DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

TEMA: IMPLEMENTACIÓN Y ANÁLISIS DE UN SISTEMA VOIP MEDIANTE CLOUD COMPUTING CON INTEGRACIÓN PARA WEBRTC

AUTOR: ARMIJOS MALDONADO TIMOTE ANDRÉS

DIRECTOR: ING. ACOSTA BUENAÑO FREDDY ROBERTO

SANGOLQUÍ

2017

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

CERTIFICACIÓN

Certifico que el presente trabajo de titulación, **"IMPLEMENTACIÓN Y ANÁLISIS DE UN SISTEMA VOIP MEDIANTE CLOUD COMPUTING CON INTEGRACIÓN PARA WEBRTC"**, realizado por el señor **ARMIJOS MALDONADO TIMOTE ANDRÉS**, ha sido revisado en su totalidad y analizado por el software anti-plagio, el mismo cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas – ESPE, por lo tanto me permito acreditarlo y autorizar el señor **ARMIJOS MALDONADO TIMOTE ANDRÉS** para que lo sustente públicamente.

Sangolquí, 18 de Enero del 2017

Ing. Acosta Buenaño Freddy Roberto
DIRECTOR

ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTORÍA DE RESPONSABILIDAD

Yo, **ARMIJOS MALDONADO TIMOTE ANDRÉS,** con cedula de identidad N° 1103746531 declaro que este trabajo de titulación, **"IMPLEMENTACIÓN Y ANÁLISIS DE UN SISTEMA VOIP MEDIANTE CLOUD COMPUTING CON INTEGRACIÓN PARA WEBRTC"**, ha sido desarrollado considerando los métodos de investigación existentes, así como también se ha respetado los derechos intelectuales de terceros considerándose en las citas bibliográficas.

Consecuentemente declaro que este trabajo es de mi autoría, en virtud de ello me declaro responsable del contenido, veracidad y alcance de la investigación mencionada

Sangolquí, 18 de Enero del 2017

Armijos Maldonado Timote Andrés

CC. 1103746531

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTORIZACIÓN

Yo, **ARMIJOS MALDONADO TIMOTE ANDRÉS,** autorizo a la Universidad de las Fuerzas Armadas – ESPE publicar en la biblioteca Virtual de la institución el presente trabajo de titulación: **"IMPLEMENTACIÓN Y ANÁLISIS DE UN SISTEMA VOIP MEDIANTE CLOUD COMPUTING CON INTEGRACIÓN PARA WEBRTC"** cuyo contenido, ideas y criterios son de mi autoría y responsabilidad.

Sangolquí, 18 de Enero del 2017

Armijos Maldonado Timote Andrés

CC. 1103746531

DEDICATORIA

A mi madre con todo mi corazón.

A mi hijo para que algún día sea mejor ser humano que yo.

F.S.

Timote A.

AGRADECIMIENTO

Agradezco a mi madre, por ser la brújula de mi vida.
A mi hermano por ser un ejemplo a seguir.

Ing. Freddy Acosta gracias por guiarme con sus conocimientos para la culminación de este proyecto.

A todas las personas con las que compartí mi vida en las aulas de nuestra querida universidad, en especial a los que se convirtieron en mis amigos.

Gracias por todo.

Timote A.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

Cloud computing es el nuevo paradigma para el alojamiento y la prestación de servicios a través de Internet. La nube es atractiva para las empresas, ya que elimina la inversión en infraestructura y permite aumentar los recursos sólo cuando hay un aumento en la demanda de servicios. Esta tesis implementa y evalúa un sistema VoIP en la nube, que sea compatible con la tecnología de Web Real-Time Communication (WebRTC). Se presenta el estado del arte de la computación en la nube, destacando sus conceptos clave, se mide las prestaciones de un servidor de Asterisk en diferentes escenarios y se presenta los resultados de las pruebas realizadas. La metodología se compone de dos escenarios de prueba: la prueba 1 implica generar varias llamadas con diferentes combinaciones de velocidad de descarga y subida. Se monitorea la utilización de la CPU y el uso de la red en el servidor. El propósito de la prueba 2 es obtener el número máximo de llamadas activas simultáneamente. Al analizar los resultados de las simulaciones realizadas, el servidor VoIP en la nube mostró total compatibilidad con la tecnología WebRTC y que el número máximo de llamadas simultaneas es 100 en una instancia t2.micro.

PALABRAS CLAVE:

- **CLOUD COMPUTING**
- **WEBRTC**
- **VOIP**
- **ESCALAMIENTO**

ABSTRACT

Cloud computing is the new paradigm for hosting and delivering services over the Internet. The cloud is attractive to enterprises as it eliminates investments in infrastructure and allows to increase resources only when there is a rise in service demand. A VoIP system is implemented on the cloud, with support for Web Real-Time Communication (WebRTC) technology. This work presents the state-of-the-art of cloud computing, highlighting its key concepts, measures the performances of an Asterisk's server on different scenarios and presents the results of conducted tests. The methodology uses two test scenarios: test 1 implies generating several calls with different combinations of download and upload speed. Monitored parameters include CPU utilization and the network usage. In test 2 the purpose is to obtain the maximum number of active calls simultaneously. By analyzing test outcomes of the performed simulations, the cloud Voip server has total compatibly with WebRTC technology and the maximum number of active calls is 100 with a t2.micro instance.

KEYWORDS:

- **CLOUD COMPUTING**
- **WEBRTC**
- **VOIP**
- **SCALING**

CAPÍTULO I

INTRODUCCIÓN

1.1 Antecedentes

La comunicación en tiempo real (RTC por sus siglas en inglés) es una parte importante en la convergencia de las redes y de las telecomunicaciones. La comunicación interactiva, la vídeo conferencia y la mensajería instantánea son servicios y aplicaciones de la red ampliamente utilizados hoy en día. Estos servicios cambiaron y mejoraron la manera en que nos comunicamos, afectando muchos aspectos de nuestras vidas: el trabajo, la educación y las relaciones interpersonales.

SIP (Session Initiation Protocol) (Znaty, Dauphin, & Geldwerth, 2005) es un protocolo de señalización para conferencia, telefonía, presencia, notificación de eventos y mensajería instantánea a través de Internet. Es un protocolo que ha estandarizado nuevos mecanismos de control para la transformación de las redes IP en plataformas para la comunicación en tiempo real. La aceptación de SIP por la comunidad de Internet ha impulsado el proceso de convergencia en las redes, donde ahora es posible integrar y mesclar diferentes tipos de servicios en un nuevo ambiente de comunicación (Campbell & Sparks, 2001), (Silvana & Schulzrinne, 2008). Esto ha sido posible también gracias a la convergencia de los teléfonos que permiten el uso de muchas aplicaciones web.

Sin embargo la web todavía no soportaba servicios con una comunicación en tiempo real. Por tal razón se estudiaba la posibilidad de la integración de los servicios SIP en tiempo real con la web. Mediante tal integración se esperaba crear un nuevo ambiente para la comunicación, en donde sea posible desarrollar nuevas características y crear un gran número de servicios multimedia.

Se desarrollaron diferentes soluciones a lo largo de los años para tal integración. La primera generación de un servicio de comunicación SIP/Web fue posible mediante el uso de tecnología privada sea en el lado del cliente o en el lado del servidor. La similitud entre SIP y HTTP llevaron a la reutilización de muchas características del protocolo más utilizado en la web dentro de SIP, produciendo la

segunda generación. Después de años de desarrollo, aparece una nueva tecnología que aporta una verdadera experiencia de comunicación en tiempo real a la web: Web Real-Time Communication (WebRTC). La tecnología WebRTC integra de manera nativa la comunicación en tiempo real dentro de los navegadores web y los habilita para comunicarse usando audio, vídeo y datos sin la necesidad de plug-ings o extensiones (Zeidan, Lehmann, & Trick, 2014).

Sin embargo los trabajos con WebRTC se centran en la comunicación de los navegadores web (peer-to-peer) para poder proveer una interfaz de comunicación universal (Vogt, Werner, & Schmidt, 2013).Dado que la tecnología WebRTC es relativamente nueva existen pocos casos de estudio que integran dicha tecnología con clientes SIP, y pocos de ellos son casos exitosos. Por lo tanto es necesario estudiar la interoperabilidad entre las comunicaciones SIP y WebRTC para buscar soluciones a los problemas que se pueden presentar y dejar un punto base para proyectos futuros.

1.2 Justificación e Importancia

La VoIP es una tecnología que permite realizar llamadas de voz utilizando una conexión a Internet, y que ha revolucionado el mundo de la telefonía, ofreciendo nuevos servicios y progresivamente mejores precios al cliente. Disponer de un servicio de VoIP en la Nube permitiría a una empresa el poder comunicarse, sin necesidad de comprar, instalar, configurar, gestionar, y mantener los equipos que realizan el procesamiento y conmutación de las llamadas telefónicas, únicamente tendría que marcar un número destino desde cualquier punto con acceso a Internet. La VoIP en la Nube es una solución que puede ayudar rápidamente a las organizaciones a optimizar recursos. Está demostrado que la mayoría de las empresas consideran que las comunicaciones de bajo costo económico son un elemento esencial de su modelo de negocio (Verkasalo, 2006).

Las empresas enfrentan dificultades en relación a la complejidad de satisfacer las necesidades tecnológicas de manera rápida, con el mínimo costo y con el menor riesgo para sus inversiones. En este sentido, la VoIP en la Nube ofrece una amplia gama de servicios a una variedad de clientes, desde hogares, Pequeñas y Medianas

Empresas (PYMES) o clientes empresariales a precios muy económicos (Sacker, Santaiti, & Spence, 2006).

Se tiene previsto que las empresas optarán por HTML5 para aplicaciones móviles en el mediano y largo plazo (Akhawe, 2014). Es por ello que se plantea la implementación de VoIP en la Nube en el que el usuario no instale ninguna aplicación en su dispositivo y sólo utilice un navegador Web compatible con la tecnología HTML5 para comunicarse. De esta manera, las comunicaciones se llevan a cabo directamente desde un navegador sin necesidad de instalar una aplicación (generalmente de escritorio) en el terminal del usuario final.

1.3 Alcance

Este proyecto plantea la implementación de un sistema VoIP en la nube con integración para WebRTC. El sistema permitirá la comunicación tradicional entre usuarios SIP o entre un usuario SIP y otro con la nueva tecnología WebRTC. Se evaluará el rendimiento del sistema mediante un load test para determinar su capacidad y sus límites y así plantear mejoras futuras.

El programa a utilizar es SIPp (SIPp, 2016) una herramienta de software libre que permite emular miles de usuarios SIP para realizar el load Test a la centralita implementada. Mediante esta herramienta se pretende analizar el comportamiento del sistema bajo la siguiente metodología:

- Implementación de diferentes escenarios.
- Implementación de diferentes protocolos (usuarios SIP, usuarios WebRTC).
- Análisis de las seguridades del sistema.

SIPp muestra en tiempo real las estadísticas del test que se está realizando, tales como el número de usuarios, el tráfico en la red, Delay, Jitter y paquetes perdidos o retrasmitidos. Todo esto nos sirve para evaluar la calidad de las llamadas.

El aporte a la investigación de este proyecto es el estudio de los nuevos protocolos de comunicación, su utilizo en el campo de las Telecomunicaciones y evaluar la compatibilidad con las tecnologías ya existentes.

1.4 Objetivos

1.4.1 General

- Implementar un sistema VoIP en la nube con integración para WebRTC.

1.4.2 Específicos

- Definir los recursos a utilizar.
- Diseñar el sistema VoIP.
- Estudio de escalamiento.
- Implementar el diseño en la nube.
- Evaluar el rendimiento del sistema mediante un Load Test.
- Analizar la seguridad del sistema.

CAPÍTULO II

ESTADO DEL ARTE

2.1 VoIP

VoIP (Voz sobre el Protocolo de Internet) es una tecnología que permite utilizar redes IP para aplicaciones de voz, como telefonía, tele-conferencias y mensajes de voz (Gómez & Gil, 2009). Esta tecnología define cómo transportar llamadas sobre una red IP, incluyendo la conversión A/D y la paquetización del flujo de voz. Los entandares de VoIP crean un sistema de telefonía donde es posible implementar características avanzadas como desvió de llamadas, buzón de voz, llamada en espera, música de espera y agenda de contactos.

Los servicios de VoIP convierten la voz a una señal digital que es transportada sobre una red IP. Si se llama a un teléfono tradicional, la señal se convierte nuevamente a su estado analógico antes que llegue a su destinación. VoIP permite realizar llamadas directamente desde la computadora mediante un softphone, un teléfono VoIP, un teléfono analógico conectado a un adaptador y últimamente también desde un Smartphone. En resume, cualquier dispositivo conectado a Internet está habilitado para utilizar los servicios de VoIP (Verkasalo, 2006) en cualquier momento.

2.1.1 Componentes de un sistema VoIP

Existen tres componentes fundamentales que, complementados con otros para ofrecer servicios adicionales, forman la estructura de un sistema VoIP.

- EndPoints: Son terminales que permite realizar y recibir llamadas hechas por el usuario. Es el encargado de codificar y empaquetar la voz en el usuario de origen y viceversa cuando llega al usuario de destino. Pueden ser estos teléfonos analógicos conectados a un adaptador (ATA), un teléfono VoIP o un softphone instalado en una computadora o un smartphone.
- Gatekeeper: Controla el procesamiento de la llamada y gestiona toda la comunicación VoIP. Además provee el manejo y control sobre los

recursos de la red. Usualmente es un servidor en el cual está instalado un software IP-PBX, como Asterisk.

- Gateway: Es el encargado de conectar una red VoIP con cualquier otro tipo de red, como la PSTN.

2.1.2 Protocolos

El objetivo principal para cualquier protocolo VoIP es el mismo: crear un flujo bidireccional en tiempo real (RTP) entre dos terminales VoIP envueltos en una conversación. Sin embargo, VoIP utiliza varios protocolos de control y señalización de llamadas que se diferencian por su arquitectura y procedimientos, entre los cuales están:

H. 323: Es un entandar que especifica los componentes, el protocolo y los procedimientos para proveer servicios de audio y video en tiempo real, y transferencia de datos sobre una red de paquetes (Joskowicz, 2015). H.323 es parte de una familia de recomendaciones de la ITU-T (Unión Internacional de Telecomunicación) llamada H.32x que define todo los aspectos para la transmisión de voz, vídeo y datos.

SIP: Session Initiation Protocol, por sus siglas en inglés, está definido en el documento RFC 3261 de la IETF (Rosenberg, y otros, 2010). Es un protocolo que define los comandos y respuestas necesarios para configurar y establecer una llamada. Además especifica características como seguridad y servicios UDP o TCP. Este protocolo define la señalización end-to-end necesaria entre dos usuarios para poder realizar una llamada. SIP es un protocolo que incorpora muchos elementos de HTTP, usando el mismo modelo de peticiones, encabezados similares y códigos de respuesta.

