP qualifying stream type: audio
> 0x71631405e5e0 -- Strict RTP qualifying stream type: video
> 0x71631405e5e0 -- Strict RTP switching source address to 41.82.170.4:57190
> 0x716314018d50 -- Strict RTP switching source address to 41.82.170.4:5608
> 0x716314074cf0 -- Strict RTP learning complete - Locking on source address 41.82.170.4:32391
> 0x71631405e5e0 -- Strict RTP learning complete - Locking on source address 41.82.170.4:57190
> 0x716314018d50 -- Strict RTP learning complete - Locking on source address 41.82.170.4:5608
-- Channel PJSIP/1003-00000008 left 'simple_bridge' basic-bridge <dcbeb9a6-2201-43a7-a5a3-ea414edf56fa>
-- Channel PJSIP/1001-00000009 left 'simple_bridge' basic-bridge <dcbeb9a6-2201-43a7-a5a3-ea414edf56fa>
== Spawn extension (rtn, 1001, 1) exited non-zero on 'PJSIP/1003-00000008'
p-10-0-1-152*CLI> 
```

## ❖ Boit vocal

Les logs dans la console Asterisk montrent que l'appel, initié par le contact 1000 vers le contact 1003, se déroule normalement. Cependant, le contact 1003 ne décroche pas l'appel dans les 20 secondes. Après ce délai, Asterisk bascule automatiquement l'appel vers la boîte vocale, permettant à l'appelant de laisser un message vocal. Ce mécanisme assure que l'appelant peut enregistrer son message, même si le destinataire ne répond pas dans le délai imparti.

```

-- Executing [1000@rtn:1] Dial("PJSIP/1003-0000000a", "PJSIP/1000,20(Moustapha") in new stack
-- Called PJSIP/1000
-- PJSIP/1000-0000000b is ringing
-- Nobody picked up in 20000 ms
-- Executing [1000@rtn:2] NoOp("PJSIP/1003-0000000a", "DIALSTATUS=NOANSWER HANGUPCAUSE=0") in new stack
-- Executing [1000@rtn:3] VoiceMail("PJSIP/1003-0000000a", "1000@default") in new stack
> 0x716314074cf0 -- Strict RTP learning after remote address set to: 192.168.96.134:4004
-- <PJSIP/1003-0000000a> Playing 'vm-intro.ulaw' (language 'fr')
> 0x716314074cf0 -- Strict RTP qualifying stream type: audio
> 0x716314074cf0 -- Strict RTP switching source address to 41.82.170.4:12817
> 0x716314074cf0 -- Strict RTP learning complete - Locking on source address 41.82.170.4:12817
-- <PJSIP/1003-0000000a> Playing 'beep.ulaw' (language 'fr')
-- Recording the message
-- User ended message by pressing #
-- <PJSIP/1003-0000000a> Playing 'auth-thankyou.ulaw' (language 'fr')
-- Auto fallthrough, channel 'PJSIP/1003-0000000a' status is 'NOANSWER'
p-10-0-1-152*CLI> 
```

## ❖ Interception d'appel

Depuis la console d'Asterisk, lorsqu'on appelle le numéro 1000 à partir du numéro 1001, l'appel sonne sur le téléphone du numéro 1000. Ensuite, si l'appel est intercepté par le téléphone du numéro 1003, Asterisk confirme que l'interception fonctionne correctement, ce qui montre que la fonctionnalité d'interception d'appel est opérationnelle.

