



**UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERÍA
CARRERA DE INGENIERÍA EN TECNOLOGÍAS DE LA
INFORMACIÓN**

**Prototipo de un Sistema en GNS3 con la Integración de Asterisk y
PostgreSQL sobre IPv6 para Consulta de Notas Académicas**

**Trabajo de Titulación para optar al título de Ingeniero en Tecnologías de
la Información**

Autor:

Salcan Chisaguano Cristian Fernando

Tutor:

Ing. Luis Gonzalo Allauca Peñafiel, MsC.

Riobamba, Ecuador. 2024

DECLARATORIA DE AUTORÍA

Yo, Cristian Fernando Salcan Chisaguano, con cédula de ciudadanía 0604938514, autor del trabajo de investigación titulado: Prototipo de un Sistema en GNS3 con la Integración de Asterisk y PostgreSQL sobre IPv6 para Consulta de Notas Académicas, certifico que la producción, ideas, opiniones, criterios, contenidos y conclusiones expuestas son de mí exclusiva responsabilidad.

Asimismo, cedo a la Universidad Nacional de Chimborazo, en forma no exclusiva, los derechos para su uso, comunicación pública, distribución, divulgación y/o reproducción total o parcial, por medio físico o digital; en esta cesión se entiende que el cesionario no podrá obtener beneficios económicos. La posible reclamación de terceros respecto de los derechos de autor de la obra referida, será de mi entera responsabilidad; librando a la Universidad Nacional de Chimborazo de posibles obligaciones.

En Riobamba, 31 de mayo del 2024.



Cristian Fernando Salcan Chisaguano

C.I: 0604938514

DICTAMEN FAVORABLE DEL PROFESOR TUTOR



Dirección
Académica
VICERRECTORADO ACADÉMICO



ACTA FAVORABLE - INFORME FINAL DEL TRABAJO DE INVESTIGACIÓN

En la Ciudad de Riobamba, a los 26 días del mes de Abril de 2024, luego de haber revisado el Informe Final del Trabajo de Investigación presentado por el estudiante **Cristian Fernando Salcán Chisaguano** con CC: **0604938514**, de la carrera Ingeniería en Tecnologías de la Información y dando cumplimiento a los criterios metodológicos exigidos, se emite el **ACTA FAVORABLE DEL INFORME FINAL DEL TRABAJO DE INVESTIGACIÓN** titulado "**Prototipo de un Sistema en GNS3 con la Integración de Asterisk y PostgreSQL sobre IPv6 para Consulta de Notas Académicas**", por lo tanto se autoriza la presentación del mismo para los trámites pertinentes.



Mgs. Gonzalo Allauca Peñafiel
TUTOR


CERTIFICADO DE LOS MIEMBROS DEL TRIBUNAL

CERTIFICADO DE LOS MIEMBROS DEL TRIBUNAL

Quienes suscribimos, catedráticos designados Miembros del Tribunal de Grado para la evaluación del trabajo de investigación **Prototipo de un Sistema en GNS3 con la Integración de Asterisk y PostgreSQL sobre IPv6 para Consulta de Notas Académicas**, presentado por **Cristian Fernando Salcan Chisaguano**, con cédula de identidad número **0604938514**, bajo la tutoría de **MsC. Luis Gonzalo Allauca Peñafiel**; certificamos que recomendamos la **APROBACIÓN** de este con fines de titulación. Previamente se ha evaluado el trabajo de investigación y escuchada la sustentación por parte de su autor; no teniendo más nada que observar.

De conformidad a la normativa aplicable firmamos, en Riobamba a 31 días del mes de mayo de 2024.

Estela Narváez, PhD.
PRESIDENTE DEL TRIBUNAL DE GRADO



FIRMA

Diego Reina, Mgs.
MIEMBRO DEL TRIBUNAL DE GRADO



FIRMA

Fernando Molina, PhD.
MIEMBRO DEL TRIBUNAL DE GRADO



FIRMA

CERTIFICADO ANTIPLAGIO

Original



Dirección
Académica
VICERRECTORADO ACADÉMICO



CERTIFICACIÓN

Que, **SALCAN CHISAGUANO CRISTIAN FERNANDO** con CC: **0604938514**, estudiante de la Carrera **Ingeniería en Tecnologías de la Información**, Facultad de **Ingeniería**; ha trabajado bajo mi tutoría el trabajo de investigación titulado "**Prototipo de un Sistema en GNS3 con la Integración de Asterisk y PostgreSQL sobre IPv6 para Consulta de Notas Académicas**", cumple con el **5 %**, de acuerdo al reporte del sistema Anti plagio **TURNITIN**, porcentaje aceptado de acuerdo a la reglamentación institucional, por consiguiente autorizo continuar con el proceso.

Riobamba, **13 de mayo de 2024**



Mgs. Gonzalo Allauca Peñafiel
TUTOR

DEDICATORIA

Con todo mi cariño y gratitud, este logro está dedicado a mi amada familia, en especial a mis queridos padres, madre y tías. Su apoyo incondicional, comprensión y sacrificio han sido el faro que ilumina mi camino hacia la realización profesional.