MGCP: Media Gateway Control Protocol, por sus siglas en inglés, es un protocolo detallado en el estándar RFC 3435. En este protocolo la señalización de una llamada está separada del tráfico de voz. Se define una nueva entidad llamada Call Agen (CA) para controlar el Gateway de voz y el procesamiento de la llamada. Esta arquitectura simplifica la migración de la red PSTN a VoIP.

SCCP: Es un protocolo de propiedad de CISCO utilizado entre sus propios dispositivos. Los terminales (teléfonos) que utilizan el protocolo SCCP son denominados clientes Skinny, estos se comunican con Cisco Unified

Communications Manager para establecer una llamada con otro terminal que puede utilizar cualquier otro protocolo, como H.323.

2.1.3 Códecs

El ancho de banda siempre ha sido un recurso limitado. Es por ello que se han hecho muchos esfuerzos para minimizar el ancho de banda requerido por un determinado servicio para lograr que la red soporte el mayor número de usuarios. En este contexto, comprimir las señales de voz y al mismo tiempo mantener la calidad, percibida por el usuario, a niveles aceptables representa un reto hasta el día de hoy.

Los códecs de voz son algoritmos que permiten al sistema transportar señales analógicas en una red digital. Existen varios códecs que se diferencian en complejidad, ancho de banda requerido y calidad de voz. Entre más ancho de banda, más calidad de voz. Pero esto nos lleva a una relación inversa entre la calidad de la señal y la eficiencia de la red. La red telefónica publica sigue muestreando la voz a 8 kHz y la cuantifica con 8 bit de acuerdo a ITU-T G.711 (ITU, T, 1988). Sin embargo se requiere una tasa de codificación inferior a los 64 kb/s si se quiere maximizar la cantidad de tráfico transportado en una red con recursos limitados. Por tal razón existen códecs que comprimen la voz con tasas de hasta 2.15 kb/s a dispensa de una cualidad muy por debajo del estándar G.711.

En la tabla 1 (Gómez & Gil, 2009), se muestran los códecs más utilizados con sus respectivas características.

Tabla 1
Comparativa de los códecs más utilizados

Nombre	Bitrate (kbps)	Audio útil (Bytes)	Ancho estimado (kbps)	Latencia (ms)	Observaciones	MOS
G.711	64	240	74.6	30	PCM. Existen dos versiones "u-law" (US, Japon) y "a-law" (Europa)	4.1
G.723.1	6.4	24	17	30	Utiliza Multipulse Maximun Likelihood Quantization (MP-MLQ) o Algebraic-Code-.Excited-Linear-Prediction (ACELP). Alta compresión manteniendo una Buena calidad de sonido	3.8 3.9
G.726	32	80	48	20	ADPCM. Sustituye a los codecs G.721 y G.723	3.85
G.728	16	60	26.6	30	Utiliza Code-Excited-Linear-Prediction (CELP) para codificar	3.61
G.729	8	20	24	20	G.729: Códec original. G.729A menos complejo que G.729 pero menor calidad (compatible con G.729). G.729B es como G.729 pero con supresión de silencios (no es compatible con las anteriores).	3.92
GSM 06.10	13.2	33	29.2	20	Utiliza Regular-Pulse-Excitation-Long-Term-Predictor (RPE-LTP). Usado por la tecnología celular GSM. Es soportado por gran cantidad de plataformas hardware y software.	3.8
LPC10	2.4	7	16.7	22.5	Linear-Predictive-Codec (LPC). La voz suena un poco "robótica".	
Speex (NB)	11.2	28	27.2	20	El bitrate es variable. Además detecta el silencio.	
iLBC	15.2	57	25.8	30	Reciente, por lo que su soporte en dispositivos comerciales es muy reducido. Requiere un importante procesamiento de sonido.	4.14

2.1.4 Calidad de Servicios

Debido a los avances en la tecnología a lo largo de los años, la red pública de telefonía se caracteriza por una alta calidad en las llamas. Pero cuando se trata de VoIP, es necesario garantizar una buena calidad de servicios para ser capaces de alcanzar los mismos niveles (Christiansen, Giotis, & Mathur, 2004). Los indicadores principales que caracterizan la calidad de voz en la Internet son tres: Delay, Jitter y paquetes perdidos.

Delay: Es el tiempo requerido a la voz para llegar desde la boca de quien habla hasta el oído de quien escucha. Round Trip Delay es la suma del retraso que ocurre en la llamada sea de ida que de regreso. En sistemas VoIP, la propagación del Delay se ve afectada también por el retardo en la paquetización y por el tiempo requerido para transmitir el paquete en la red. Todo esto afecta en el retardo al momento de realizar una llamada. Según la recomendación de la ITU-T G.114 (ITU-T, 2003), el Delay aceptable para una llamada tiene que estar entre los 150 y 400 ms.

Jitter: Es la variación en el tiempo de llegada de los paquetes en lado del receptor. Esto afecta de madera considerable la calidad de la llamada, incluso más que el Delay. Se espera que el transmisor envié cada paquete de voz a intervalos regulares y de esta forma en el lado del receptor se reconstruye la voz de manera continua. Pero debido al Jitter los paquetes de voz no llegan al mismo tiempo y el decodificador reconstruye la voz de manera incorrecta. La ruta de transmisión de un paquete en una red IP no siempre es la misma, paquetes pertenecientes al mismo flujo de voz puede llegar a destino por diferentes caminos y esto genera el problema de Jitter.

Paquetes Perdidos: Algunas veces, debido al Delay o al Jitter presentes en la red, los paquetes transmitidos no llegan a destinación y son descartados. Debido a la perdida de paquetes la llamada se vuelve discontinua e incomprensible. En aplicaciones que no son en tiempo real está perdida de paquetes es transparente para el usuario y se soluciona con mecanismos de retransmisión. Pero en VoIP no se pueden aplicar estos métodos debido a que los paquetes retransmitidos llegarían demasiado tarde y esto interrumpiera el flujo de la conversación.

2.2 Cloud Computing

Con el rápido desarrollo de las tecnologías de procesamiento y almacenamiento de información y junto a la Internet, los recursos de computación se han vuelto cada vez más económicos, más potentes y presentes en cualquier parte del mundo. Esta tendencia ha propiciado un nuevo modelo de computación llamado Cloud Computing, donde estos recursos, como CPU y almacenamiento, son entregados como utilidades que pueden ser utilizados por el usuario bajo demanda a través del Internet. En Cloud Computing, se tiene dos tipos de proveedores: el de infraestructura, quien es el encargado de manejar la plataforma en la nube y quien entrega los recursos en donde el usuario paga por lo que usa; y el proveedor de servicios, quien renta sus servicios a los usuarios finales (Zhang, Cheng, & Boutaba, 2010).

El rápido desarrollo de esta tecnología ha tenido un gran impacto en la industria de la información, captando la atención de grandes compañías como Alphabet, Amazon y Microsoft quien se disputan entre ellos para desarrollar una plataforma en la nube que sea la más potente, confiable y económica, como lo muestra la figura 1 (Leong, 2016).

Figura 1. Cuadrante de los proveedores de Cloud Computing. Fuente: (Leong, 2016)

2.2.1 Arquitectura de Cloud Computing

La arquitectura de Cloud Computing se puede dividir en 4 capas (Zhang, Cheng, & Boutaba, 2010):

La capa de hardware: Esta capa es la responsable de manejar los recursos físicos de la nube, que incluyen los servidores físicos, routers, switches, y sistemas de energía e enfriamiento. En la práctica, la capa de hardware se implemente directamente en el data center. Un data center está constituido por miles de servidores que están organizados en racks y están interconectados mediante routers y switches.

La capa de infraestructura: También conocida como la capa de virtualización, crea los recursos de almacenamiento y computacionales que serán ofrecidos al usuario mediante tecnologías de virtualización. La capa de infraestructura es un componente esencial de Cloud Computing, debido a que características, como la asignación dinámica de los recursos, solo son posibles gracias a la virtualización.

La capa de plataforma: Construida sobre la capa de infraestructura, es el sistema que nos permite crear y administrar los recursos entregados. En esta capa se puede construir, probar e implementar la aplicación deseada.

La capa de aplicación: Es la última capa en la estructura, aquí es donde se encuentra la aplicación en la nube que se pretende entregar al usuario. En este caso de estudio seria el servidor Asterisk como central telefónica en la nube.

A diferencia de una granja tradicional de servidores, la arquitectura de Cloud Computing es modular. Una arquitectura modular permite implementar un amplio rango de aplicaciones y reduce los problemas de mantenimiento y manejo.

2.2.2 Modelos de Cloud Computing

Existen muchos factores a considerar cuando se decide implementar una aplicación en la nube. Por ejemplo, algunas aplicaciones se enfocan en reducir costos mientras otras prefieren una alta seguridad y estabilidad. Para cada uno de estos requerimientos existen diferentes modelos de nubes.

Cloud Publica: La nube pública ofrece sus recursos como servicio al público en general. Esto representa un beneficio para el usuario dado que no tendrá que invertir ningún capital inicial en infraestructura. Sin embargo el usuario no tendrá el control completo sobre su información, red o configuración de seguridad.

Cloud Privada: También conocida como nube interna, este tipo de modelo está diseñado para uso exclusivo de una sola empresa quien será la encargada de construirla y configurarla según sus requerimientos. Una nube pública ofrece el control total de sus recursos y una buena seguridad. Sin embargo esto no sería muy diferente a tener una granja de servidores tradicional y eliminaría uno de los más importantes beneficios de Cloud Computing que es la inversión de capital inicial.

Cloud Híbrida: Una nube híbrida es la combinación de una nube pública con una privada. Parte de los servicios de infraestructura está en la nube privada y la otra parte está en la nube pública. Este modelo junta las mejores características de cada uno, ofreciendo flexibilidad, control y seguridad. Pero esto requiere determinar cuidadosamente cuáles recursos serán asignados a la nube pública y cuáles a la nube privada.

Cloud Privada Virtual: Como alternativa a los modelos precedentes, una nube privada virtual se basa en una conexión VPN que permite configurar su propia

topología de red y reglas de seguridad. Esto significa que no solo se está virtualizando los servidores sino también la capa de red. Este modelo es un puente entre la infraestructura de red de una empresa y los servicios de una nube.

Para una empresa es fundamental decidir cuál modelo de nube se acopla mejor a su modelo de negocio. Las nubes híbridas dominan el sector pero cada vez se está prestando más atención a la nube privada virtual.

2.2.3 Proveedores Comerciales

Existen ya varios proveedores de infraestructura en la nube y cada vez se crean más debido a que es un modelo de negocio que genera mucha rentabilidad. En la tabla 2 (Zhang, 2010) se resume los tres proveedores más populares actualmente y se clasifican en base a sus modelos de computación, almacenamiento y autoescalamiento.

Tabla 2
Comparación de los proveedores de Cloud Computing

Proveedor	**Amazon**	**Windows Azure**	**Google App Engine**
Arquitectura	Infraestructura como servicio	Infraestructura como servicio	Plataforma como servicio
Aplicaciones	Aplicaciones de tipo general	Aplicaciones de Windows de tipo general	Aplicaciones Web con soporte para diferentes interfaces
Computación	Xen Virtual Machine	Microsoft Common Language Runtime (CLR) VM	Interfaces predefinidas para aplicaciones Web
Almacenamiento	Amazon Simple Storage Service(S3) Amazon Simple DB	Azure Storage Service SQL Data Services	BigTable y Mega Store
Escalamiento	Escalamiento automático dependiendo de los parámetros que el usuario configure.	Escalamiento automático dependiendo de los parámetros que el usuario configure.	Escalamiento automático que es transparente para el usuario

2.3 WebRTC

La comunicación en tiempo real, (RTC) por sus siglas en inglés, se caracteriza por transmitir la información sin ningún retraso, usualmente es utilizado en una comunicación peer-to-peer. WebRTC es uno de los estándares desarrollado por el

Consorcio World Wide Web (W3C), que permite a navegadores web, como Chrome, Mozilla u Opera, comunicarse en tiempo real gracias a un simple API integrado directamente en el navegador. Esto elimina el problema de incompatibilidad entre diferentes navegadores, estandarizando la comunicación y habilitando la transmisión de audio o vídeo (Vogt, Werner, & Schmidt, 2013).

Hasta hace poco, los navegadores web eran utilizados para navegar en la web, email y otros pocos servicios. Sin embargo el mayor problema que presentaban los navegadores era su bajo rendimiento en una comunicación de audio o vídeo. Los algoritmos de compresión y descompresión de audio y vídeo eran privativos y costosos. Después se desarrollaron los Plugins en los años 90, que permitían reproducir vídeos utilizando la tecnología flash, estos abrieron un nuevo camino para realizar vídeo llamadas. Pero no es hasta la llegada de WebRTC que los navegadores web son capaces de enviar y recibir información en tiempo real.

WebRTC es una solución ideal para el servicio al cliente, dado que permite un acceso directo al usuario con un agente de servicio mediante cualquier navegador web. Por ejemplo el usuario puede comunicarse desde la misma página web de la empresa sea desde una computadora o su Smartphone con acceso a Internet.

2.3.1 Arquitectura de WebRTC

La arquitectura de WebRTC se basa en un modelo cliente-servidor y el flujo de datos se realiza directamente entre iguales (peer-to-peer), como lo muestra la figura 2 (Deniz, 2016).

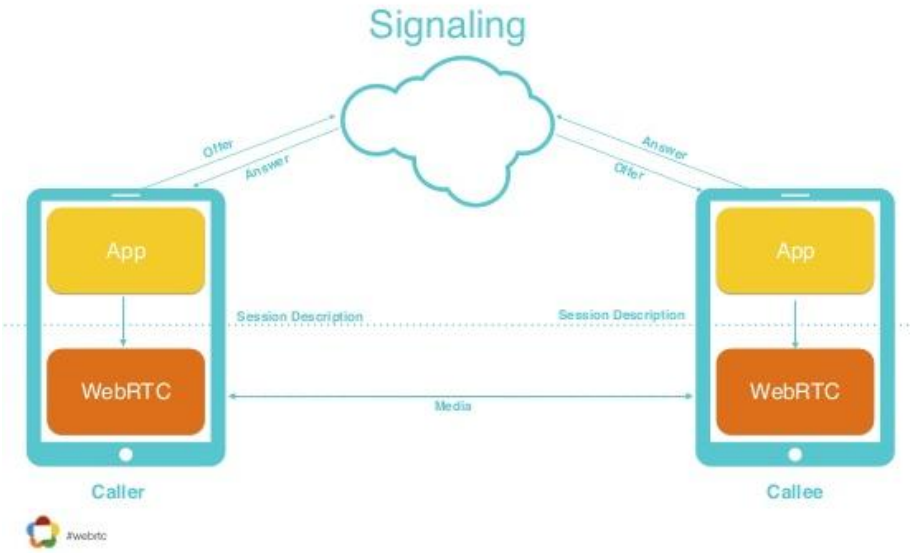


Figura 2. Señalización WebRTC entre dos dispositivos móviles
Fuente: (Deniz, 2016)

La señalización necesaria para establecer la comunicación se envía mediante HTTP o WebSockets al servidor, quien es el encargado de traducir y manejar las señales. En este caso de estudio se pretende que el servidor sea Asterisk y el navegador web pueda realizar llamadas como si fuera una extensión SIP.