```

-- Executing [1000@rtn:1] Answer("PJSIP/1001-00000000", "") in new stack
> 0x707850074bf0 -- Strict RTP learning after remote address set to: 192.168.215.134:56731
-- Executing [1000@rtn:2] Set("PJSIP/1001-00000000", "CALLERID(name)=Utilisateur 1001") in new stack
-- Executing [1000@rtn:3] Dial("PJSIP/1001-00000000", "PJSIP/1000,60,tT") in new stack
-- Called PJSIP/1000
> 0x707850074bf0 -- Strict RTP qualifying stream type: audio
> 0x707850074bf0 -- Strict RTP switching source address to 41.82.170.44:37333
-- PJSIP/1000-00000001 is ringing
> 0x707850074bf0 -- Strict RTP learning complete - Locking on source address 41.82.170.44:37333
> 0x7078500c49d0 -- Strict RTP learning after remote address set to: 192.168.215.134:4000
-- PJSIP/1003-00000002 answered PJSIP/1001-00000000
-- Channel PJSIP/1003-00000002 joined 'simple_bridge' basic-bridge <8dc168c3-57c8-4430-a99f-3e75004fbcc4>
-- Channel PJSIP/1001-00000000 joined 'simple_bridge' basic-bridge <8dc168c3-57c8-4430-a99f-3e75004fbcc4>
> 0x7078500c49d0 -- Strict RTP learning after remote address set to: 192.168.215.134:4000
> 0x7078500c49d0 -- Strict RTP qualifying stream type: audio
> 0x7078500c49d0 -- Strict RTP switching source address to 41.82.170.44:64665
> 0x7078500c49d0 -- Strict RTP learning complete - Locking on source address 41.82.170.44:64665
-- Channel PJSIP/1001-00000000 left 'simple_bridge' basic-bridge <8dc168c3-57c8-4430-a99f-3e75004fbcc4>
Spawn extension (rtn, 1000, 3) exited non-zero on 'PJSIP/1001-00000000'
-- Channel PJSIP/1003-00000002 left 'simple_bridge' basic-bridge <8dc168c3-57c8-4430-a99f-3e75004fbcc4>
0-0-1-152*CLI> █

```

### ❖ Gestion de conférence

Le numéro 1000 a composé le 666, qui est le numéro défini pour les utilisateurs des conférences dans Asterisk. Ensuite, une voix lui demande de saisir le code d'accès, qu'il entre comme étant 1234. Après avoir validé ce code, on lui annonce d'attendre l'administrateur de la conférence.

L'administrateur, identifié par le numéro 777, se connecte à son tour. On lui demande également de saisir son code d'accès. Une fois que ce dernier est vérifié, on lui annonce qu'il n'y a qu'un seul participant dans cette conférence, avant que la conférence ne commence.

Cela montre que le système fonctionne parfaitement, comme vérifié dans la console Asterisk.

```

-- Executing [666@rtn:1] Progress("PJSIP/1000-0000000a", "") in new stack
-- Executing [666@rtn:2] Wait("PJSIP/1000-0000000a", "1") in new stack
> 0x70785001a920 -- Strict RTP learning after remote address set to: 41.82.170.44:30512
-- Executing [666@rtn:3] ConfBridge("PJSIP/1000-0000000a", "1,default_bridge,default_user") in new stack
-- <PJSIP/1000-0000000a> Playing 'conf-getpin.gsm' (language 'fr')
> 0x70785001a920 -- Strict RTP qualifying stream type: audio
> 0x70785001a920 -- Strict RTP switching source address to 41.82.170.44:47251
> 0x70785001a920 -- Strict RTP learning complete - Locking on source address 41.82.170.44:47251
-- Channel CBAnn/1-00000001;2 joined 'softmix' base-bridge <la2ee444-95e0-4b55-9de9-bed951c8c58e>
== Begin MixMonitor Recording CBRRec/1-00000000
-- Channel CBRRec/1-00000000 joined 'softmix' base-bridge <la2ee444-95e0-4b55-9de9-bed951c8c58e>
-- <PJSIP/1000-0000000a> Playing 'conf-waitforleader.gsm' (language 'fr')
-- Started music on hold, class 'default', on channel 'PJSIP/1000-0000000a'
-- Channel PJSIP/1000-0000000a joined 'softmix' base-bridge <la2ee444-95e0-4b55-9de9-bed951c8c58e>
-- <CBAnn/1-00000001;1> Playing 'confbridge-join.gsm' (language 'fr')