Agradezco infinitamente su fe en mí, su constante aliento y el inquebrantable respaldo que me han brindado en cada paso de este camino. Son mi mayor inspiración y el motor que impulsa mis sueños hacia la realidad.

También quiero extender mi reconocimiento a mis compañeros de clase y a mis estimados profesores. La colaboración desinteresada y la generosidad en compartir conocimientos de ambos han sido pilares fundamentales en este viaje académico.

A todas las personas que directa o indirectamente con sus palabras, ánimos y apoyo siempre me incentivaron a no claudicar y esforzarme para poder cumplir esta meta, por todo, ESTE LOGRO TAMBIÉN ES SUYO.

C.F.S.CH

AGRADECIMIENTO

Le agradezco a Dios por poder alcanzar uno de mis objetivos.

A mi familia que siempre ha estado apoyándome incondicionalmente y aconsejando que no desmaye para poder alcanzar la meta.

Deseo agradecer a mi tutor de tesis, al MsC. Luis Gonzalo Allauca Quiero expresar mi más sincero agradecimiento a nuestro tutor de tesis. Su guía, experiencia y dedicación fueron fundamentales para el desarrollo de este trabajo. Sus valiosos consejos y agudeza me permitieron mantenerme centrado y alcanzar mi objetivo académico. Estoy sinceramente agradecido por su paciencia y su compromiso inquebrantable.

ÍNDICE GENERAL

DECLARATORIA DE AUTORÍA	
DICTAMEN FAVORABLE DEL PROFESOR TUTOR	
CERTIFICADO DE LOS MIEMBROS DEL TRIBUNAL	
CERTIFICADO ANTIPLAGIO	
DEDICATORIA	
AGRADECIMIENTO	
ÍNDICE DE TABLAS	
ÍNDICE DE FIGURAS	
RESUMEN	
ABSTRACT	
CAPÍTULO I. INTRODUCCIÓN	16
1.1. Antecedentes	16
1.2. Planteamiento del Problema.....	19
1.2.1. Problema y Justificación	19
1.2.2. Formulación del Problema	20
1.3. Objetivos	21
1.3.1. Objetivo General	21
1.3.2. Objetivos Específicos	21
CAPÍTULO II. MARCO TEÓRICO	22
2.1. Prototipo de sistema	22
2.1.1. Características de un prototipo de sistema	22
2.2. GNS3.....	22
2.3. Asterisk	24
2.4. PostgreSQL	25
2.5. IPV6	27
2.6. Integración de plataformas.....	28
2.7. WebServices	29
2.8. AGI (Asterisk Gateway Interface).....	30
2.9. ODBC (Open Database Connectivity).....	31
2.10. Rendimiento del sistema	32
2.11. SIPp.....	32

2.13.	Máquinas virtuales (MV)	34
2.14.	Virtual Box	35
CAPÍTULO III. METODOLOGIA		36
3.1.	Tipo de Investigación	36
3.2.	Diseño de la Investigación	36
3.3.	Técnicas de recolección de Datos	36
3.4.	Población de estudio y tamaño de muestra	37
3.5.	Métodos de análisis y procesamiento de datos	37
3.6.	Identificación de variables	38
3.6.1.	Variable Dependiente	38
3.6.2.	Variable Independiente	38
3.7.	Operacionalización de Variables	38
3.8.	Procedimiento de implementación	40
3.9.	Desarrollo	42
3.9.1.	Arquitectura funcional del prototipo	42
3.9.2.	Topología de red IPv6 del prototipo en GNS3	43
3.9.3.	Implementación de servidor PostgreSQL con IPV6	44
3.10.	Implementación de Servidor Asterisk	45
3.10.1.	Instalación de TEXT TO SPEECH (TTS)	46
3.10.2.	Creación de los canales SIP	46
3.10.3.	Plan de marcado	47
3.11.	Integración del Sistema con ODBC (Escenario I)	48
3.12.	Integración del Sistema con WebService (Escenario II)	49
3.13.	Implementación de Cliente Softphone	50
CAPÍTULO IV. RESULTADOS Y DISCUSIÓN		51
4.1.	Resultados de Integración de Plataformas	51
4.2.	Resultados de la Implementación del Prototipo	51
4.3.	Pruebas de Desempeño	52
4.3.1.	Pruebas y Resultados Escenario I	53
4.3.2.	Pruebas y Resultados Escenario II	56
4.4.	Discusión	58
CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES		59
5.1.	Conclusiones	59

5.2. Recomendaciones.....	60
BIBLIOGRAFÍA.....	61
ANEXO.....	65

ÍNDICE DE TABLAS

Tabla 1. Operacionalización de las Variables	39
Tabla 2. Comparativa de Tecnologías de Integración	51
Tabla 3. Recopilación de resultados obtenidos del escenario 1 hasta 1000 llamadas	55
Tabla 4. Recopilación de resultados obtenidos del escenario 2 hasta 1000 llamadas	57