2.3.2 WebRTC no es un Webphone

Implementar un Webphone con las funciones de voz, video, y chat en general requiere de un gran esfuerzo debido a la complejidad y a la compatibilidad de plataformas, más aun en el caso de una aplicación para celulares (Android, iOS). Esto conlleva un alto costo económico y largos tiempos de desarrollo. Como alternativa se presenta WebRTC que es compatible con cualquier plataforma (navegadores de escritorio o móviles) y permite crear aplicaciones con toda la flexibilidad y funcionalidades que están permitidas en Internet.

WebRTC puede desarrollar un Webphone pero no solo eso, también cualquier tipo de aplicaciones java en tiempo real para transmitir datos, mensajes, audio, video etc. Esto es posible gracias a un interfaz de programación de aplicaciones (API) llena de funcionalidades como por ejemplo: conexión peer-to-peer (RTCPeerConnection), estadísticas de la conexión (RTCStats) y obviamente flujo de video y audio (getUserMedia).

Como se muestra en la figura 3 (Sipwise , 2016), con WebRTC se puede desarrollar un Webphone que interactúe con la central telefónica en la nube de la misma forma que lo hace un cliente SIP normal. Con las mismas funcionalidades de VoIP pero con una interfaz que hace más simple la conexión con otros clientes desde un navegador web.

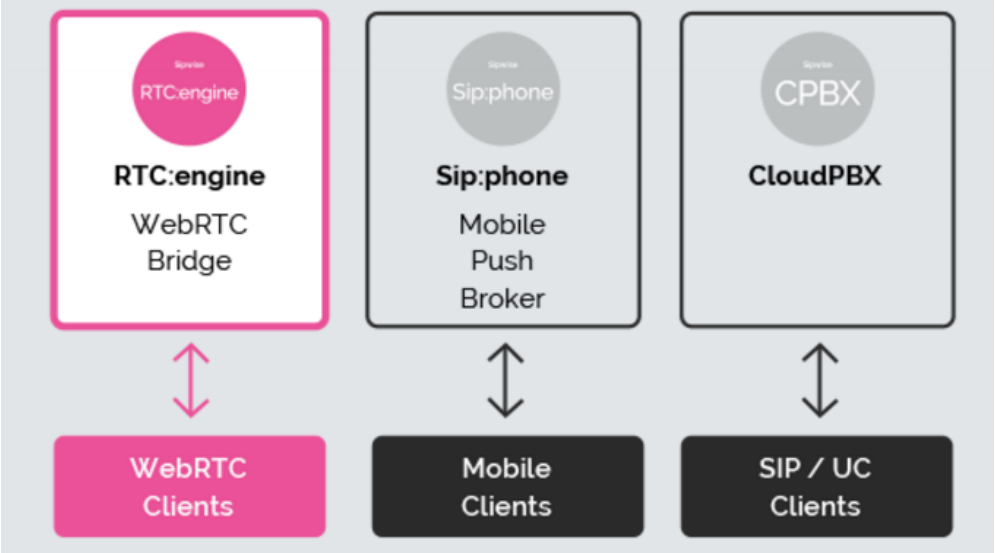


Figura 3. Clientes SIP y WebRTC para un servidor en la nube
Fuente: (Sipwise , 2016)

2.4 SIPp

SIPp (SIPp, 2016) es una herramienta que permite realizar pruebas de rendimiento para el protocolo SIP. Fue desarrollado por Richard Gayraud y Olivier Jacques de la compañía Hp, pero ahora se mantiene como un proyecto Open Source. SIPp es ampliamente utilizado por la comunidad científica para realizar pruebas en sistemas VoIP generando trafico SIP y mostrando las estadísticas respectivas. Los datos analizados son el tiempo de respuesta, la duración de la llamada, número de llamadas exitosas y falladas, número de llamadas concurrentes, etc.

SIPp puede generar múltiples llamadas hacia un servidor remoto. Además, se puede crear simultáneamente varias instancias de SIPp en la misma computadora, siempre y cuando se asigne diferentes sockets a cada una.

2.4.1 Parámetros de configuración

SIPp (SIPp, 2016) se puede ejecutar desde la línea de comandos de cualquier sistema GNU/Linux y necesita de varios parámetros de configuración. El formato del comando principal es:

```
sipp  remote_host  [options]
```

Donde remote_host es el servidor destino hacia donde se generan las llamadas, comúnmente es la dirección IP publica del servidor VoIP. Después de la ejecución del comando, SIPp comienza a generar y a enviar mensajes SIP hacia el servidor. La secuencia de los mensajes está definida dentro de los escenarios por default o por un escenario externo en formato XML defino por el usuario. El parámetro [options] permite configurar diferentes atributos del escenario como por ejemplo el puerto de salida, la tasa de llamadas concurrentes, el límite máximo de llamadas, etc.

Después de que se creó una instancia de SIPp, en la pantalla se muestra el diagrama de los mensajes SIP y también las estadísticas en tiempo real, como se muestra en la figura 4.

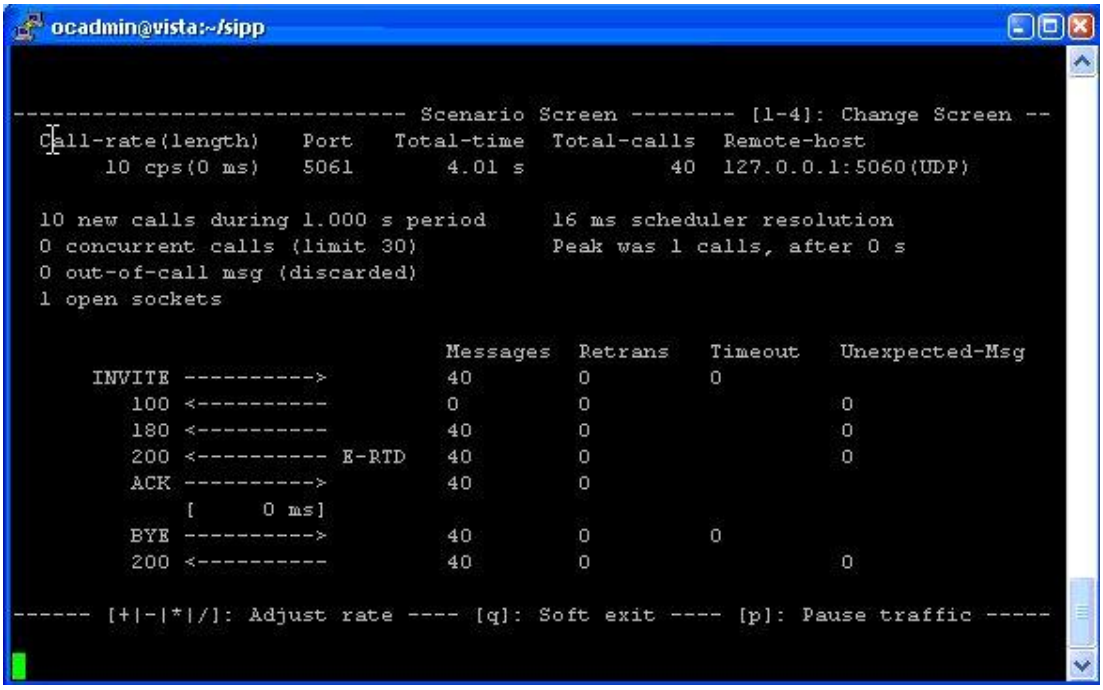


Figura 4. Herramienta SIPp
Fuente: (SIPp, 2016)

CAPÍTULO III

IMPLEMENTACIÓN DEL SISTEMA

3.1 Arquitectura

La arquitectura del sistema está compuesta por el servidor en la nube y los terminales en una red local, como se puede ver en el diagrama de la figura 5. El servidor Asterisk se encuentra dentro de la plataforma de Amazon Web Services. Los usuarios se pueden registrar desde cualquier red local conectándose a la IP pública del servidor.

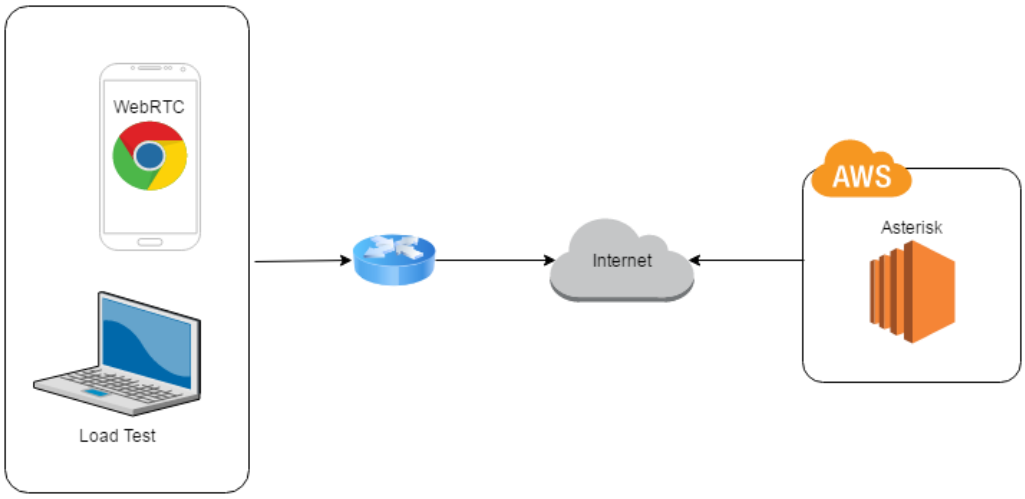


Figura 5. Arquitectura del sistema

3.2 Servidor

El servidor se implementa en una instancia de Amazon Web Services y para hacer esto necesitamos seguir los pasos que se detallan a continuación.

3.2.1 Crear una cuenta de Amazon Web Services.

Para utilizar los servicios de Amazon se necesita crear una cuenta y asignar a esta una tarjeta de crédito para poder pagar los servicios que se utilizan. Para esto Amazon guía al usuario paso a paso comenzando desde la página de registro como se indica en la figura 6.

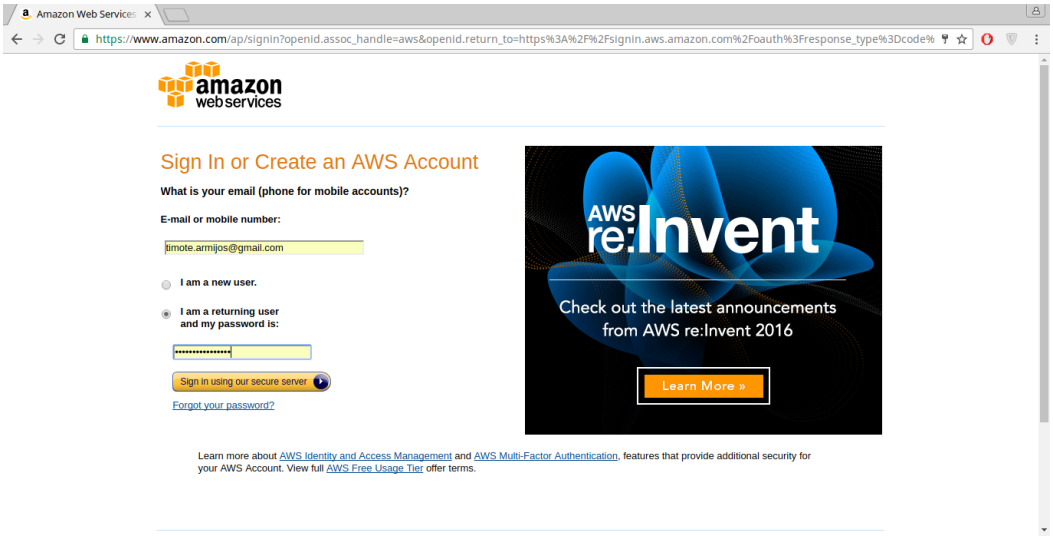


Figura 6. Página de inicio para crea una cuenta en Amazon Web Services
Fuente: (Amazon, 2016)

3.2.2 Configuración de una instancia EC2.

Ahora ya es posible configurar una instancia Linux desde la página de inicio, el procedimiento se explica ampliamente en el anexo 1. En esta sección vamos a resaltar las partes más importantes y que son fundamentales para que la central telefónica Asterisk funcione correctamente.

Primero se necesita analizar las características necesarias para que el servidor tenga un buen rendimiento. En la tabla 3 se presenta los requerimientos mínimos recomendados por (Van Meggelen, Madsen, & Smith, 2007) para un servidor Asterisk.

Tabla 3
Requerimientos mínimos para un sistema Asterisk

Propósito	**Número de Canales**	**Recomendación Mínima**	
Pruebas	No más de 5	400 MHz x86	256 MB RAM
SOHO	5 a 10	1 GHz x86	512 MB RAM
Sistema pequeño	Hasta 15	3 GHz x86	1 GB RAM
Sistema mediano-grande	Más de 15	CPUs duales con arquitectura distribuida	

Amazon EC2 le permite elegir entre dos tipos de instancias: de desempeño fijas, denominadas M3, C3 y R3, y de desempeño con ráfagas, denominadas T2. Estas últimas proporcionan un desempeño base pero tienen la capacidad de elevar su rendimiento cuando sea necesario. Las instancias T2 se utilizan en implementaciones que no usan la CPU por completo de manera frecuente o constante, pero que en ocasiones necesitan alcanzar ráfagas, como sería el caso de una ráfaga de llamadas simultaneas. Muchas aplicaciones, como servidores web y bases de datos, no necesitan de manera constante altos niveles de CPU, pero en ocasiones puntuales necesitan disponer de un núcleo de prestaciones elevadas cuando la aplicación lo necesite realmente. Las instancias T2 están pensadas para estos casos de uso.

Junto a una instancia de Amazon EC2 se necesita del servicio de Amazon Elastic Block Store (EBS) que permite crear volúmenes de almacenamiento. Una vez creado un volumen se puede darle cualquier uso que se daría a un dispositivo de almacenamiento físico. Amazon EBS proporciona dos categorías principales: almacenamiento respaldado por SSD, para mayor desempeño (tipo io1) o de uso general (tipo gp2), y almacenamiento respaldado por HDD para cargas de trabajo a los que se accede con menos frecuencia.