-- Executing [777@rtn:1] Progress("PJSIP/1003-0000000b", "") in new stack
-- Executing [777@rtn:2] Wait("PJSIP/1003-0000000b", "1") in new stack
> 0x70785007a330 -- Strict RTP learning after remote address set to: 192.168.215.134:4004
> 0x70785007a330 -- Strict RTP qualifying stream type: audio
> 0x70785007a330 -- Strict RTP switching source address to 41.82.170.44:43529
> 0x70785007a330 -- Strict RTP learning after remote address set to: 192.168.215.134:4004
-- Executing [777@rtn:3] ConfBridge("PJSIP/1003-0000000b", "1,default_bridge,admin_user") in new stack
-- <PJSIP/1003-0000000b> Playing 'conf-getpin.gsm' (language 'fr')
> 0x70785007a330 -- Strict RTP learning after remote address set to: 192.168.215.134:4004
> 0x70785007a330 -- Strict RTP learning complete - Locking on source address 41.82.170.44:43529
-- Stopped music on hold on PJSIP/1000-0000000a
-- <PJSIP/1003-0000000b> Playing 'conf-onlyone.gsm' (language 'fr')
-- <CBAnn/1-00000001;1> Playing 'confbridge-conf-begin.gsm' (language 'fr')
-- Channel PJSIP/1003-0000000b joined 'softmix' base-bridge <la2ee444-95e0-4b55-9de9-bed951c8c58e>
-- <CBAnn/1-00000001;1> Playing 'confbridge-join.gsm' (language 'fr')
-- Channel PJSIP/1003-0000000b left 'softmix' base-bridge <la2ee444-95e0-4b55-9de9-bed951c8c58e>
-- Channel PJSIP/1000-0000000a left 'softmix' base-bridge <la2ee444-95e0-4b55-9de9-bed951c8c58e>
-- <PJSIP/1000-0000000a> Playing 'conf-kicked.gsm' (language 'fr')
-- <CBAnn/1-00000001;1> Playing 'confbridge-leave.gsm' (language 'fr')
-- <CBAnn/1-00000001;1> Playing 'confbridge-leave.gsm' (language 'fr')
-- Channel CBRRec/1-00000000 left 'softmix' base-bridge <la2ee444-95e0-4b55-9de9-bed951c8c58e>
== MixMonitor close filestream (mixed)
== End MixMonitor Recording CBRRec/1-00000000
-- Channel CBAnn/1-00000001;2 left 'softmix' base-bridge <la2ee444-95e0-4b55-9de9-bed951c8c58e>
-- Auto_fallback_channel 'PJSIP/1000-00000001' etatut is 'INVISIBLE'

```

### ❖ Transfert d'appel aveugle et supervisé

Le numéro 1001 appelle le numéro 1000. Une fois que le numéro 1000 décroche, il demande à 1001 de saisir le numéro auquel il souhaite transférer l'appel, par exemple, le numéro 1003.

Le numéro 1000 appuie sur la touche de transfert et compose le numéro 1003. Cette action est confirmée dans la console Asterisk.

```

src/debug is still 2.
-- Executing [1000@rtn:1] Answer("PJSIP/1001-0000000c", "") in new stack
> 0x70785001a920 -- Strict RTP learning after remote address set to: 192.168.215.134:62655
-- Executing [1000@rtn:2] Set("PJSIP/1001-0000000c", "CALLERID(name)=Utilisateur 1001") in new stack
-- Executing [1000@rtn:3] Dial("PJSIP/1001-0000000c", "PJSIP/1000,60,tT") in new stack
-- Called PJSIP/1000
-- PJSIP/1000-0000000d is ringing
> 0x70785001a920 -- Strict RTP qualifying stream type: audio
> 0x70785001a920 -- Strict RTP switching source address to 41.82.170.44:64164
> 0x70785001a920 -- Strict RTP learning complete - Locking on source address 41.82.170.44:64164
> 0x70785007a330 -- Strict RTP learning after remote address set to: 41.82.170.44:51546
-- PJSIP/1000-0000000d answered PJSIP/1001-0000000c
-- Channel PJSIP/1000-0000000d joined 'simple_bridge' basic-bridge <f2ed4494-1a69-49d9-a586-5f17f6bad318>
-- Channel PJSIP/1001-0000000c joined 'simple_bridge' basic-bridge <f2ed4494-1a69-49d9-a586-5f17f6bad318>
> 0x70785007a330 -- Strict RTP qualifying stream type: audio
> 0x70785007a330 -- Strict RTP switching source address to 41.82.170.44:44826
-- Channel PJSIP/1000-0000000d: Started DTMF blind transfer.