ÍNDICE DE FIGURAS

Figura 1. Entorno de trabajo de GNS3 con un diagrama de red	23
Figura 2. Asterisk y sus diferentes integraciones.....	25
Figura 3. Características de PostgreSQL.....	26
Figura 4. Procedimiento de comunicación de un Webservice.....	29
Figura 5. Tráfico de llamadas entre UAC y UAS de SIPp.....	32
Figura 6. Entorno del software Softphone Zoiper.....	34
Figura 7. Diagrama de pruebas del escenario 1 de integración utilizando ODBC.....	37
Figura 8. Diagrama de pruebas del escenario 2 de integración utilizando Webservice y AGI	38
Figura 9. Arquitectura funcional del prototipo.....	43
Figura 10. Topología de red ipv6 del prototipo	43
Figura 11. Configuración de direcciones de red estática con IPv6	44
Figura 12. Configuración de PostgreSQL con IPv6.....	44
Figura 13. Diseño de base de datos relacional sistema académico.	45
Figura 14. Configuraciones de Asterisk con IPv6.....	46
Figura 15. Configuración básica de servidor festival.....	46
Figura 16. Configuración de extensiones SIP en Asterisk 13.38	47
Figura 17. Archivo del plan de marcado consulta_db.....	47
Figura 18. Archivo del plan de marcado consulta_webservice.....	48
Figura 19. Conexión de PostgreSQL con Asterisk a través de ODBC	48
Figura 20. Configuración para la conexión de una base de datos con ODBC	49
Figura 21. Funciones para la extracción de registros de la base de datos con ODBC	49
Figura 22. Archivo PHP AGI para la búsqueda dentro de la base de datos.....	50
Figura 23. Configuración de extensión y dirección del dominio del servicio.....	50
Figura 24. Pruebas de funcionamiento del sistema.....	52
Figura 25. Esquema de red para pruebas de estrés con SIPp	53
Figura 26. Uso de recursos pretest escenario 1	53
Figura 27. Respuesta software SIPp en 1000 llamadas escenario 1	54
Figura 28. Respuesta de log de SIPp para calcular la latencia del escenario 1	55
Figura 29. Prueba de uso de recursos con 1000 llamadas en escenario 1	55
Figura 30. Estado de recursos del servidor pretest escenario 2.....	56
Figura 31. Resultado de prueba estrés 100 llamadas en escenario 2.....	56
Figura 32. Respuesta de log de SIPp para calcular la latencia del escenario 2.....	57
Figura 33. Prueba de Uso de recursos del servidor Asterisk durante la prueba de estrés del escenario 2.....	57
Figura 35. Topología de red IPv6 del prototipo	65
Figura 36. Características de la imagen del router cisco c3725	66
Figura 37. Configuración ipv6 red de pc1.....	66
Figura 38. Respuesta ping pc1 a pc2.....	67
Figura 39. Respuesta ping pc1 a pc3.....	67
Figura 40. Configuración IPv6 windows 7 – softphone	68
Figura 41. Configuración IPv6 Asterisk - servidor VoIP.....	68

Figura 42. Configuración IPv6 Ubuntu - servidor PSQL	69
Figura 43. Configuración de máquina virtual Ubuntu – PostgreSQL.....	69
Figura 44. Configuración de redes de escucha en PostgreSQL	70
Figura 45. Configuración de dirección de servidor PostgreSQL	71
Figura 46. Verificación de base de datos " Sistema_Academico " en PSQL.....	71
Figura 47. Tabla Notas_Estudiantes de la base de datos Sistema_Academico PSQL	72
Figura 48. Tabla Materias de base de datos Sistema_Academico	72
Figura 49. Configuración máquina virtual servidor Asterisk.....	73
Figura 50. Verificación de conexión de red y configuración de protocolo IPv6	73
Figura 51. Configuración de archivo manager.conf.....	74
Figura 52. Configuración de archivo sip_general_additional.conf.....	74
Figura 53. Comando para instalación de TTS Festival	75
Figura 54. Configuración básica de servidor Festival.....	75
Figura 55. Status de servidor Festival activo	75
Figura 56. Configuración de extensión PJSIP 1888.....	76
Figura 57. Parámetros de ingreso para autorización de enlace PJSIP.....	76
Figura 58. Configuración de máquina virtual softphone - windows 7.....	77
Figura 59. Configuración de extensión y dirección del dominio del servicio.....	77
Figura 60. Interfaz de Llamada Zoiper.....	78
Figura 61. Conexión del softphone zoiper dentro de la interfaz de Asterisk	78
Figura 62. Creación de objeto de drivers odbc psql.....	79
Figura 63. Conexión de ODBC con la base de datos Sistemas_Academico.....	79
Figura 64. Comprobación de comunicación entre PSQL y Asterisk por ODBC.....	80
Figura 65. Habilitación de inicio de objeto Sistema_Academico_PSQL	80
Figura 66. Generación de funciones de llamada ODBC en archivo <i>func_odbc.conf</i>	81
Figura 67. Dialplan escenario 1 - ingreso de Datos	82
Figura 68. Dialplan menú de selección de visualización de notas	82
Figura 69. Dialplan respuesta de extensión menú 2.....	83
Figura 70. Respuesta en Asterisk de Llamada con CI=0604938514	83
Figura 71. Respuesta en Asterisk para la consulta por código.....	84
Figura 72. Dialplan del escenario 2 mediante AGI.....	85
Figura 73. Dialplan selección de menú escenario 2.....	86
Figura 74. Archivo PHP AGI para la búsqueda dentro de la base de datos.....	87
Figura 75. Código PHP para la conexión con la BD.....	87
Figura 76. Respuesta del sistema escenario 2	88
Figura 77. Respuesta selección de línea dos para búsqueda de todas las asignaturas.....	88