Analizando la tabla 3 y comparando estudios similares, la instancia con la que se va a realizar las pruebas es una T2.micro con las siguientes características:

- Procesador: 1 vCPUs, 2.5 GHz hasta 3.3 GHz, Intel Xeon Family,
- Memoria: 1 GiB
- Disco Duro: Amazon Elastic Block Store (EBS gp2)

Esta instancia presenta un equilibrio entre recursos de informática, memoria y red para aplicaciones interactivas de baja latencia, desarrollo y pruebas.

3.2.3 Instalación Asterisk.

Ahora ya es posible instalar Asterisk en la instancia creada, el procedimiento se explica ampliamente en el anexo 2. En esta sección vamos a resaltar las partes más importantes y que son fundamentales para configurar la central telefónica Asterisk en la nube.

Asterisk trabaja mediante archivos de configuración que se encuentran normalmente en el directorio /etc/Asterisk y los más importantes, para este caso de estudio, son sip.conf, extensions.conf, rtp.conf y http.conf.

3.2.3.1 *Configuración del archivo sip.conf*

Para que los teléfonos puedan comunicarse entre ellos, es necesario configurar un usuario para cada uno y especificar que protocolo será utilizado. Se establece que todos los teléfonos utilizaran el protocolo SIP y para ello el archivo correspondiente es sip.conf. En la figura 7, se muestra el archivo sip.conf y podemos identificar la sección [general] y sucesivamente las secciones que corresponden a los teléfonos: [6001], [6002] y [6003].

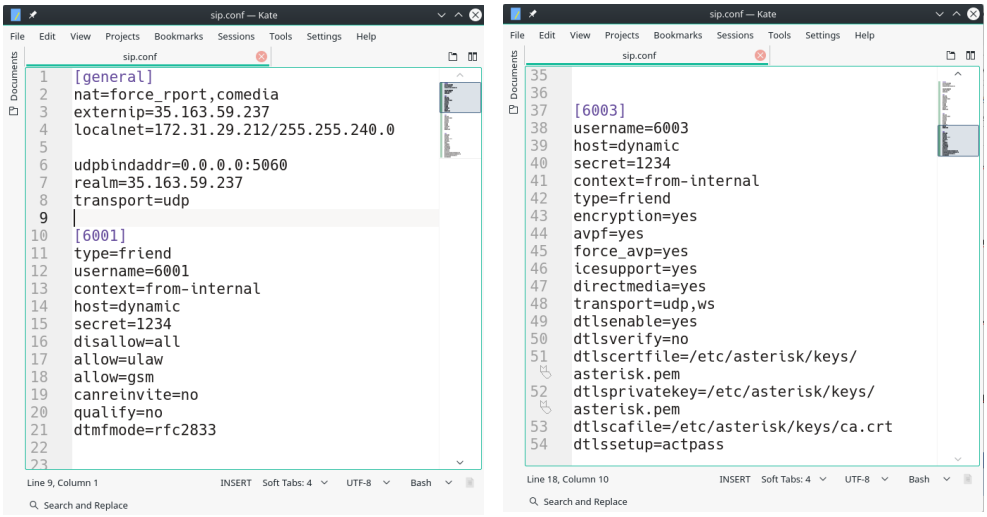


Figura 7. Configuración del archivo sip.conf

Esta configuración sigue los lineamentos especificados en el archivo de ejemplo que proveen los desarrolladores de Asterisk y dentro de este archivo se puede leer que hace cada opción en detalle. Pero debido a que esta configuración es de modelo base se tiene que hacer algunas modificaciones para que pueda funcionar en la nube.

En la sección [general] es necesario especificar que nuestra instancia se encuentra detrás de un NAT y por ende debemos especificar nuestra dirección IP externa y la dirección IP en la red local. Sin estos parámetros los dispositivos se podrán conectar al servidor Asterisk pero tendrán problemas de audio. Estos valores se obtienen desde la página de control de la plataforma de Amazon para la IP pública y con el comando ifconfig para la red interna del servidor, como se ve en la figura 8.

```
nat=force_rport,comedia
externip=35.163.59.237
```

```
localnet=172.31.29.212/255.255.240.0
```

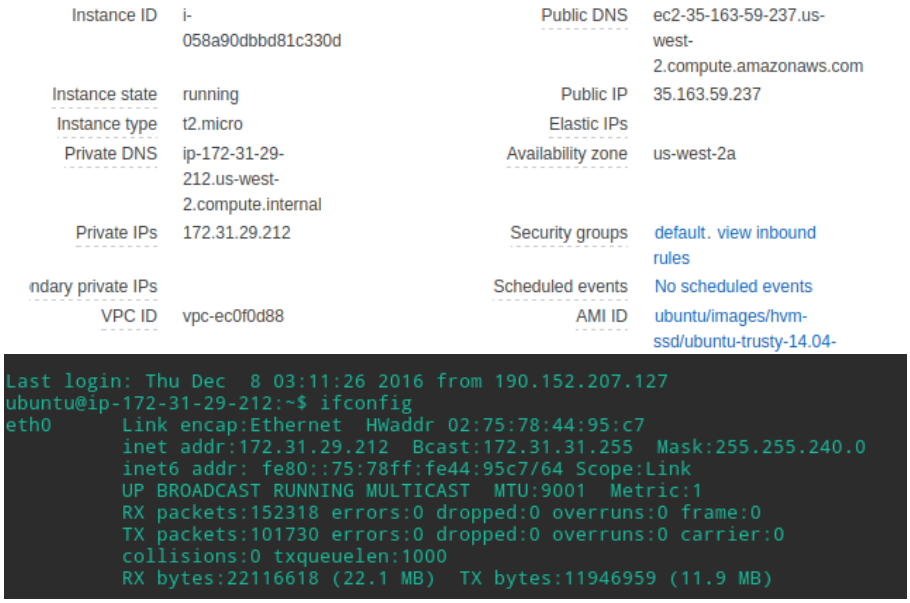


Figura 8. Configuración de red de la instancia en la nube

Sucesivamente se tiene la sección de los teléfonos, para este caso de estudio se declaran tres usuarios: [6001], [6002] y [6003]. Este último utiliza la tecnología WebRTC por ende necesita opciones adicionales para su correcta configuración.

Cuando se tiene el servidor y los teléfonos en la misma red local la configuración es muy simple y directa. Pero cuando sea el teléfono A que el teléfono B están detrás de un NAT y necesitan conectarse y transmitir información, las cosas se vuelven más complicadas. Este problema es muy común en un sistema VoIP implementado en la nube por tal razón ha surgido el protocolo Interactive Connectivity Establishment (ICE), una nueva solución para resolver este tipo de problemas. ICE es un mecanismo estandarizado para establecer una comunicación en tiempo real entre dos agentes que se encuentran detrás de un cortafuego NAT. Junto con un servidor STUN, encargado de detectar la dirección pública de los agentes y el puerto asociado a estos por el NAT, ICE encontrará una conexión para conectar los dos agentes, como se muestra en la figura 9.

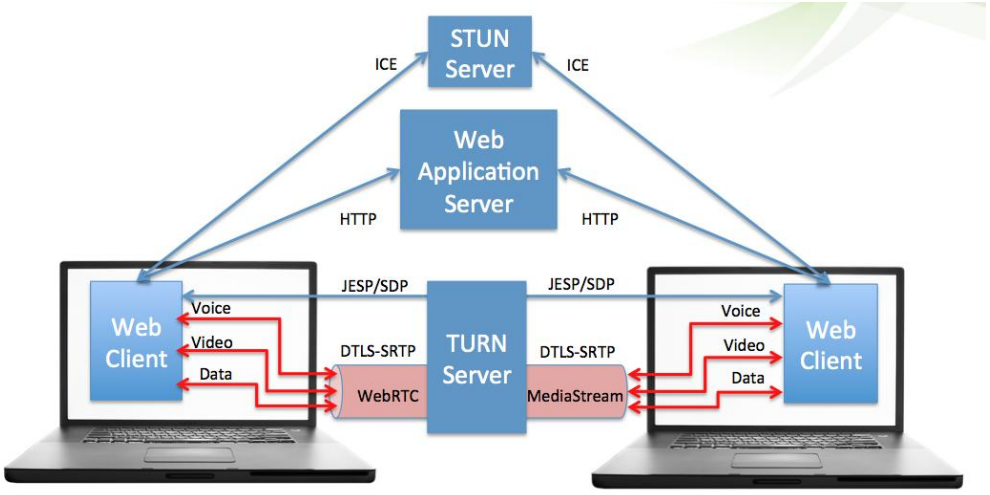


Figura 9. Comunicación mediante el protocolo ICE

Para activar esta solución en Asterisk, con las dos opciones icesupport y transport, se especifica que este teléfono tiene soporte para ICE y que se conectara mediante un websocket.

```
icesupport=yes
transport=udp,ws
```

Una vez encontrado el camino para una conexión en tiempo real debemos especificar con qué estándar se realizará la llamada. El estándar por defecto en la tecnología WebRTC es AVPF que se utiliza sea para audio que para video. Como este estándar no es el utilizado por defecto en Asterisk se necesita agregar las siguientes opciones a la configuración del teléfono.

```
avpf=yes
force_avp=yes
```

Por cuestiones de seguridad toda comunicación que se realiza con la tecnología WebRTC es encriptada. Para esto se utiliza el protocolo criptográfico TLS (Transport Layer Security) que asegura que ningún agente externo puede saber quién es el destinatario de la llama y tampoco pueda escuchar la conversación. Se necesita crear una certificado autofirmado para Asterisk utilizando el script:

```
./ast_tls_cert -C 35.163.59.237 -O "Asterisk" -d /etc/asterisk/keys
```

Donde se especifica la dirección IP del servidor, su nombre y donde se desea guardar los archivos resultantes. Estos serán utilizados por el teléfono [6003] en donde se declara el certificado y la clave, como se muestra con los comandos a continuación.

```
encryption=yes
dtlsenable=yes
dtlsverify=no
dtlssetup=actpass
dtlscertfile=/etc/asterisk/keys/asterisk.pem
dtlsprivatekey=/etc/asterisk/keys/asterisk.pem
dtlscafile=/etc/asterisk/keys/ca.crt
```

Con esta configuración el teléfono puede comunicarse de manera segura y sin ningún problema sea con otro usuario con tecnología WebRTC o con un usuario SIP.

3.2.3.2 *Configuración del archivo extensions.conf*

Ahora que ya se configuraron los teléfonos es necesario definir el comportamiento de Asterisk cuando un teléfono marque una determinada extensión. En la figura 10, se muestra las extensiones configuradas en el servidor.

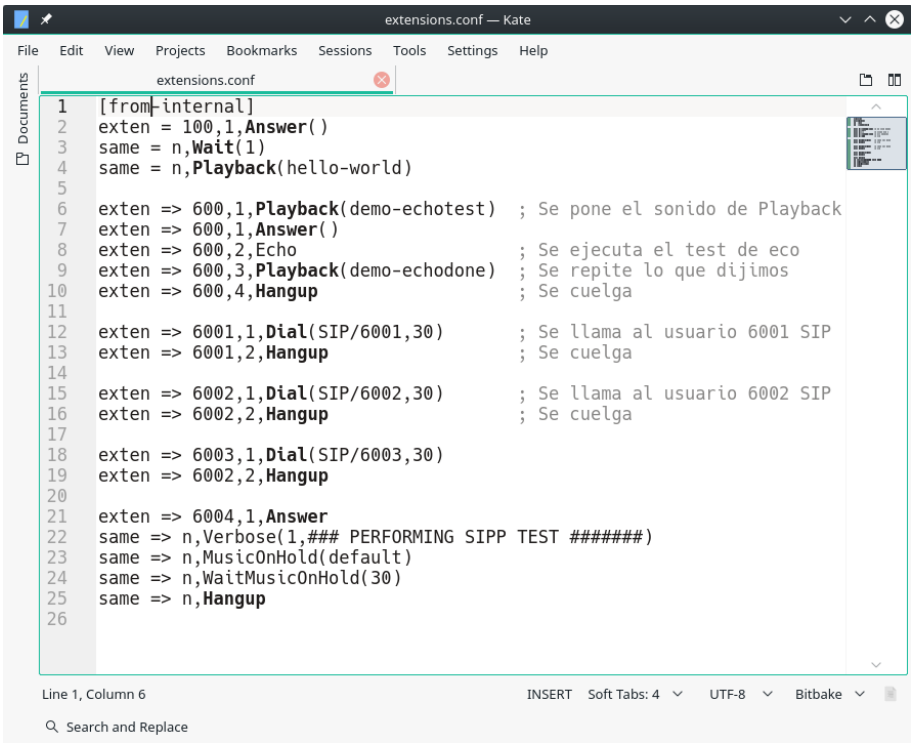


Figura 10. Configuración del archivo extensión.conf

La extensión 100 es para pruebas de configuración, cuando se marca Asterisk contesta la llamada espera un segundo y reproduce un mensaje de bienvenida.

Las extensiones 6001, 6002, 6003 llaman respectivamente a los teléfonos correspondientes.

La extensión 600 se creó para evaluar el servidor cuando está bajo estrés, debido a que su función es la de repetir todo lo que recibe hacia el usuario. Con esto se puede identificar la calidad de la llamada y bajo qué condiciones la llamada se degrada hasta que una conversación sea imposible.

La extensión 6004 se utiliza cuando se realiza el test con la herramienta SIPp, Asterisk contestará la llamada y reproducirá una música de espera por 30 segundos.

3.2.3.3 *Configuración del archivo rtp.conf*

Para transmitir la voz entre dos teléfonos se utiliza el protocolo UDP mediante un puerto específico que se negocia cuando se establece la llamada. El la figura 11 se muestra el archivo rtp.conf donde se establece el rango para estos puertos que inicia en 10000 y termina en 20000.

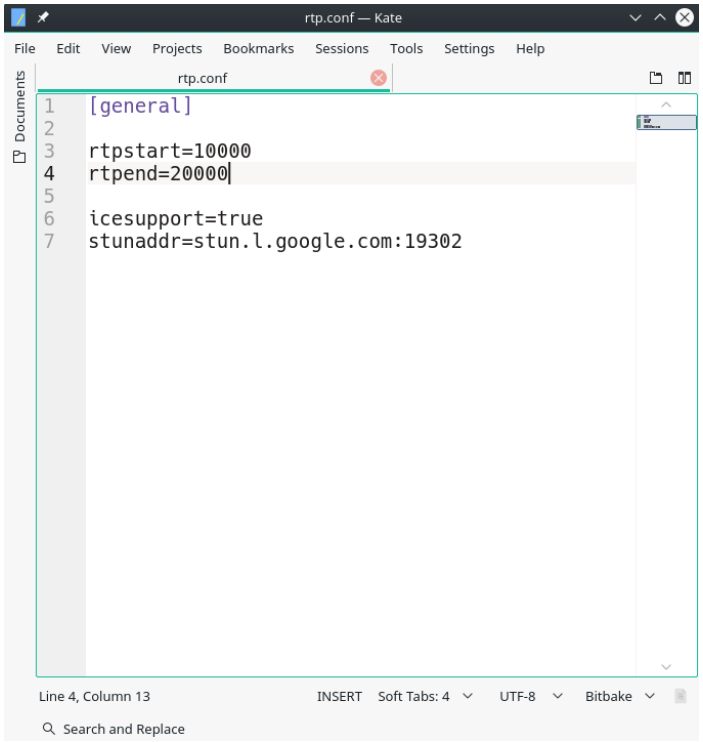


Figura 11. Configuración del archivo rtp.conf

Se necesita abrir los mismos puertos en la configuración del firewall de la instancia donde está el servidor Asterisk. En Amazon se puede abrir un puerto específico o un rango en los dos protocolos posibles TCP o UDP, como se muestra en la figura 12.