-- Started music on hold, class 'default', on channel 'PJSIP/1001-0000000c'
-- <PJSIP/1000-0000000d> Playing 'pbx-transfer.gsm' (language 'fr')
> 0x70785007a330 -- Strict RTP learning complete - Locking on source address 41.82.170.44:44826
-- Channel PJSIP/1000-0000000d left 'simple_bridge' basic-bridge <f2ed4494-1a69-49d9-a586-5f17f6bad318>
-- Stopped music on hold on PJSIP/1001-0000000c
-- Channel PJSIP/1001-0000000c left 'simple_bridge' basic-bridge <f2ed4494-1a69-49d9-a586-5f17f6bad318>
-- Executing [1003@rtn:1] Answer("PJSIP/1001-0000000c", "") in new stack
-- Executing [1003@rtn:2] Set("PJSIP/1001-0000000c", "CALLERID(name)=Utilisateur 1001") in new stack
-- Executing [1003@rtn:3] Dial("PJSIP/1001-0000000c", "PJSIP/1003,60,tT") in new stack
-- Called PJSIP/1003
-- PJSIP/1003-0000000e is ringing
-- PJSIP/1003-0000000e is ringing
> 0x70785007a330 -- Strict RTP learning after remote address set to: 192.168.215.134:4006
-- PJSIP/1003-0000000e answered PJSIP/1001-0000000c
-- Channel PJSIP/1003-0000000e joined 'simple_bridge' basic-bridge <8e358736-028b-4aae-b7e6-9a15273488a0>
-- Channel PJSIP/1001-0000000c joined 'simple_bridge' basic-bridge <8e358736-028b-4aae-b7e6-9a15273488a0>
> 0x70785007a330 -- Strict RTP qualifying stream type: audio
> 0x70785007a330 -- Strict RTP switching source address to 41.82.170.44:33379
> 0x70785007a330 -- Strict RTP learning complete - Locking on source address 41.82.170.44:33379
-- Channel PJSIP/1001-0000000c left 'simple_bridge' basic-bridge <8e358736-028b-4aae-b7e6-9a15273488a0>
-- Spawn extension (rtn, 1003, 3) exited non-zero on 'PJSIP/1001-0000000c'
-- Channel PJSIP/1003-0000000e left 'simple_bridge' basic-bridge <8e358736-028b-4aae-b7e6-9a15273488a0>

```

## ❖ Parking-d'appels

Pour le parking d'appels, le numéro 1003 appelle le 1001. Ensuite, 1001 décroche et garde l'appel dans le parking en appuyant sur la touche 700. Une voix lui annonce alors le numéro de récupération de l'appel dans le parking, qui est le 701. Ensuite, le numéro 1000 compose ce numéro pour récupérer l'appel, ce qui a été enregistré et visualisé par Asterisk.

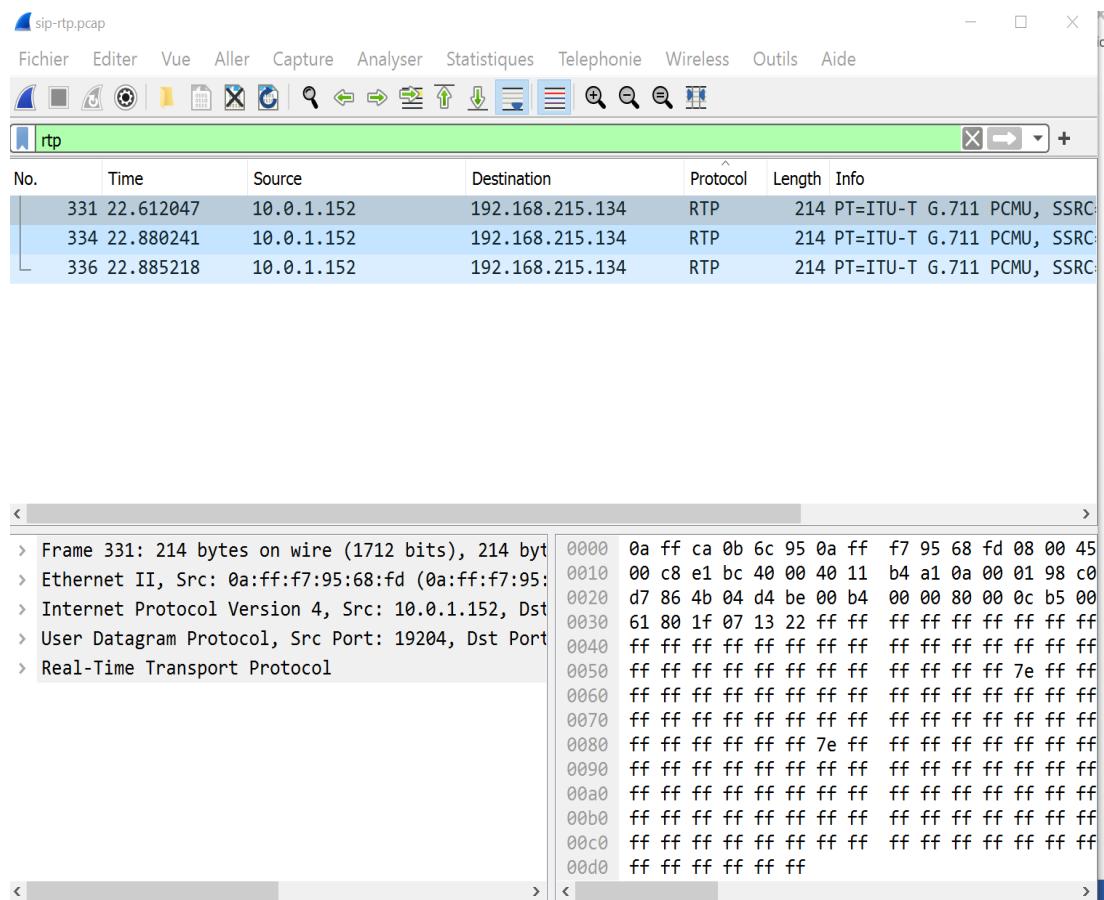
```

-- Executing [1001@rtn:1] Answer("PJSIP/1003-00000013", "") in new stack
> 0x70784002d020 -- Strict RTP learning after remote address set to: 192.168.215.134:4010
-- Executing [1001@rtn:2] Set("PJSIP/1003-00000013", "CALLERID(name)=Moustapha") in new stack
-- Executing [1001@rtn:3] Dial("PJSIP/1003-00000013", "PJSIP/1001,60,tT") in new stack
-- Called PJSIP/1001
> 0x70784002d020 -- Strict RTP learning after remote address set to: 192.168.215.134:4010
> 0x70784002d020 -- Strict RTP qualifying stream type: audio
> 0x70784002d020 -- Strict RTP switching source address to 41.82.170.44:43073
-- PJSIP/1001-00000014 is ringing
> 0x70784001b000 -- Strict RTP learning after remote address set to: 192.168.215.134:65182
-- PJSIP/1001-00000014 answered PJSIP/1003-00000013
-- Channel PJSIP/1001-00000014 joined 'simple_bridge' basic-bridge <b6074a94-918b-4081-a8c1-fd6aa727fafafa>
-- Channel PJSIP/1003-00000013 joined 'simple_bridge' basic-bridge <b6074a94-918b-4081-a8c1-fd6aa727fafafa>