RESUMEN

En la presente investigación se implementó un prototipo de un sistema en GNS3 para la integración de Asterisk y PostgreSQL sobre IPv6 para consulta de notas académicas, se inicia investigando las tecnologías que permitan esta integración y lograr así implementar el mencionado prototipo y su posterior evaluación de su rendimiento, utilizando dos escenarios funcionales; el primero a través de la conexión directa de Asterisk a la base de datos PostgreSQL utilizando ODBC y el segundo a través de la integración a la base de datos utilizando Webservice con el módulo AGI de Asterisk. Se utiliza como usuario final el softphone Zoiper para las pruebas funcionales de consulta de notas a través de la implementación de un IVR en Asterisk. El tipo de investigación es cuantitativa, puesto que se han definido indicadores como el tiempo de respuesta, la latencia y el uso de recursos para medir el rendimiento en los dos escenarios de integración, empleando la herramienta de esfuerzo SIPp. Al comparar las dos opciones de integración para la consulta de notas académicas, se encontró que la conexión directa a la base de datos ofrece un mejor rendimiento en términos de tiempo de respuesta y latencia, con diferencias de 9.15 ms y 1.8 ms respectivamente. Además, se observó una ligera diferencia en el uso de recursos del sistema, con un 1.74% más de uso de CPU y un mínimo aumento del 0.04% en el uso de memoria RAM en comparación con la integración a través de Webservice AGI. Estos hallazgos ofrecen información técnica importante para aplicar esta solución de integración a distintos ámbitos educativos o en su defecto adaptar a otros requerimientos de concurrencia en áreas distintas a la educativa.

Palabras claves: Asterisk, PostgreSQL, IPV6, ODBC, AGI, SIPp

ABSTRACT

ABSTRACT

In the present investigation, a system prototype was implemented in GNS3 to integrate Asterisk and PostgreSQL over IPv6 for consulting academic notes. It begins by investigating the technologies that allow this integration and thus achieving the implementation of the prototype and its subsequent evaluation of its performance, using two functional scenarios: the first through the direct connection of Asterisk to the PostgreSQL database using ODBC and the second through integration to the database using Webservice with the Asterisk AGI module. The Zoiper softphone is used as the end user for functional tests to consult notes through implementing an IVR in Asterisk. The type of research is quantitative since indicators such as response time, latency, and resource use have been defined to measure performance in the two integration scenarios using the SIPp effort tool. When comparing the two integration options for consulting academic notes, it was found that the direct connection to the database offers better performance in terms of response time and latency, with differences of 9.15 ms and 1.8 ms, respectively. Additionally, a slight difference in system resource usage was observed, with 1.74% more CPU usage and a minimal 0.04% increase in RAM usage compared to the integration of Webservice AGI. These findings offer important technical information to apply this integration solution to different educational areas or, failing that, adapt it to other concurrency requirements in areas other than education.

Keywords: Asterisk, PostgreSQL, IPV6, ODBC, AGI, SIPp.



Firmado electrónicamente por:
DARIO JAVIER
CUTIOPALA LEON

Reviewed by:
Mg. Dario Javier Cutiopala Leon
ENGLISH PROFESSOR
c.c. 0604581066

CAPÍTULO I. INTRODUCCIÓN

En la actualidad, la era digital ha impulsado el uso cada vez más generalizado de aplicaciones que operan bajo el protocolo IPv6, marcando un hito significativo en la evolución tecnológica. Esta evolución conlleva la integración de diversas tecnologías, transformando la manera en que las organizaciones administran sus comunicaciones y datos. La integración facilitada por el protocolo IPv6 entre las Private Branch Exchange (PBX) y los sistemas de gestión de bases de datos no solo optimiza la administración de la información y las interacciones comunicativas, sino que también desencadena nuevas oportunidades para mejorar la eficiencia operativa. Este enfoque promueve una infraestructura de red más flexible y adaptable, fomentando así una mayor colaboración y agilidad en las comunicaciones en diversos entornos, desde académicos hasta empresariales.

Por lo anteriormente expuesto, la presente investigación hace referencia a la integración de Asterisk y PostgreSQL sobre protocolo IPv6. Este proyecto se centra en la creación de un prototipo utilizando un simulador de red como GNS3 para evaluar la integración de estas tecnologías. Por ello, Asterisk es reconocido por su papel como sistema de telefonía IP de código abierto, y PostgreSQL como un sistema de gestión de bases de datos robusto, se integran con el objetivo de permitir la consulta a una base de datos, utilizando dos escenarios de integración. Se utiliza dos tablas en la base de datos con información académica para la ejecución de las pruebas y el análisis de resultados, proyectando así el uso de IPv6 en entornos académicos o en cualquier otro entorno. El primer escenario de integración de las plataformas de Voz y Base de Datos se realiza con una conexión directa desde Asterisk hacia la base de datos PostgreSQL, mientras que en el segundo escenario se utiliza una conexión a través del uso de WebService con el módulo AGI de Asterisk.