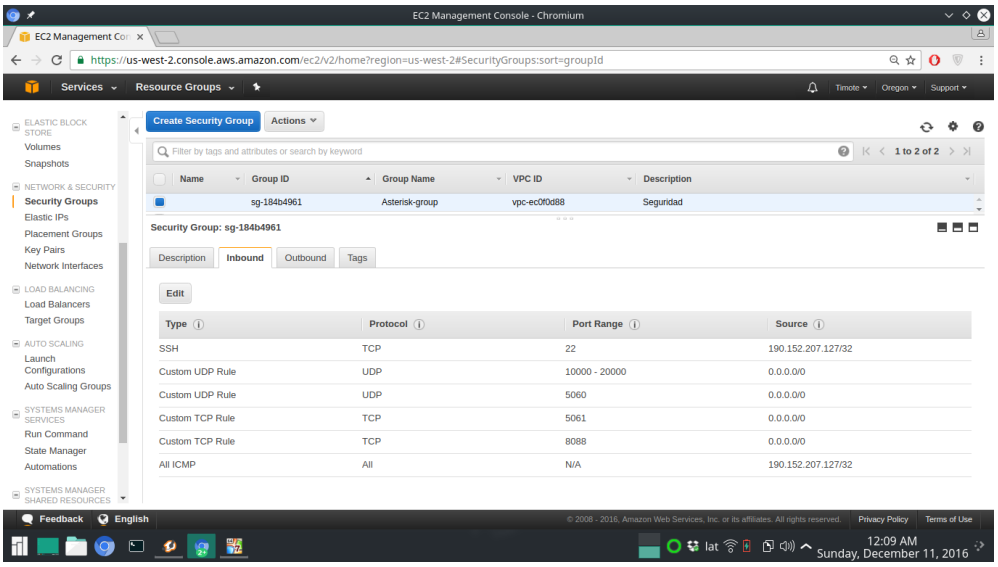


Figura 12. Configuración de seguridad en Amazon Web Services

El puerto TCP 22 es necesario para poderse conectar al servidor mediante el protocolo SSH. Amazon permite especificar desde que dirección IP es posible acceder por este puerto, las opciones son dos desde cualquiera o una especificada por el usuario. Como se puede ver en la imagen se recomienda limitar el acceso a una sola IP para aumentar la seguridad del servidor.

El rango de puertos UDP 10000-20000 son los mismos que se configuraron en el archivo rtp.conf.

El puerto UDP 5060 y el TCP 5061 es el puerto por donde se escucha las peticiones de llamadas desde los teléfonos.

Con la tecnología WebRTC se utilizan websockets y estos trabajan con el puerto 8088 configurado en el archivo http.conf.

Por último se activa el protocolo ICMP para realizar pruebas de conectividad con el comando ping.

3.2.3.4 *Configuración del archivo http.conf*

Se utiliza el servidor HTTP integrado en Asterisk para recibir los pedidos mediante websocket. Para esto se activa el servidor y se indica el puerto asociado al mismo dentro del archivo de configuración http.conf, como se muestra en la figura 13.

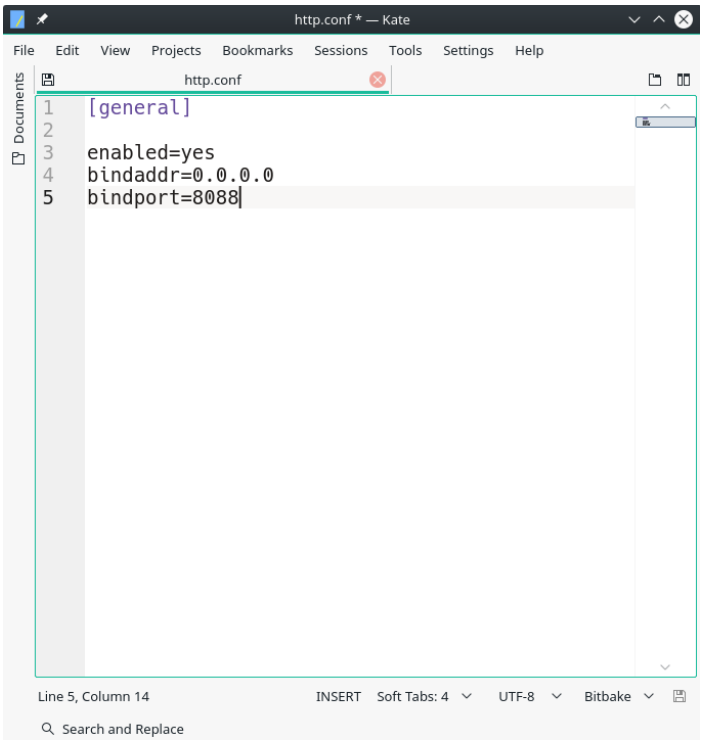


Figura 13. Configuración del archivo http.conf

3.3 Terminales

Para conectarse al servidor se necesita de un softphone, en este caso se utiliza Zoiper que es fácil de instalar y utilizar. Además su soporte multiplataforma (Android, GNU/Linux y Windows) lo hace compatible con todo tipo de usuario independientemente de la plataforma que utilice.

Una vez instalado se presenta la pantalla de inicio, como se muestra en la figura 14. Sucesivamente es necesario configurar el softphone para que se conecte con el servidor en la nube y para ello se tiene que entrar en la pestaña configuración.

Figura 14. Softphone Zoiper

En la configuración se tiene que ingresar los mismo valores creados en el archivo sip.conf en donde se declaran los teléfonos. En este caso se crea el usuario 6001 que se encuentra en la dirección IP 35.163.59.237. Si la información es correcta se informa que el usuario está registrado con un visto verde como se ve en la figura 15.

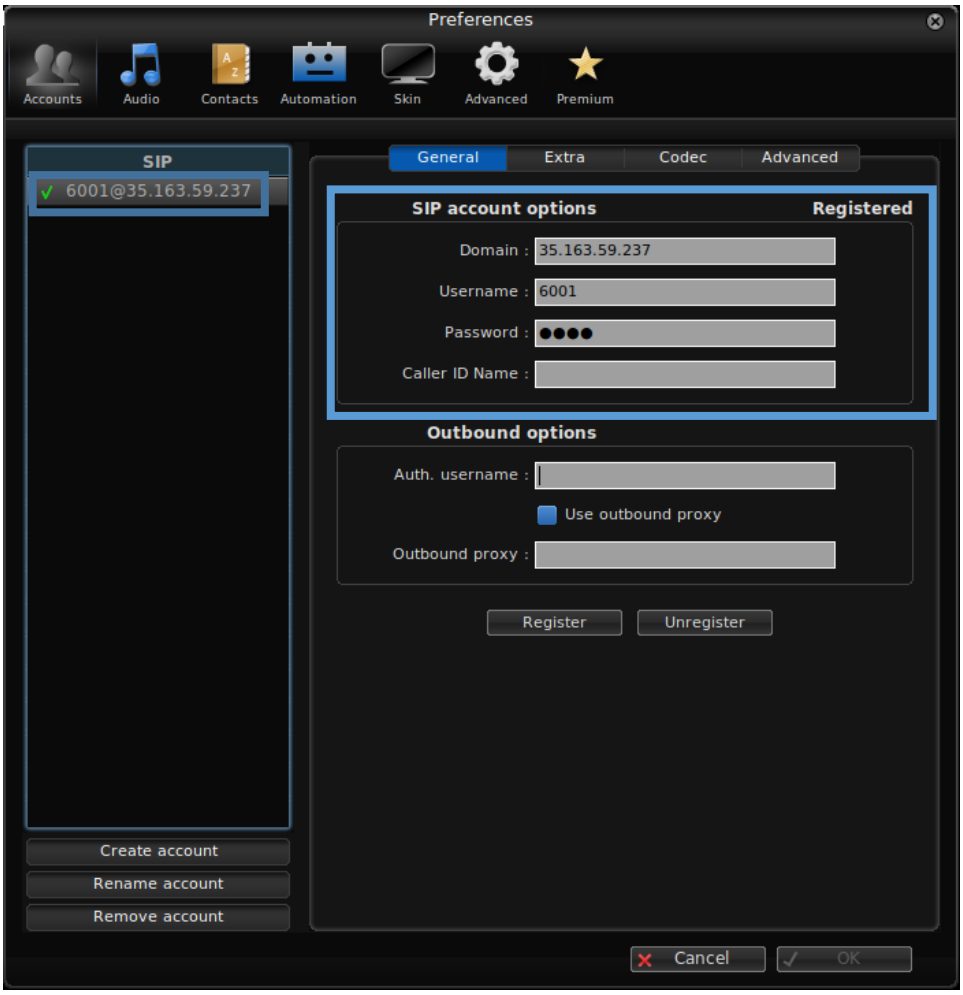


Figura 15. Configuración del usuario 6001 en Zoiper

Sucesivamente se utiliza un cliente remoto WebRTC, existen muchos en la red, en este caso se utiliza la versión demo de SIPML5. Para ello en un navegador se abre la dirección https://www.doubango.org/sipml5/call.htm?svn=250#, como aparece en la figura 16.

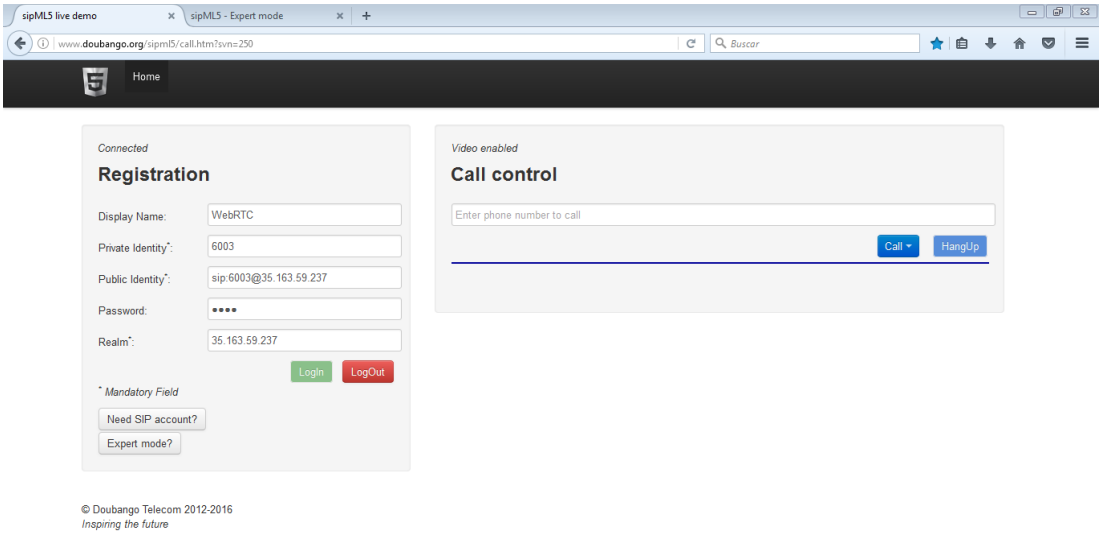


Figura 16. Configuración del cliente remoto WebRTC con sipML5

En la página de registro se tiene que ingresar la misma configuración del archivo sip.conf para el teléfono [6003], ingresando la dirección IP del servidor de Asterisk, el nombre de usuario y la contraseña.

A continuación se tiene que especificar el puerto del websocket y el servidor ICE configurado en el archivo rtp.conf, como se muestra en la figura 17.

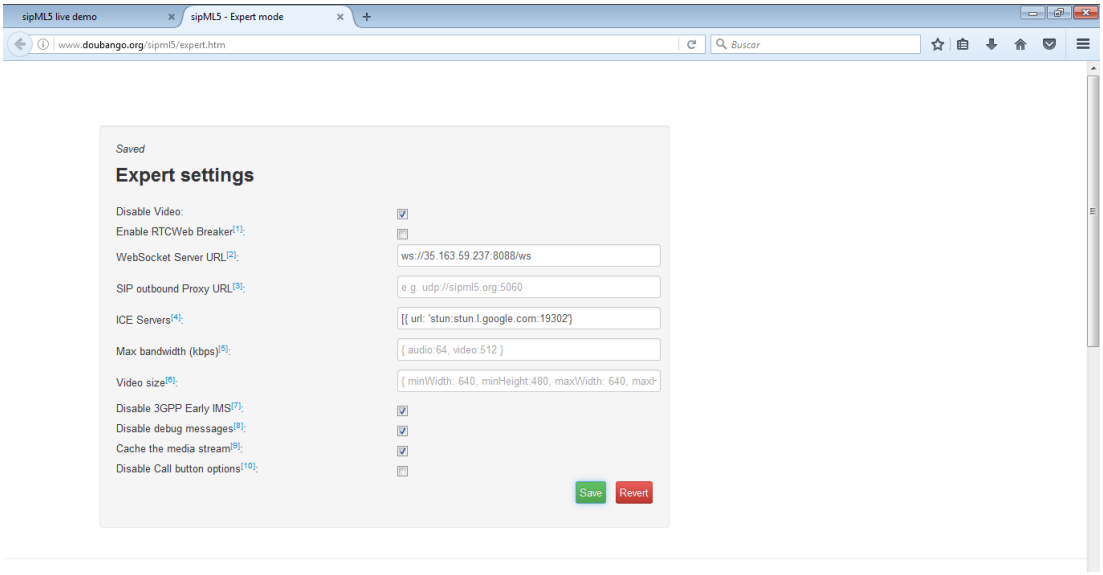


Figura 17. Configuración avanzada del cliente sipML5

En el lado del servidor podemos comprobar que los teléfonos se registran correctamente con el comando

```
sip set debug on
```

La consola nos muestra los mensajes SIP que envían los teléfonos para registrarse como se muestra en las figuras 18 y 19 para el usuario 6001 y 6003 respectivamente.