> 0x70784001b000 -- Strict RTP qualifying stream type: audio
> 0x70784001b000 -- Strict RTP switching source address to 41.82.170.44:11331
> 0x70784002d020 -- Strict RTP learning complete - Locking on source address 41.82.170.44:43073
> 0x70784001b000 -- Strict RTP learning complete - Locking on source address 41.82.170.44:11331
> 0x70784001b000 -- Strict RTP learning after remote address set to: 192.168.215.134:65182
-- Started music on hold, class 'default', on channel 'PJSIP/1003-00000013'
-- Executing [700@rtn:1] Park("PJSIP/1001-00000015", "") in new stack
> 0x7078400e7100 -- Strict RTP learning after remote address set to: 192.168.215.134:65193
> 0x7078400e7100 -- Strict RTP qualifying stream type: audio
> 0x7078400e7100 -- Strict RTP switching source address to 41.82.170.44:63882
-- Parking 'PJSIP/1001-00000015' in 'default' at space 701
-- Channel PJSIP/1001-00000015 joined 'holding_bridge' parking-bridge <651082bf-7305-457f-999f-3591936167d3>
-- <PJSIP/1001-00000015> Playing 'digits/7.gsm' (language 'fr')
-- <PJSIP/1001-00000015> Playing 'digits/0.gsm' (language 'fr')
-- <PJSIP/1001-00000015> Playing 'digits/1.gsm' (language 'fr')
-- Started music on hold, class 'default', on channel 'PJSIP/1001-00000015'
> 0x7078400e7100 -- Strict RTP learning complete - Locking on source address 41.82.170.44:63882
-- Executing [701@rtn:1] ParkedCall("PJSIP/1001-00000016", "default,701") in new stack
> 0x707840069c80 -- Strict RTP learning after remote address set to: 41.82.170.44:31010
-- Channel PJSIP/1001-00000015 left 'holding_bridge' parking-bridge <651082bf-7305-457f-999f-3591936167d3>
-- Stopped music on hold on PJSIP/1001-00000015
-- Channel PJSIP/1001-00000015 joined 'simple_bridge' basic-bridge <16d20b4c-de09-46c9-9c86-3759be850f18>
-- Channel PJSIP/1000-00000016 joined 'simple_bridge' basic-bridge <16d20b4c-de09-46c9-9c86-3759be850f18>
> Bridge 16d20b4c-de09-46c9-9c86-3759be850f18: switching from simple_bridge technology to native_rtp
> Locally RTP bridged 'PJSIP/1000-00000016' and 'PJSIP/1001-00000015' in stack
> 0x707840069c80 -- Strict RTP qualifying stream type: audio
> 0x707840069c80 -- Strict RTP switching source address to 41.82.170.44:45780
> 0x707840069c80 -- Strict RTP learning complete - Locking on source address 41.82.170.44:45780

```

#### **4.3.6. Evaluation de la qualité de service (QoS)**

Pour évaluer la qualité de service, nous avons utilisé WinSCP pour télécharger le fichier de capture de paquets généré par Asterisk lors d'un appel. Ensuite, nous avons analysé les paquets RTP avec Wireshark. Cette analyse nous permet d'identifier les éléments pouvant affecter la qualité du service. Voici les paquets rtp filtré dans wireshark :



Une fois les paquets RTP filtrés, nous avons accédé aux journaux téléphoniques dans Wireshark en sélectionnant RTP, puis en choisissant Analyse du flux de paquets RTP. Cela nous donne accès à une fenêtre affichant une visualisation détaillée des flux RTP, incluant des informations telles que le délai, la gigue et la perte de paquets, qui permettent d'évaluer la qualité de l'appel.