La selección de GNS3 como plataforma para el desarrollo del prototipo permite una simulación controlada y precisa, previo su implementación en entornos de producción reales. Esta investigación permite validar la funcionalidad de la integración propuesta, y proporcionar una comprensión de las alternativas de integración de Asterisk y PostgreSQL sobre IPv6 y su rendimiento.

1.1. Antecedentes

El presente proyecto se sustentó en avances tecnológicos, específicamente en el contexto de la integración de Asterisk y PostgreSQL sobre IPv6. Para contextualizar esta iniciativa, se exploran antecedentes relevantes que han sentado las bases para la convergencia de estas tecnologías en el ámbito académico y de gestión de datos.

1.1.1. Evolución de IPv6 y su aplicación práctica

La adopción gradual de IPv6 como sucesor de IPv4 ha sido un fenómeno en constante evolución. Ante la inminente escasez de direcciones IPv4, la transición a IPv6 se ha vuelto imperativa. Investigaciones previas destacan la importancia de configurar sistemas, como PostgreSQL, en entornos IPv6 para aprovechar al máximo las capacidades de direccionamiento avanzadas y garantizar la sostenibilidad a largo plazo de las infraestructuras de red (Zhang, Li, & Zhang, 2020).

1.1.2. Asterisk como solución de Telefonía IP

La literatura técnica, ejemplificada en "Asterisk: The Definitive Guide" ha consolidado a Asterisk como un pilar en el ámbito de las comunicaciones. Su papel como Private Branch Exchange (PBX) de código abierto lo posiciona como una solución versátil para la telefonía IP. Sin embargo, la exploración específica de su integración con bases de datos en entornos IPv6 constituye una dirección menos explorada y un enfoque clave de este proyecto (empleadores de red, como GNS3; ha sido una tendencia en el desarrollo y prueba de prototipos antes de la implementación en entornos de producción. Por consiguiente, la comunidad de GNS3, respaldada por recursos como su sitio web www.gns3.com, ha establecido esta plataforma como una herramienta esencial para la creación de entornos simulados que imitan condiciones de red reales. Esto permite evaluar la integración de tecnologías, como Asterisk y PostgreSQL sobre IPv6, en un escenario controlado antes de su despliegue efectivo.

1.1.3. Optimización de rendimiento en sistemas académicos

En el ámbito académico, la gestión óptima de datos, especialmente en consultas de notas académicas, ha sido un desafío constante. Han explorado la integración de PostgreSQL con Asterisk para aplicaciones de telefonía en tiempo real. Sin embargo, la evaluación específica del rendimiento en el contexto de IPv6 y la consulta de notas académicas se presenta como un nicho donde se busca aportar contribuciones significativas.

1.1.4. Estudios de Caso Relevantes

Se realizó la revisión de tres casos de estudio y proyectos que han implementado exitosamente la integración de Asterisk con Gestores de Bases de Datos y la implementación de Asterisk sobre protocolo IPv6. Esto proporcionó una perspectiva práctica de cómo otras organizaciones han abordado la integración y los beneficios que han obtenido.

Primer caso de estudio: Universidad de California, Berkeley

La Universidad de California, Berkeley, implementó Asterisk sobre IPv6 en 2016 para mejorar la conectividad y la seguridad de sus sistemas de telefonía IP. La implementación se realizó en dos fases: la primera fase consistió en la migración de los servidores Asterisk existentes a IPv6, y la segunda fase consistió en la implementación de nuevas funciones de seguridad IPv6. Los beneficios de la implementación incluyeron:

- **Mejor rendimiento:** La implementación de IPv6 redujo la latencia y el tiempo de respuesta de las llamadas telefónicas.
- **Mayor seguridad:** La implementación de IPv6 mejoró la seguridad de los sistemas de telefonía IP al proporcionar una mayor protección contra ataques.
- **Simplificación de la administración:** La implementación de IPv6 simplificó la administración de los sistemas de telefonía IP al eliminar la necesidad de NAT.

La implementación de Asterisk sobre IPv6 en la UC Berkeley ha tenido un impacto positivo en la conectividad y la seguridad de los sistemas de telefonía IP de la universidad. Las

llamadas telefónicas son ahora más confiables y seguras, y la administración de los sistemas de telefonía IP es más sencilla.

Aquí hay algunos ejemplos específicos de cómo la implementación de Asterisk sobre IPv6 ha beneficiado a la UC Berkeley:

- **Los estudiantes y los profesores pueden ahora comunicarse entre sí de forma más fluida y eficaz.** Las llamadas telefónicas son ahora más confiables y seguras, lo que permite a los estudiantes y los profesores colaborar de forma más fluida.
- **La UC Berkeley puede ahora reducir sus costos de red.** La implementación de IPv6 ha reducido la necesidad de NAT, lo que ha reducido los costos de enrutamiento y administración de la red.