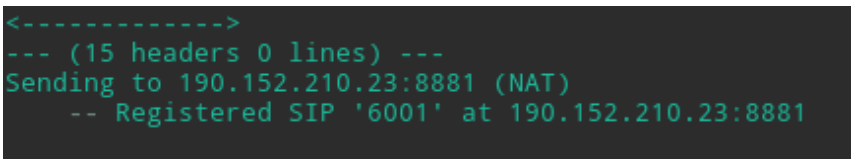


Figura 18. Registro del usuario 6001 en el servidor

Como se puede observar en la figura 17 la consola nos indica que el agente registrado es Z 3.3, que es la versión de Zoiper utilizada, y que se encuentra detrás de un NAT con dirección IP 190.152.210.23 con el puerto 8881.

Lo mismo sucede cuando se registra el cliente remoto de WebRTC, como se muestra en la figura 18.

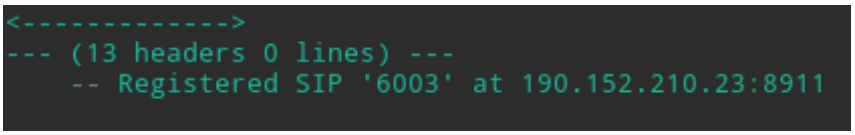


Figura 19. Registro del usuario 6003 en el servidor

Lo que cambia es el agente sipML5-v1 y la dirección IP 190.152.210.23 con el puerto 8911.

Ahora es posible ver que usuarios están registrados en Asterisk con el comando

```
sip show peers
```

Demostrando que los usuarios están registrados correctamente y están en línea, como se muestra en la figura 20.

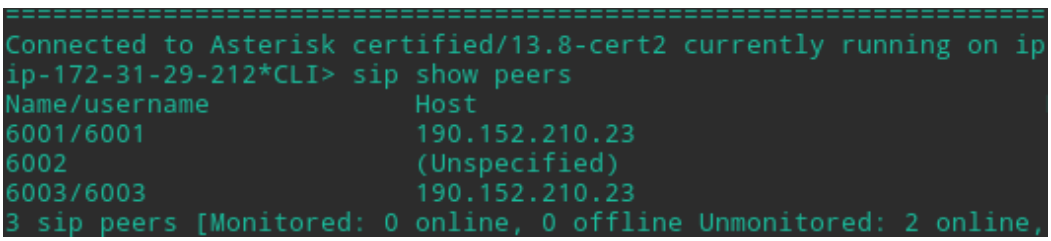


Figura 20. Lista de los usuarios conectados en el servidor

CAPÍTULO IV

RESULTADOS Y ANÁLISIS

Después de la implementación del servidor en la nube, se realizaron diferentes test en diferentes escenarios. El objetivo de cada uno de ellos fue evaluar la calidad de las llamadas replicando condiciones reales en pequeñas o medianas empresas. Al final del capítulo se analiza los resultados obtenidos, los problemas que se presentaron y posibles soluciones.

4.1 Arquitectura

La arquitectura consiste en dos máquinas principales y los usuarios que realizan las llamadas. La máquina principal es el servidor que se encarga de recibir las llamadas y la segunda maquina es la encargada de realizarlas. El servidor se encuentra ubicado en la nube, y la máquina que realiza las llamadas simultáneas se encuentra en la misma subred de los usuarios que evalúan la aplicación, como se ve en la figura 21.

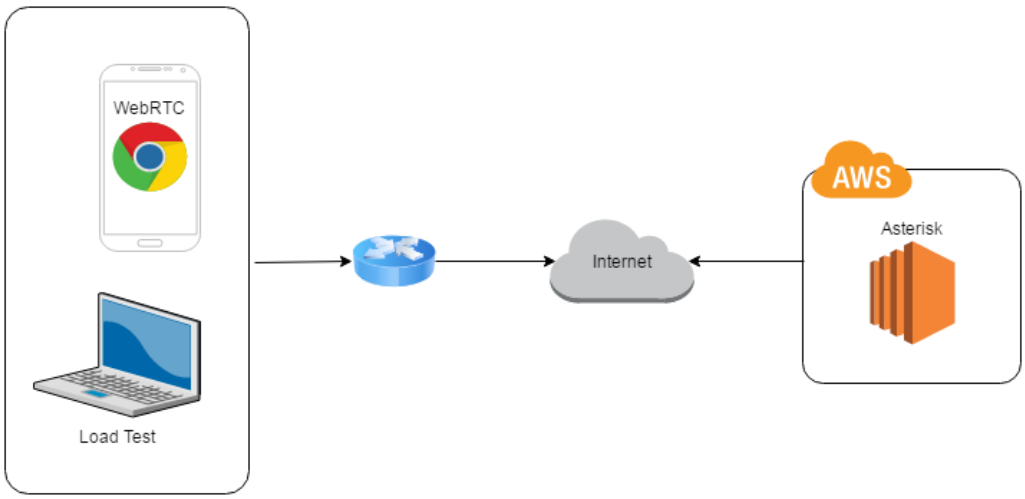


Figura 21. Arquitectura para los escenarios de prueba

En el servidor se configuraron dos métodos de respuesta: el primero es para los usuarios, donde el servidor contesta, marca la extensión deseada y al final corta la llamada; el segundo es para las llamadas simultaneas, donde el servidor contesta, reproduce una música en espera y finaliza la llamada.

La máquina para realizar el load test tiene como sistema operativo LinuxMint 17.2 64-bit y utiliza el programa SIPp que permite realizar múltiples llamadas por medio de escenarios XML.

Especificaciones de hardware:

- Procesador: Intel Core i5-3210M CPU 2.5 GHz x 2
- Memoria: 8 GB de RAM
- Disco duro: 750 GB HDD

Especificaciones de hardware del servidor en una instancia t2.micro:

- Procesador: 1 vCPUs, 2.5 GHz hasta 3.3 GHz, Intel Xeon Family,
- Memoria: 1 GiB
- Disco Duro: Amazon Elastic Block Store (EBS gp2)

4.2 Metodología Escenario 1

Para este escenario se realizaron múltiples llamadas simultáneas desde la computadora y se monitorea cual es el número máximo de llamadas posibles cuando se tienen diferentes velocidades de subida y bajada.

Para limitar el ancho de banda de la red se utiliza el siguiente comando de Linux:

```
sudo wondershaper eth0 1000 100
```

Los parámetros que se utilizan en el comando son:

- **“eth0”** indica la interfaz de red que se va a utilizar
- **“1000”** establece el límite de la velocidad de bajada, en kbps
- **“100”** establece el límite de la velocidad de subida, en kbps

Paulatinamente se va aumentando la capacidad del ancho de banda hasta llegar a los límites contratados al proveedor de Internet, respectivamente 5 Mbps y 0.5 Mbps.

Para realizar el load test se utiliza la herramienta SIPp con el script que se muestra en el anexo 3. Los parámetros que se utilizan en el comando son:

- **“-d”** utilizado para especificar la durada de la llamada, en milisegundos
- **“-r”** utilizado para especificar la tasa de llamadas por segundo
- **“-m”** establece el número máximo de llamadas totales

./sipp -sf /Tesis/Scenario/sipp_uac_No_Register.xml 35.167.91.121
-inf /Tesis/Scenario/register_data_uac_3cp.csv -d 10000 -r 10 -m 100

4.3 Metodología Escenario 2

Para el segundo escenario se realizaron múltiples llamadas simultáneas desde la computadora y se monitorea la carga en el CPU y los parámetros de red en el servidor.

Para realizar el Load test se utiliza la herramienta SIPp con el mismo escenario anterior pero en una red diferente. Se utiliza la red inalámbrica de la Universidad de las Fuerzas Armadas – Espe que está dividida en subredes para diferentes sectores del campus. En esta infraestructura se implementan varios servicios propios de la universidad y está libre el acceso para estudiantes.

4.4 Resultados

Los resultados del escenario 1 se muestran en la figura 22. Se puede observar que el número máximo de llamadas simultáneas es 6 cuando se tiene una velocidad de subida de 500 kbps. El factor predominante entre la velocidad de subida y de bajada, es la primera porque no importa si se aumenta la velocidad de bajada de 1 Mbps a 5 Mbps, el número máximo de llamadas simultaneas está limitado por la velocidad de subida.

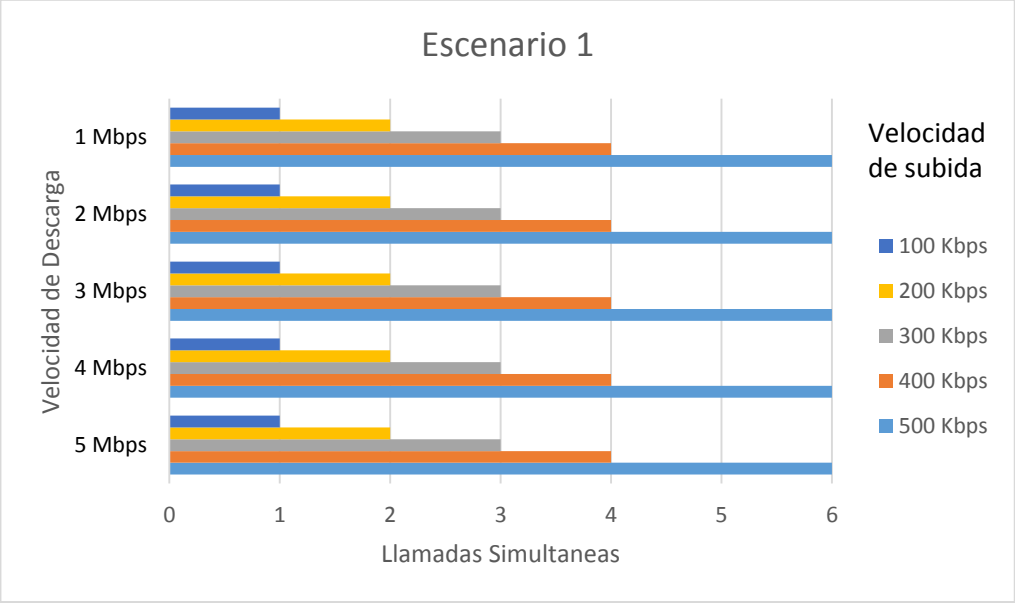


Figura 22. Resultados del escenario 1

Este resultado se respalda calculando el número de llamadas posibles mediante la siguiente formula.

$$Numero\ maximo\ de\ llamadas\ simultaneas = \frac{Velocidad\ de\ subida}{Trama\ del\ codec}$$

Como se demuestra en (Saltos Rodríguez & Chávez Arias, 2015), el códec G711 necesita 80 Kbps por llamada, aproximadamente, por lo tanto:

$$Numero\ maximo\ de\ llamadas\ simultaneas = \frac{500\ kbps}{80\ kbps} = 6.25\ \approx 6$$

Sucesivamente en el mismo escenario con el comando a continuación se analiza las estadísticas de las llamadas.

sip show channelstats

Como resultado se muestra los paquetes enviados y recibidos, los paquetes perdidos y el Jitter en cada llamada, como se muestra en la figura 23.

```
ip-172-31-22-238*CLI> sip show channelstats
Peer             Call ID      Duration Recv: Pack  Lost       (     %) Jitter
190.152.200.56   16a1dc5e378  00:05:53 0000017573  0000000000 ( 0.00%) 0.0000
190.152.200.56   0d9a2e0d-d8  00:05:53 0000017538  0000000000 ( 0.00%) 0.0000

                                      Send: Pack  Lost       (     %) Jitter
2 active SIP channels                 0000017538  0000000014 ( 0.08%) 0.0015
ip-172-31-22-238*CLI>                 0000017562  0000000014 ( 0.08%) 0.0041
```

Figura 23. Estadísticas de una llamada.

Como se puede observar, cuando se tiene dos canales o llamadas se tiene una mínima cantidad de paquetes perdidos, exactamente el 0.08% que no afectan a la calidad percibida por el usuario. El Jitter en el lado del receptor es de 1,5 ms que es muy inferior al valor recomendado de 100 ms.

Este procedimiento se realizó para el número máximo de llamadas simultáneas posibles y los resultados se exponen en la tabla 4.

Tabla 4
Estadísticas del escenario 1

Número de llamadas simultaneas	Paquetes enviados	Paquetes recibidos	Paquetes perdidos (%)	Jitter (ms)
1	17573	17538	0.08	1.5
2	17562	17536	0.15	25
3	32991	32823	0.51	62
4	27329	27097	0.85	84
5	30652	30373	0.91	114
6	30652	30273	1.1	124

Como se puede observar en la tabla 4 el número máximo de llamadas simultáneas es 6, debido a que el Jitter tiene un valor de 124, superior al valor aconsejable de 100 ms. Con el Jitter también aumenta el porcentaje de paquetes perdidos llegando a un valor del 1% deteriorando así la calidad de las llamadas.

Para el escenario 2 se presentan los resultados en la tabla 5. Se puede observar que el servidor llega a su máximo rendimiento con 100 llamadas simultáneas.

Tabla 5
Resultados del escenario 2

Número de llamadas simultaneas	Rendimiento de Asterisk (%)	MOS
10	2	5
20	5	4.5
30	10	4
40	45	4
50	60	3
60	79	3.5
70	79	2
80	88	2
90	100	2
100	100	-

Existe una relación inversa entre el número de llamadas simultáneas y la calidad percibida por los usuarios, como se puede ver en la figura 24 conforme van aumentado las llamadas disminuye la calidad de la misma.

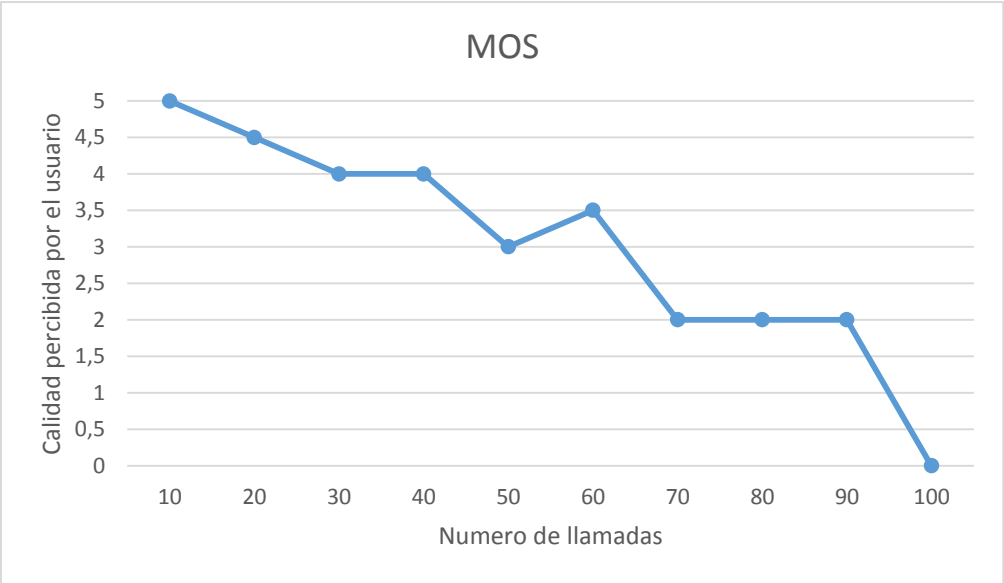
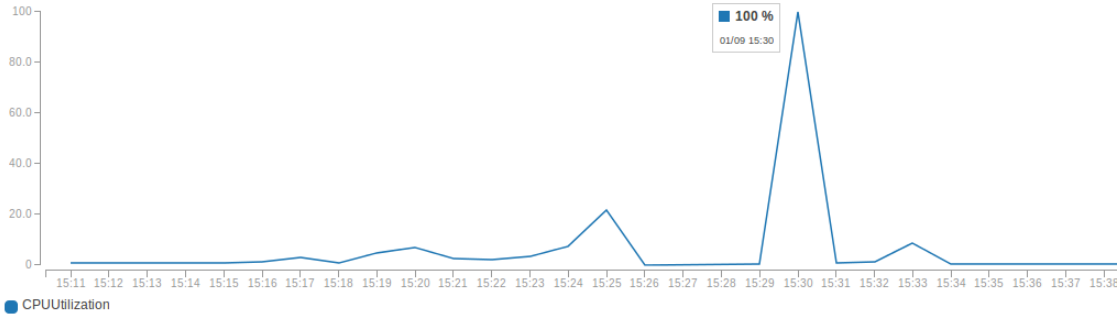


Figura 24. Calidad vs Número de llamadas

Las siguientes imágenes muestran el rendimiento del servidor al momento de realizar el segundo test.