<b>Stream</b>	<b>Paquet</b>	<b>Séquence</b>	<b>Delta(ms)</b>	<b>Gigue(ms)</b>	<b>Déviation</b>	<b>Bandes</b>
10.0.1.152:19204 → 192.168.215.134:54462	331	3253	0.000000	0.000000	0.000000	
<b>SSRC</b> 0x1f071322	334	3254	268.194000	15.512125	-248.194000	
<b>Max Delta</b> 268.194000 ms @ 334	336	3255	4.977000	15.481555	-233.171000	
<b>Max Jitter</b> 15.512125 ms						
<b>Mean Jitter</b> 15.496840 ms						
<b>Max Skew</b> -248.194000 ms						
<b>RTP Packets</b> 3						
<b>Expected</b> 3						
<b>Lost</b> 0 (0.00 %)						
<b>Seq Errs</b> 0						
<b>Start at</b> 22.612047 s @ 331						
<b>Duration</b> 0.27 s						
<b>Clock Drift</b> -243 ms						
<b>Freq Drift</b> 100 Hz (-88.82 %)						

L'analyse détaillée des éléments fournis et leur interprétation pour l'évaluation de la qualité de service: Le trafic RTP analysé montre une transmission depuis l'adresse 10.0.1.152:19204 vers 192.168.215.134:54462, avec une durée très courte de 0,27 s. La qualité du réseau est bonne : aucun paquet perdu, pas d'erreurs de séquence, et un jitter moyen de 15,496 ms, considéré comme acceptable pour la VoIP. L'analyse de ces différents éléments nous permet de conclure que les résultats obtenus sont acceptables, compte tenu des conditions dans lesquelles les tests ont été effectués. En effet, la qualité de la connexion Internet était limitée, car nous utilisions le partage de connexion mobile au lieu d'un réseau Wi-Fi stable. De plus, l'accès restreint aux ressources AWS, notamment les instances EC2 avec un espace de stockage limité, a également pu impacter la performance du système. Malgré ces contraintes, l'absence de perte de paquets et le niveau de gigue relativement bas indiquent que la qualité du flux RTP reste globalement correcte dans ce contexte.



## **CONCLUSION GENERALE**

Au terme de ce travail, nous avons étudié l'intégration d'une solution VoIP sur une infrastructure Cloud, en mettant en lumière les apports respectifs de ces deux technologies et leur complémentarité dans la mise en place d'un système de communication moderne, performant et économique.

La VoIP, en tant que technologie de téléphonie sur IP, offre une alternative puissante aux systèmes traditionnels, en réduisant significativement les coûts de communication et en proposant une flexibilité d'usage accrue. Cependant, elle soulève plusieurs défis techniques, notamment liés à la gestion de la bande passante, à la latence, à la gigue et à la qualité du signal audio. Une bonne maîtrise des protocoles et des composants VoIP est donc indispensable pour garantir une qualité de service optimale.

Parallèlement, le Cloud Computing s'est révélé être un levier stratégique pour le déploiement et la gestion d'une infrastructure VoIP évolutive. Grâce à la virtualisation, à la scalabilité et à la disponibilité élevée des ressources, le Cloud permet une gestion souple des services tout en optimisant les coûts d'infrastructure. Toutefois, cette solution impose une attention particulière aux questions de sécurité, de confidentialité des données et de dépendance vis-à-vis des fournisseurs.