Segundo caso de estudio: Monitoreo Vehicular

En el trabajo realizado de Monitoreo Vehicular, se planteó el diseño e implementación de una aplicación que pueda acceder a una base de datos que posee las calles de la ciudad de Quito, vinculadas a las coordenadas de monitoreo GPS, del cual se pueden recibir y generar llamadas para otorgar información de la posición actual del vehículo con una base de alarmas. Dentro de este trabajo se analizaron los códecs de telefonía IP que relacionan a Asterisk, así como las tecnologías de TTS. Se implementó la comunicación hacia la base de datos María DB mediante el uso de AGI PHP.

En esto se consiguió ejecutar con éxito el plan de marcado diseñado con los respectivos scripts de PHP para la lectura de la base de datos y así se obtuvo el sistema final con un costo de implementación representativo, sin embargo, se analizó que es representativo tomando en cuenta los valores que poseen los softwares especializados (Lema & Ruiz, 2014).

Tercer caso de estudio: Consulta remota a base de datos

En el estudio para la consulta remota a base de datos, se implementó un sistema de consulta remota para el control y gestión de almacenes de la empresa Innova Networks, que, por medio de Asterisk generaron un sistema que permita la consulta de información de los productos registrados en su web. Por medio del IVR, el usuario puede ejecutar las consultas por medio de scripts programados en PHP, que por medio de AGI permite ejecutar estas instrucciones dentro del Dialplan diseñado.

En los resultados se obtuvo una consulta correcta de los productos, mostrando la información solicitada por el usuario, las alertas de desabastecimiento oportunamente y permitiendo a los encargados la toma de decisiones adecuada para mejorar el control de los productos (Rojas, 2013).

Los antecedentes descritos proporcionaron una sólida base para el desarrollo de la investigación, al resaltar la relevancia del "Prototipo de un Sistema en GNS3 con la Integración de Asterisk y PostgreSQL sobre IPv6 para Consulta de Notas Académicas". Además, los estudios de caso y proyectos ofrecen información valiosa sobre los posibles beneficios de la integración, así como los desafíos que pueden surgir. Por lo tanto, es crucial tener una comprensión detallada de Asterisk y su adaptabilidad a IPv6 para garantizar el

desarrollo efectivo del prototipo en GNS3. Esto asegurará una implementación adecuada en el contexto de la consulta de notas académicas sobre IPv6.

1.1.5. Desafíos específicos

- Implementar un prototipo mediante el uso de un simulador que permita de manera controlada evaluar el rendimiento de la integración de las plataformas de voz y base de datos.
- Desplegar sobre el prototipo dos alternativas de integración de Asterisk con PostgreSQL con el protocolo ipv6
- Desplegar herramientas de esfuerzo para medir y evaluar el rendimiento de la integración de las plataformas.

1.1.6. Impacto potencial

La investigación permitió detalladamente guiar la integración PBX Asterisk con Gestores de Base de datos no solo de ámbito académico sino de cualquier otro ámbito y fundamentalmente elegir cual alternativa de integración implementar en base a los resultados y conclusiones que presenta la investigación, previo a la puesta en producción.

1.2. Planteamiento del Problema

1.2.1. Problema y Justificación

En la actualidad, la integración de servicios y plataformas bajo el protocolo IPv6 representa un desafío fundamental para el desarrollo de soluciones tecnológicas innovadoras y eficaces. Este desafío se manifiesta de manera destacada en el ámbito educativo, donde la transición hacia IPv6 se presenta como una necesidad apremiante. En este contexto, se planteó la integración específica de Asterisk y PostgreSQL sobre IPv6 como una solución estratégica para la consulta de notas académicas, una tarea central en las operaciones diarias de las instituciones educativas (Fernández, 2023).

El proyecto no se centró exclusivamente en la consulta de notas académicas, sino que su objetivo principal radica en evaluar el rendimiento de Asterisk integrado con PostgreSQL mediante IPv6. Esto implica un enfoque integral que va más allá de la importancia exclusiva de la base de datos académica. El énfasis de la investigación por lo tanto se centró en medir el rendimiento de la integración utilizando servicios web, así como sin servicios web, reconociendo la importancia de evaluar, cómo las diferentes configuraciones, afectan la eficiencia del sistema (Momtaz & Noor, 2015).

La justificación para desarrollar un prototipo en GNS3 se refuerza con la consideración de medir la concurrencia de solicitudes al servicio. Este enfoque práctico se convierte en un componente esencial del proyecto, ya que permite evaluar cómo el sistema responde en situaciones de carga intensiva en un escenario realista en entornos educativos donde las consultas de notas académicas pueden ocurrir simultáneamente.

Además, la evaluación del rendimiento no solo se limita a la eficiencia interna del sistema, sino que también considera cómo la integración de Asterisk y PostgreSQL sobre IPv6 puede influir con el uso de servicios web. La medición de indicadores como la latencia, el tiempo

de respuesta y la capacidad de carga se realiza con el propósito de comprender a fondo el impacto de la implementación en la experiencia del usuario final y en la eficacia global del sistema (Singh, 2023).