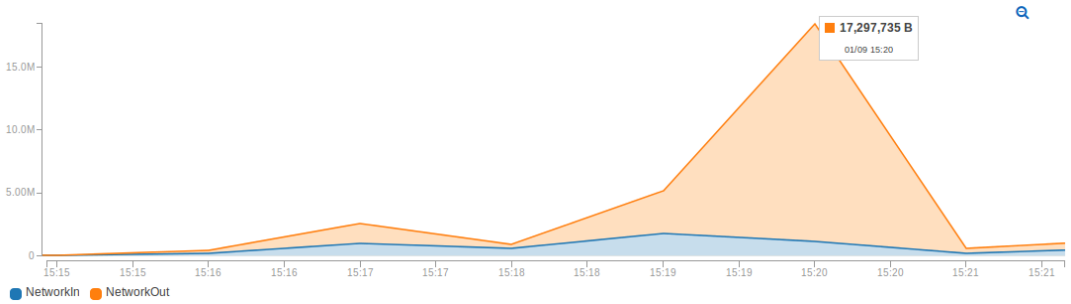


Figura 25. Graficas del rendimiento del servidor en el escenario 2

Como se puede ver en las imágenes el servidor tiene una carga del 100% al momento de realizar 100 llamadas simultáneas. Es aquí donde los usuarios reportan problemas en las llamadas. Sucesivamente el servidor reporta un error interno y es necesario reiniciarlo, como se ve en la figura 26. También se puede observar que el tráfico de salida es muy superior al de entrada debido a que el servidor es el encargado de reproducir la música de espera hacia los usuarios creados en el load test.

```
    -- Executing [6004@from-internal:2] Verbose("SIP/6002-0000016f", "1,### PERF
ORMING SIPP TEST #######") in new stack
 ### PERFORMING SIPP TEST #######
    -- Executing [6004@from-internal:3] MusicOnHold("SIP/6002-0000016f", "defaul
t") in new stack
    -- Started music on hold, class 'default', on channel 'SIP/6002-0000016f'
[Jan  9 15:33:47] ERROR[1373][C-0000016f]: netsock2.c:305 ast_sockaddr_resolve:
getaddrinfo("ip-172-31-22-238", "(null)", ...): System error
ip-172-31-22-238*CLI>
Disconnected from Asterisk server
Asterisk cleanly ending (0).
Executing last minute cleanups
ubuntu@ip-172-31-22-238:~$
```

Figura 26. Error en el servidor en el escenario 2

4.5 Escalamiento

La posibilidad de escalar el servidor en la nube permite incrementar o disminuir la capacidad de una instancia EC2 acorde a una condición predefinida por el administrador. En este caso de estudio se estableció que cuando la carga en el CPU es superior al 50% se agregue una instancia y cuando es inferior al 40% se remueva una, manteniendo siempre disponible una instancia para el servidor. Estas reglas fueron definidas en la plataforma de Amazon mediante un asistente que guía paso a paso al administrador, como se muestra en el anexo 4; el resultado se puede ver en la figura 27.

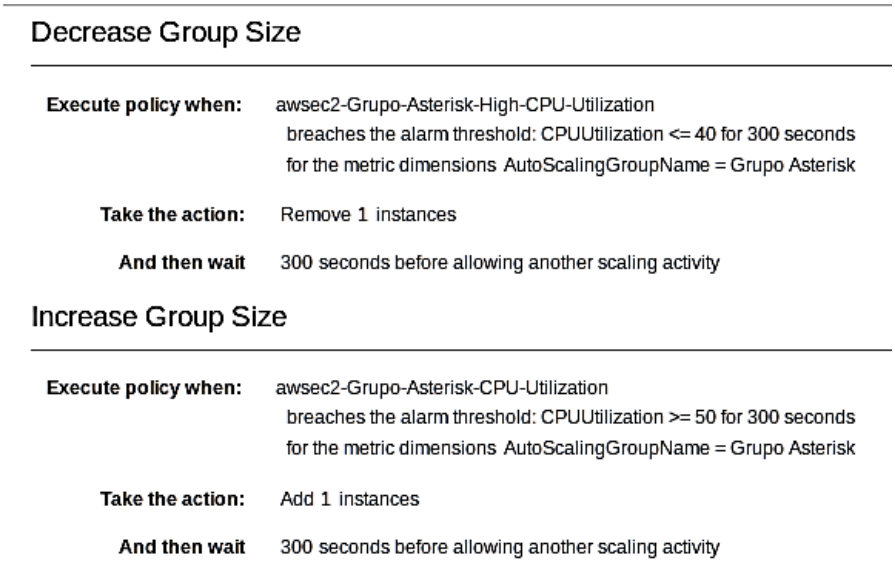


Figura 27. Reglas para escalar el servidor en la nube

En una empresa de telefonía VoIP se puede determinar en qué momento a lo largo del día suceden picos de carga y configurar que por determinados periodos de tiempo, por ejemplo 1 o 2 horas, se aumente una instancias para procesar la carga adicional.

Otra posibilidad que ofrece tener múltiples instancias de un servidor Asterisk es administrarlas mediante un SIP proxy que será el encargado de distribuir las llamadas entre los varios servidores disponibles. Una de las posibilidades puede ser Kamailio, que es software libre; este actuaría como un balanceador de carga y permitiría aumentar el número de llamadas simultáneas que puede soportar el sistema. Una posible implementación se puede observar en la figura 28.

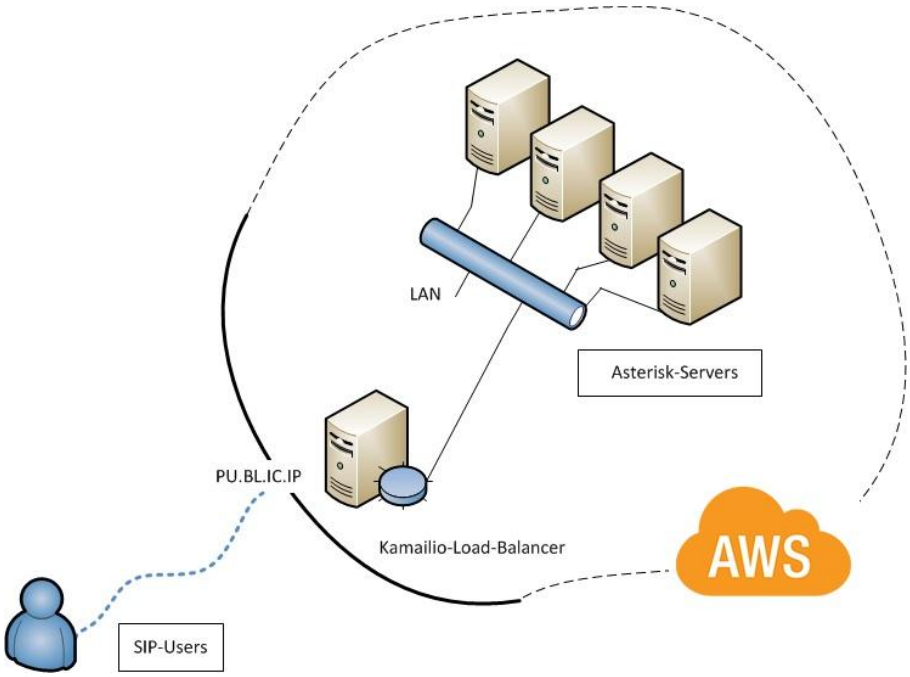


Figura 28. Implementación de un SIP proxy en Amazon

4.6 Seguridad

La tecnología WebRTC utiliza el protocolo RTP, en específico una variante de este llamada SRTP, Secure Real-Time Transport Protocol por sus siglas en ingles. SRTP es una plataforma que permite encriptación y autentificación al usuario, por ende garantiza un medio seguro para realizar llamadas, como se ve en la figura 29. El servidor HTTP integrado en Asterisk es el encargado de administrar los Websockets y restringe el acceso a usuarios que se conecten solo con HTTPS, como se pudo comprobar mediante la página *https://www.doubango.org/sipml5/call.htm?svn=250*. Adicionalmente el API de WebRTC restringe por defecto el acceso al micrófono y la cámara solo a páginas HTTPS agregando un nivel adicional de seguridad para realizar llamadas.

```
<------------->
--- (13 headers 30 lines) ---
Using INVITE request as basis request - 709f31b7-ebd3-5c14-2534-61c8a26af740
Found peer '6003' for '6003' from 190.152.205.80:9354
   == DTLS ECDH initialized (automatic), faster PFS enabled
   == Using SIP RTP CoS mark 5
Found RTP audio format 109
Found RTP audio format 9
Found RTP audio format 0
Found RTP audio format 8
```

Figura 29. Autentificación de un usuario WebRTC mediante DTLS

Al momento de crear una instancia Amazon EC2 se necesita especificar a qué grupo de seguridad pertenece, como se puede ver en el anexo 1. El grupo de seguridad actúa como un firewall que permite decidir qué protocolos y puertos están abiertos a internet. Se puede escoger el grupo de seguridad por default pero se aconseja crear un grupo de seguridad acorde a las necesidades del servidor a implementar.

Existen varios servicios preconfigurados dentro de las opciones, como se puede ver en la figura 30, pero los protocolos relativos a VoIP son TCP, UDP e ICMP (utilizado para realizar pruebas de ping). Para cada protocolo se especifica un puerto específico o un rango.

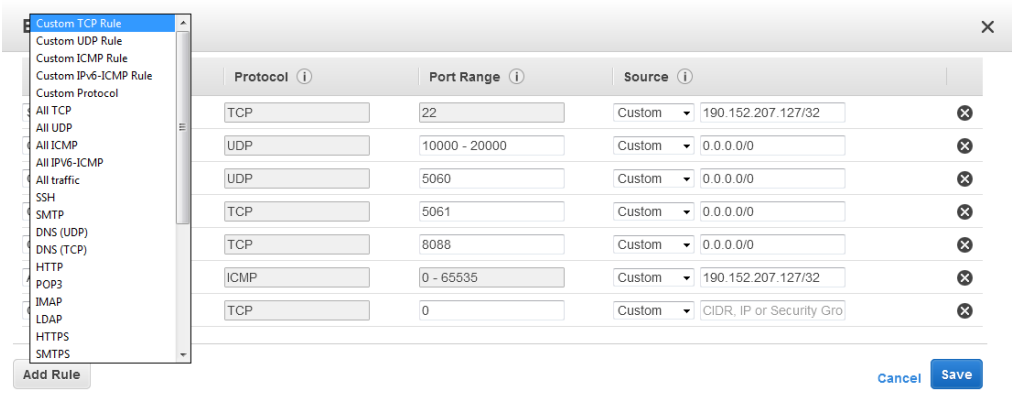


Figura 30. Grupos de seguridad

Lo importante en un grupo de seguridad es la dirección IP de origen, existen dos opciones posibles: “0.0.0.0/0” o una dirección IP en específico, como por ejemplo “190.152.207.127/32”. La primera significa que cualquier persona puede acceder al protocolo y puerto en cuestión, pero esta no es una práctica muy segura debido a que expone el servidor ante cualquier amenaza. La segunda opción nos permite especificar que una sola IP tenga acceso a determinado protocolo. Por ejemplo el protocolo TCP con el puerto 22 solo es accesible desde la dirección segura “190.152.207.127/32” para entrar al servidor mediante el servicio SSH.

Amazon provee diferentes herramientas para incrementar y monitorear la seguridad de un servidor, en este caso de estudio se implementó un firewall virtual mediante los grupos de seguridad asociados a cada instancia que controlan sea el tráfico de llegada que de salida. Sin embargo, se puede agregar una segunda capa de

seguridad mediante una lista de control de acceso a la red, Network Access Control Lists (ACLs), que funciona como un firewall asociado a una subnet en particular (Amazon Web Services, Inc., 2017). Para monitorear las direcciones IP que se conectan al servidor, Amazon provee una herramienta que registra todo flujo de entrada y salida para una instancia, subred o un grupo de seguridad. Esta herramienta es muy útil para diagnosticar que usuario se está conectando o restringir IP no deseadas (Amazon Web Services, Inc., 2017). El diagrama en la figura 31 incluye todo los niveles de seguridad que ofrece Amazon.