Le choix d'Astérisk comme serveur VoIP s'est justifié par sa robustesse, son adaptabilité et sa large communauté open source. Quant à AWS, il a été sélectionné pour sa fiabilité, ses performances et la richesse de ses services adaptés à l'hébergement de solutions critiques. L'association de ces deux outils a permis de déployer une architecture téléphonique capable de répondre efficacement aux besoins d'une organisation.

L'implémentation pratique a couvert la configuration des services cloud, l'installation et la personnalisation du serveur Astérisk, ainsi que l'intégration de fonctionnalités telles que la gestion des appels, la messagerie vocale, les conférences et les messages audio personnalisés. Des tests rigoureux ont permis de valider le bon fonctionnement du système dans divers contextes. Enfin, l'analyse de la qualité de service (QoS) a mis en évidence les impacts directs des conditions réseau sur les performances de la VoIP. Des pistes d'optimisation ont été identifiées pour améliorer la fluidité des communications et renforcer la fiabilité du système.

Il apparaît pertinent d'envisager l'intégration plus poussée de la VoIP avec le réseau téléphonique commuté (RTC), afin d'assurer une interopérabilité complète avec les infrastructures traditionnelles encore en usage. Une telle interconnexion permettrait de joindre tout type de correspondant, quelle que soit la technologie utilisée, renforçant ainsi la

continuité des communications. Par ailleurs, les plateformes cloud comme AWS offrent désormais des services d'intelligence artificielle avancés (Amazon Transcribe, Amazon Translate, Amazon Chime SDK) qui ouvrent la voie à des applications innovantes, notamment dans le cadre de réunions internationales entre chefs d'État ne parlant pas la même langue. Grâce à la transcription en temps réel, la traduction automatique et la synthèse vocale, il devient possible d'organiser des échanges multilingues sécurisés, fluides et instantanés. Cette convergence entre VoIP, Cloud et IA pourrait transformer profondément les modes de communication institutionnelle à l'échelle mondiale

## **BIBLIOGRAPHIES**

[3]- Rebha Bouzaida « **Étude et Mise en place d'une Solution VOIP Sécurisée** » Mémoire de Master à L'Ecole Supérieure d'Informatique des Télécommunications à Casablanca, 2011.

[4]- Didi Souhila, Guerriche meryem « **La Téléphonie sur IP** », (ToIP) Mémoire de Licence en Informatique a l'université Abou Bekr Belkaid en Algérie, 2014.

[5]- Mourad el allia « **développement d'un environnement de communication multicast (voix et vidéo) sur internet** », Mémoire de master à l'université de Montréal au Québec en 2002.

[11]- Yannick Yani Kalomba «**Etude et mise au point d'un système de communication VOIP**» Mémoire d'un diplôme d'ingénieur à l'université Protestante de Lubumbashi, 2009

[13]-Mohamed Taib Benisse, «**transmission média sur les réseaux ip en utilisant les protocoles sip et iax**» Mémoire d'un diplôme d'ingénieur à l'école de technologie supérieure des télécommunications au Québec, 2009.

[16] -Module INF - 180-IT-Virtualisation-Support de Cours Michel Mestrallet -130409 pages : p26.p27.

[22]-Alber Matakano «**Approche du design et implémentation d'un lan hiérarchique redondant pour la haute disponibilité d'une infrastructure réseau. Cas du réseau de la drkat / Lubumbashi**» Mémoire de licence en système et réseau à l'université Méthodiste au Katanga /Mulungwishi au Congo, 2015.