La latencia, representa el tiempo que tarda la información en viajar desde la fuente hasta el destino y se evalúa para garantizar tiempos de respuesta aceptables. El tiempo de respuesta, mide el intervalo entre la emisión de una solicitud y la recepción de la respuesta, se monitorea para asegurar una interacción fluida entre Asterisk, PostgreSQL y el servicio web (formaciongcc.com, 2020). Adicionalmente, la capacidad de carga del sistema es un parámetro crucial para evaluar su desempeño bajo diferentes niveles de demanda. Esto permite identificar posibles cuellos de botella y asegurar que el sistema sea capaz de manejar eficientemente un volumen significativo de consultas académicas. Por ende, estos indicadores combinados proporcionan una visión completa de la eficiencia y efectividad de la integración, contribuyendo así a una evaluación más exhaustiva de los escenarios propuestos (Madsen, 2018).

A su vez con la integración de aplicaciones para el uso de diversas tecnologías que trabajan en diversas plataformas genera un incentivo para el desarrollo e investigación relacionado con la integración con Bases de Datos, Central Telefónica y WebServices. Esto genera un nivel de importancia relevante para plantear sistemas más complejos basados en IPV6 e incluso logrando integrar tecnologías emergentes (Arranz, 2016).

Para lograr una evaluación más completa, se plantea la implementación de dos escenarios distintos. El primero involucra la integración de Asterisk con servicios web, mientras que el segundo conecta directamente con PostgreSQL. La elección de herramientas para medir el rendimiento es crucial en este contexto, y se propone utilizar SIPp, una herramienta similar a JMeter, pero específicamente diseñada para evaluar el rendimiento en llamadas. Esta estrategia permite no solo analizar el rendimiento general del sistema sino también identificar cómo diferentes configuraciones y conexiones impactan la eficiencia operativa y la experiencia del usuario en ambos escenarios (Madsen, 2018).

El proyecto de investigación busca proporcionar una solución integral que aproveche las capacidades de IPv6, optimice la integración de Asterisk con PostgreSQL y garantice un rendimiento, incluso en condiciones de carga concurrente alta. La implementación de dos escenarios distintos, uno con la integración de Asterisk y servicios web; y otro conectado directamente a PostgreSQL, respalda esta búsqueda exhaustiva de soluciones. El prototipo en GNS3 se presenta como una herramienta esencial para probar y validar las soluciones propuestas antes de su implementación en un entorno educativo en producción.

1.2.2. Formulación del Problema

¿Cuál será la importancia de evaluar el rendimiento en la integración de Asterisk y PostgreSQL con IPv6 sobre un prototipo en GNS3?

1.3. Objetivos

1.3.1. Objetivo General

Implementar un Prototipo de un Sistema en GNS3 con la Integración de Asterisk y PostgreSQL sobre IPv6 para Consulta de Notas Académicas.

1.3.2. Objetivos Específicos

- Investigar las tecnologías que permitan la integración de Asterisk y PostgreSQL a través del IPv6.
- Desarrollar el prototipo para la integración de Asterisk y PostgreSQL con IPv6 sobre GNS3.
- Evaluar el rendimiento de las tecnologías utilizadas para la integración de Asterisk y PostgreSQL sobre IPv6.

CAPÍTULO II. MARCO TEÓRICO

2.1. Prototipo de sistema

Un prototipo es una implementación parcial y tangible de un sistema o una parte del mismo, diseñado para explorar diversas cuestiones durante el desarrollo del sistema. Se destaca como una herramienta esencial para la comunicación, participación, exploración, evaluación y documentación en el proceso de desarrollo. Sirve como el primer paso crucial para convertir ideas abstractas en elementos concretos y testeables, permitiendo a diseñadores y usuarios involucrarse activamente en el proceso de desarrollo y mejorar la calidad y la exhaustividad de las especificaciones funcionales. Asimismo, facilita la exploración de nuevas ideas y conceptos tecnológicos, ayudando a los diseñadores a tomar decisiones informadas al seleccionar entre diversas alternativas de diseño (Universidad de Lleida, 2024).

2.1.1. Características de un prototipo de sistema

Los prototipos poseen diversas características fundamentales que guían su desarrollo y utilización en el proceso de diseño de sistemas. Entre las más destacadas se pueden mencionar las siguientes:

- **Propósito:** Se refiere a la razón detrás de la creación de un prototipo, que puede ser para explorar, comunicar, desarrollar, mejorar o validar el producto en desarrollo (Bjarnason, Lang, & Mjöberg, 2023).
- **Alcance:** Indica la extensión y profundidad de las funcionalidades, aspectos visuales, interactivos y de datos que son representados por el prototipo.
- **Medio:** Describe el método o herramienta utilizada para crear el prototipo, como papel, software o hardware, definiendo así la forma en que se materializa la representación del sistema.
- **Uso:** Se refiere al momento y contexto en el que el prototipo es presentado o probado con los usuarios o clientes, determinando así la oportunidad y el entorno para su evaluación.
- **Estrategia de exploración:** Indica el grado de variación y paralelismo de los prototipos, así como el nivel de participación de los usuarios o clientes en el proceso de prototipado, lo que influye en la profundidad de la exploración y la retroalimentación obtenida durante el desarrollo del sistema.

2.2. GNS3

En la figura 1, se observa el entorno de trabajo de GNS3 en el cual se encuentra un diagrama de red con diferentes componentes que tiene el simulador.