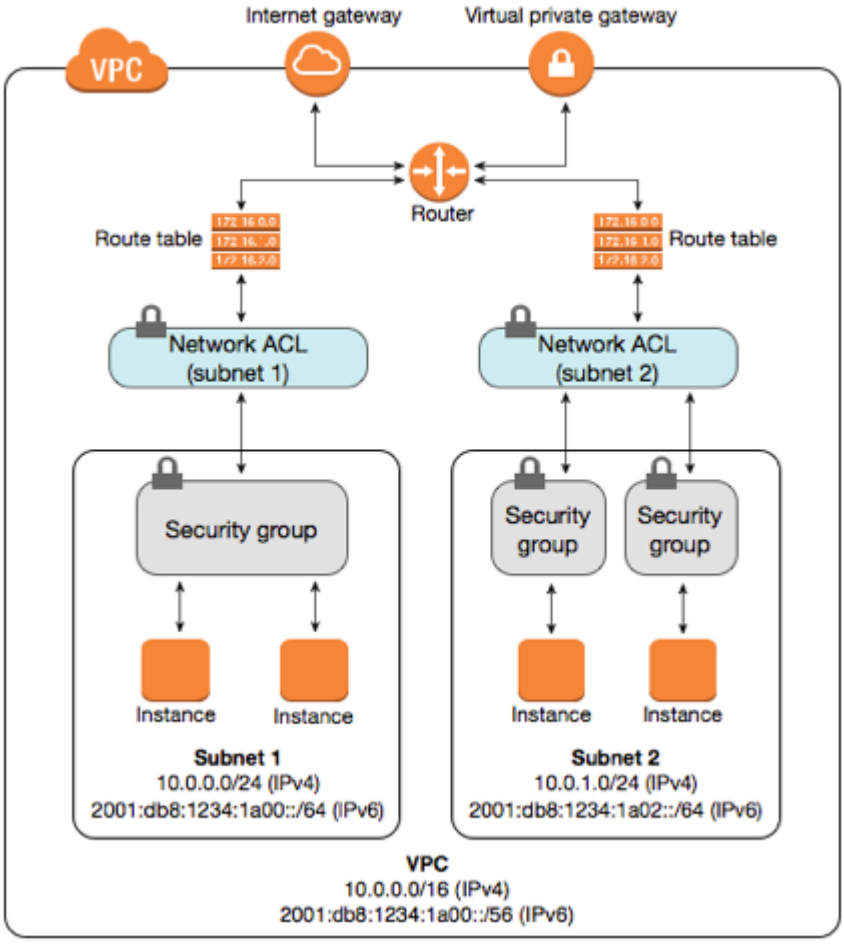


Figura 31. Capas de seguridad en Amazon Web Services

4.7 Costos

Para estimar el costo que tendría implementar un servidor VoIP en una instancia Amazon EC2 se tienen que considerar las siguientes variables:

- **Tiempo de uso:** Se paga por lo que se consume. Por ejemplo se paga por una IP pública desde que se la asigna a una instancia hasta que es liberada nuevamente.
- **Configuración de la maquina:** El precio depende de la maquina implementada: memoria, núcleos y la región en donde está ubicada.
- **Número de instancias:** Se puede implementar múltiples instancias para soportar picos de carga.
- **Trafico:** La cantidad de información que procesa una instancia al mes, sea tráfico de llegada que de salida.

La tabla 6 muestra las características del servidor que se implementó y el uso estimado que tendría en un mes.

Tabla 6
Variables para el cálculo del costo

Variable	Uso estimado	Descripción
Tiempo de uso	24 horas/ día	En promedio se tiene 30.5 días al mes, por ende la instancia corre 732 horas/mes.
Configuración de la maquina	t2.micro	1GB de memoria 1 vCPU 8 GB SDD de almacenamiento
Almacenamiento	SDD	No se necesita almacenamiento adicional
Escalamiento	1	Numero de instancias utilizadas
Trafico	Data In: 5 GB/día Data Out: 50 GB/día	Se estima que un servidor consume aproximadamente 55 GB/día.
Carga mensual	Horas: 732 horas/mes Data: 1677.5 GB/mes	Carga total usada por 24 horas/día por 7 días/semana. Carga total generada por un tráfico de 55 GB/día

El costo total al mes es la suma de todas las variables anteriores, en la tabla 7 se realiza el cálculo estimado para un servidor VoIP en la nube.

Tabla 7
Costo de un servidor Asterisk en Amazon

Variable	Formula	Calculo
Costo de una instancia	Costo per hora de la instancia x Numero de instancias x Tiempo de uso	$ 0.07 x 1 x 732 = $ 204.96
Costo de trafico	Data In (GB) x Costo GB In + Data Out (GB) x Costo GB Out	152.5 x $ 0.00 + 152.5 x 0.12 = $ 182.88
Costo de carga	Tiempo de Uso x Costo + Información Procesada x Costo	732 x $0.025 + 1677.5 x $0.008 = $ 31.72
Costo Total		= $ 419.56

Amazon permite estimar el precio de una instancia mediante una calculadora mensual sencilla, donde se puede evaluar soluciones completas utilizando varios servicios de Amazon Web Services como almacenamiento, base de datos, múltiples instancias etc. Se compara el costo calculado previamente de manera manual con una plantilla de un servidor multimedia, como se puede ver en la figura 32.

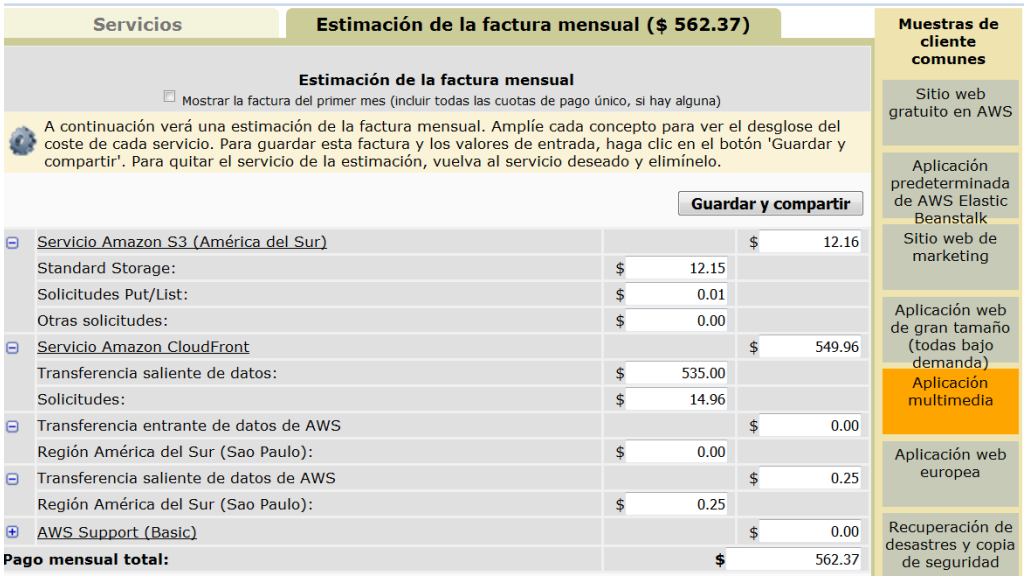


Figura 32. Costo de un servidor multimedia en Amazon Web Services

Como se puede observar el precio, 562.37 dólares, no es muy diferente al calculado de manera manual, 419.56 dólares; el mayor gasto lo representa el trafico saliente.

Dependiendo de las necesidades de la aplicación a implementar, se puede reducir costos utilizando una instancia reservada, estas permiten un ahorro considerable en

comparación con una instancia bajo demanda. Para una instancia reservada se pueden hacer pagos por el periodo de 1 a 3 años, entre mayor es el pago mayor es el descuento obtenido. Adicional a esto Amazon ofrece una capa gratuita de sus servicios que está diseñada para permitir implementar cualquier aplicación de forma gratuita por el periodo de 12 meses, dentro de los límites de uso. Cabe recalcar que el servidor en la nube en este caso de estudio utiliza la capa gratuita de Amazon para su funcionamiento.

CAPÍTULO V

CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

- Los beneficios que ofrece la computación en la nube a cualquier empresa son: bajos costos, agilidad para ampliar sus servicios y flexibilidad. La nube elimina la necesidad de hardware dedicado, ahorrando a las empresas inversiones en infraestructura y mantenimiento, dado que pueden contratar infraestructura como servicio y pagar bajo demanda.
- Se pudo comprobar la interoperabilidad de la tecnología WebRTC con usuarios SIP. Las llamadas se realizaron satisfactoriamente y presentaron una latencia inferior a los 150 ms recomendados para evitar que el usuario se percate de problemas de retorno o delay.
- Se resalta la estabilidad y la funcionalidad de la tecnología WebRTC. En este caso de estudio se implementó un webphone de fácil uso, al cual se puede agregar características adicionales como vídeo llamadas, compartición de pantalla o envió de archivos. WebRTC es la tecnología del futuro y empresas como Facebook o Slack ya la implementaron en sus servicios.
- El análisis en diferentes escenarios permite optimizar los recursos de red en futuras implementaciones, debido a que ya se sabe qué se puede esperar de la red y se puede adelantar hasta qué punto se degrada la calidad de las llamadas dependiendo del rendimiento de la infraestructura. Los diferentes escenarios muestran que existe una relación linear entre las llamadas simultáneas y la degradación de la calidad. Este problema no es debido sólo a los recursos de la red sino que también está relacionado con la capacidad del servidor.
- Mediante el autoescalamiento se puede implementar un SIP proxy que será el encargado de distribuir las llamadas entre varias instancias de Asterisk. El proxy actuaría como un balanceador de carga y permitiría

aumentar el número de llamadas simultáneas que puede soportar el sistema.

- La tecnología WebRTC es una plataforma que permite encriptación y autentificación al usuario, por ende garantiza un medio seguro para realizar llamadas en Internet. Además Amazon ofrece muchas herramientas para proteger sus servidores, como firewall virtuales y restricción de IP. Esto garantiza la seguridad de la información de cualquier empresa.

5.2 Recomendaciones

- Al momento de realizar una llamada es posible que se establezca correctamente pero no se escuche el audio en ninguno de los dos extremos, para evitar este problema se recomienda indicar al servidor cuál es su dirección IP externa y cómo está estructurada la red interna, para que los paquetes no se pierdan en las rutas de entrada o salida.
- Para monitorear de mejor manera el rendimiento del servidor se recomienda activar la opción de monitoreo detallado con intervalos de 1 minuto. Amazon utiliza por default intervalos de 5 minutos.
- Asterisk por default tiene un límite de ficheros abiertos establecido en 1024, cuando se realizan pruebas de llamadas simultaneas es aconsejable aumentar este límite para no tener problemas con los sockets a un valor de 65535.

5.3 Trabajos Futuros

- Comparar precios entre los tres líderes en Cloud Computing es complejo y difícil, debido a que continuamente como estrategia para ganar mercado se reducen costos. De hecho Amazon, Google y Microsoft proporcionan una calculadora para estimar el costo y los ahorros que conlleva implementar una determinada aplicación en su propia infraestructura. Pero no existe una calculadora que compare los costos bajo un mismo

escenario de los tres proveedores, debido a que cada uno tienen características diferentes y beneficios propios, como alianzas estratégicas. Por ende el único modo de comparar el costo real de un servidor en la nube es directamente implementar otro servidor en la infraestructura de Google o Microsoft.

- Como trabajos futuros se plantea implementar a la página web de una empresa la tecnología WebRTC para mejorar la atención al usuario y crear una línea directa donde el usuario pueda recibir ayuda. Un ejemplo de esto puede ser la aplicación de Amazon MayDay donde el usuario llama al servicio al cliente y es atendido de forma interactiva mediante comandos en su propia pantalla, todo esto gracias a la tecnología WebRTC.
- Uno de los retos siempre constantes en VoIP es mantener una buena calidad de llamada con insuficiente ancho de banda. Una de las posibles soluciones es definir el códec de audio apropiado a las condiciones del usuario. WebRTC utiliza el nuevo códec Opus que es totalmente abierto y está diseñado para transmisiones de audio en Internet. Se plantea realizar una comparación del códec Opus con los más utilizados actualmente. Asterisk ya tiene soporte para Opus y es el encargado de realizar la conversión entre códecs, pero este procesamiento representa una carga adicional al servidor.

REFERENCIAS

Akhawe, D. M. (2014). Towards High Assurance HTML5 Applications.

Amazon. (Diciembre de 2016). Obtenido de https://aws.amazon.com/es/

Amazon Web Services, Inc. (2017). *Amazon Virtual Private Cloud.* Obtenido de http://docs.aws.amazon.com/es_es/AmazonVPC/latest/UserGuide/VPC_ACLs.html

Amazon Web Services, Inc. (2017). *Amazon Virtual Private Cloud.* Obtenido de http://docs.aws.amazon.com/es_es/AmazonVPC/latest/UserGuide/flow-logs.html

Campbell, B., & Sparks, R. (2001). Control of service context using SIP Request-URI.

Christiansen, T., Giotis, I., & Mathur, S. (2004). Performance evaluation of voip in different settings. *Project Papers of University of Washington. Consulta*.

Deniz, B. (Diciembre de 2016). *WebRTC on Mobile*. Obtenido de http://www.slideshare.net/BuraDenizCSM/webrtc-voxxed?ref=http://busradeniz.com/webrtc-nedir/

Gómez, J., & Gil, F. (2009). *VoIP y Asterisk Redescubriendo la telefonía.* México: Alfaomega.

ITU, T. (1988). Recommendation G. 711. *Pulse Code Modulation (PCM) of voice frequencies*.

ITU-T. (2003). Recommendation G. 114. *One-Way Transmission Time, Standard G*.

Joskowicz, J. (2015). *Señalización H. 323 y SIP*.

Leong, L. a. (2016). *Magic Quadrant for Cloud Infrastructure as a Service, Worldwide.* Gartner.

Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., . . . Schooler, E. (2010). *RFC 3261: SIP: Session initiation protocol.* Obtenido de www.ietf.org/rfc/rfc3261.txt

Sacker, S., Santaiti, M., & Spence, C. (2006). The Business Case for Enterprise VoIP. *Intel Corporation*.

Saltos Rodríguez, D. J., & Chávez Arias, M. J. (2015). *Evaluación de un prototipo de central Asterisk implementado sobre la tarjeta Raspberry PI.* Universidad de las Fuerzas Armadas ESPE.

Silvana, G. P., & Schulzrinne, H. (2008). SIP and 802.21 for Service Mobility and Pro-active Authentication. *Communication Networks and Services Research Conference, 2008. CNSR 2008. 6th Annual*, (págs. 176--182).

SIPp. (Diciembre de 2016). Obtenido de http://sipp.sourceforge.net/

Sipwise . (Diciembre de 2016). *RTC:engine WebRTC Telecommunication Api*. Obtenido de https://www.sipwise.com/products/rtc-engine-webrtc-telecommunication-api/

Van Meggelen, J., Madsen, L., & Smith, J. (2007). *Asterisk: the future of telephony.* O'Reilly Media, Inc.

Verkasalo, H. (2006). Emerging trends in the Mobile VoIP business. *Seminar on Networking Business, Helsinki University of Technology*.

Vogt, C., Werner, M. J., & Schmidt, T. C. (2013). Leveraging WebRTC for P2P content distribution in web browsers. *21st IEEE International Conference on Network Protocols (ICNP), 2013* (págs. 1--2). IEEE.

Zeidan, A., Lehmann, A., & Trick, U. (2014). WebRTC enabled multimedia conferencing and collaboration solution. *WTC 2014; World Telecommunications Congress 2014*, (págs. 1--6).

Zhang, Q. a. (2010). Cloud computing: state-of-the-art and research challenges. *Journal of internet services and applications, 1*, 7--18.

Zhang, Q., Cheng, L., & Boutaba, R. (2010). Cloud computing: state-of-the-art and research challenges. *Journal of internet services and applications*.

Znaty, S., Dauphin, J.-L., & Geldwerth, R. (2005). SIP: Session Initiation Protocol. *SIP: Session Initiation Protocol*.