## **WEBOGRAPHIES**

[1]- <https://www.redhat.com/fr/topics/virtualization> Consulté le 03/08/2024

[2]- <file:///C:/Users/pc/Downloads/0227-formation-reseau-voip.pdf> Consulté le 03/02/2024

[10]- <https://www.testeur-voip.com/technologie-voip-> Consulté le 06/03/2024

[12]- <https://www.testeur-voip.com/technologie-voip-explication.php> Consulté le 20/03/2024

[14]- <https://www.ibm.com/fr-fr/topics/cloud-architecture> Consulté le 08/04/2024

[15] - <https://aws.amazon.com/fr/what-is/virtualization/> Consulté le 11/05/2024

[17] - <https://www.oracle.com/sn/cloud/what-is-cloud-computing/> Consulté le 14/05/2024

[18]-<https://mysaas.fr/2010/10/04/private-cloud-public-cloud-> Consulté le 12/03/2024

[19]- <https://www.oracle.com/fr/cloud/definition-cloud-prive/> Consulté le 13/04/2024

[20]- <https://www.oracle.com/fr/cloud/definition-cloud-hybride/> Consulté le 13/04/2024

[21] -<https://cloud.google.com/learn/advantages-of-cloud> Consulté le 15/05/2024

[23]- <https://www.ninjaone.com/fr/blog/types-de-sauvegarde-> Consulté le 17/06/2024

[24]- <https://www.weodeo.com/blog-materiel/quest-ce> Consulté le 20/08/2024

- [25] -<https://www.appviewx.com/education-center/load-balancer> Consulté le 20/11/2024
- [26] -<https://www.asterisk.org/> Consulté le 28/12/2025
- [27]- <https://freeswitch.com/> Consulté le 13/01/2025
- [28]- <https://www.kamailio.org/> Consulté le 24/01/2025
- [29]- <https://www.opensips.org/> Consulté le 28/01/2025
- [30]- <https://www.resiprocate.org/> Consulté le 27/02/2025
- [31]- <https://vegastack.com/tutorials/how-to-in> Consulté le 05/02/2025
- [32]- <https://hotkey404.com/installing-asterisk-20-from> Consulté le 10/02/2025
- [33]- <https://doc.ubuntu-fr.org/asterisk?do> Consulté le 17/02/2025
- [35]- <https://fr.scribd.com/document/636712333/Installation-> Consulté le 26/02/2025
- [36]- <https://www.wireshark.org/download.html> Consulté le 05/03/2025
- [37]- <https://aws.amazon.com/> Consulté le 05/03/2025
- [38]- <http://downloads.asterisk.org /asterisk/asterisk-20> Consulté le 04/04/2025
- [39]- <https://ressourcesinformatiques.com/article.php?article=612i> Consulté le 19/04/2025
- [40]- <https://berenger-benam.over-blog.com/2023/06/> Consulté le 17/04/2025
- [41]- <https://fr.linux-console.net/?p=22256> Consulté le 05/05/2025
- [42]- <https://sip.goffinet.org/asterisk/uc-asterisk-intermediaire/> Consulté le 07/05/2025
- [43]- <https://community.asterisk.org/t/pjsip-with-authentication> Consulté le 09/05/2025
- [44]- <https://zadarma.com/fr/support/instructions/asteriskpjsip/> Consulté le 20/05/2025
- [45]- [https://aws.amazon.com/fr/what-is/interoperability/?utm\\_source](https://aws.amazon.com/fr/what-is/interoperability/?utm_source) Consulté le 29/05/2025
- [45]- <https://www.cio-online.com/actualites/lire-la-complexite> Consulté le 02/06/2025
- [45]- <https://www.docaufutur.fr/de-linteroperabilite> Consulté le 05/06/2025 .
- [45]- <https://blogs.manageengine.com/fr/2023/08/07/interoperabilite> Consulté le 12/06/2025
- [46]- <https://aws.amazon.com/fr/connect/> Consulté le 14/06/2025
- [47]- <https://azure.microsoft.com/en-us/products/> Consulté le 20/06/2025
- [48]- <https://cloud.google.com/apis/docs/overview?hl> Consulté le 28/06/2025
- [49]- <https://www.asterisk.org/> consulté le 07/04/2025
- [50]- <https://www.freepbx.org/> consulté le 07/04/2025

[51]- <https://www.fusionpbx.com/> consulté le 08/04/2025

[52]- <https://www.issabel.org/> consulté le 11/04/2025

[53]- <https://www.opensips.org/> consulté le 12/04/2025

[54]- Cisco : « Understanding Packet Loss »  
<https://www.cisco.com/c/en/us/support/docs/voice/voice-quality/7409-packet-loss.html> Consulté le 10/04/2025

[55]– ITU-T G.114 – « One-way transmission time » (section sur la gigue en VoIP)  
<https://www.itu.int/rec/T-REC-G.114> consulté le 12/04/2025

[56]– ITU-T P.800 – « Methods for subjective determination of transmission quality »  
<https://www.itu.int/rec/T-REC-P.800> consulté le 17/05/2025

[57]– ITU-T G.114 – « One-way transmission time » (recommandations de latence pour la voix)  
<https://www.itu.int/rec/T-REC-G.114> consulté le 29/05/2025