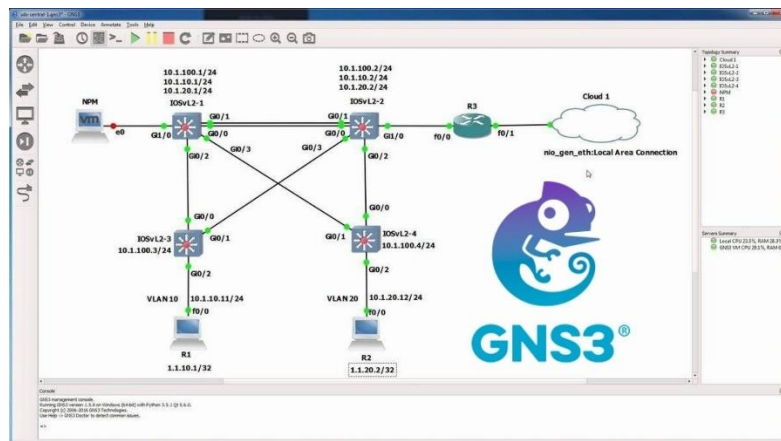


Figura 1. Entorno de trabajo de GNS3 con un diagrama de red

Fuente: (GNS3, 2024)

Esta es una herramienta esencial en el ámbito de las redes de computadoras, reconocida por su capacidad como simulador de redes completo y funcional. Desarrollado como software libre y basado en Python, GNS3 ha sido construido y mejorado por una comunidad dedicada de desarrolladores. Su versatilidad le permite simular una amplia gama de topologías de red, desde las más simples hasta las más complejas, y su compatibilidad con todos los sistemas operativos lo hace accesible para una amplia variedad de usuarios, incluyendo ingenieros, investigadores, estudiantes y profesionales de la informática (González, 2022).

Ventajas de GNS3

Esta ofrece una serie de ventajas significativas que lo hacen destacar como una herramienta de simulación de redes altamente valorada:

- **Software libre y de código abierto:** GNS3 es gratuito y su código fuente está disponible para su modificación y mejora por parte de la comunidad de usuarios (Bustos, 2023).
- **Sin tarifas de licencia:** No hay costos asociados con el uso de GNS3, ya que no requiere tarifas de licencia mensuales o anuales.
- **Escalabilidad ilimitada:** No hay límite en la cantidad de dispositivos compatibles que pueden ser simulados en GNS3, aunque su rendimiento está limitado por el hardware disponible (CPU y memoria).
- **Compatibilidad con múltiples opciones de conmutación:** GNS3 soporta una variedad de opciones de conmutación, incluyendo el módulo Etherswitch NMESW16 y diversas imágenes de IOU/IOL Layer 2.
- **Soporte para todas las imágenes VIRL:** GNS3 es compatible con todas las imágenes VIRL, lo que incluye una amplia gama de dispositivos de Cisco y otros proveedores.
- **Flexibilidad en la ejecución:** Puede funcionar con o sin hipervisores, y es compatible con una variedad de hipervisores gratuitos y de pago, como VirtualBox, VMware Workstation, VMware Player, ESXi y Fusion.

- **Dispositivos preconfigurados disponibles:** GNS3 ofrece dispositivos descargables, gratuitos, preconfigurados y optimizados para simplificar la implementación de redes simuladas.
- **Soporte nativo para Linux:** GNS3 puede ejecutarse en sistemas Linux sin necesidad de software de virtualización adicional.
- **Acceso a software de múltiples proveedores:** Además de dispositivos de Cisco, GNS3 ofrece acceso gratuito a software de otros proveedores.
- **Gran comunidad de usuarios:** GNS3 cuenta con una comunidad grande y activa, con más de 800,000 miembros, lo que facilita el intercambio de conocimientos y la resolución de problemas.

Prototipo con GNS3

Se destaca como un simulador de red que permite a los usuarios emular topologías de red medianamente complejas y simular la gestión de la red. A diferencia de Cisco Packet Tracer, este software ofrece la capacidad única de emular tarjetas de red reales, lo que permite el uso de puertos ethernet tanto física como lógicamente (Bustos, 2023). Asimismo, para facilitar la realización de simulaciones completas, GNS3 incluye una variedad de componentes que se presentan:

- **Dynamips:** Este componente actúa como un emulador de IOS, permitiendo a los usuarios ejecutar binarios de imágenes IOS de Cisco Systems, lo que facilita la simulación de dispositivos Cisco en entornos virtuales (AJPD soft, 2020).
- **Dynagen:** Este front-end basado en texto para Dynamips, Qemu y VirtualBox ofrece una interfaz de usuario que permite a los usuarios gestionar y controlar la simulación de redes de manera eficiente. Además, Dynagen permite la integración de máquinas virtuales, como el firewall PIX, en las topologías de red simuladas.
- **VPCS:** Se trata de un emulador de PC que ofrece funciones básicas de networking, como ping y traceroute, lo que permite a los usuarios realizar pruebas y diagnósticos de conectividad en las simulaciones de red.
- **IOU (IOS on Unix):** Esta característica proporciona compilaciones especiales de IOS proporcionadas por Cisco para ejecutarse directamente en sistemas UNIX y derivados, lo que amplía las capacidades de emulación de dispositivos Cisco en entornos virtuales.

2.3. Asterisk

En la figura 2, se observa el software Asterisk con los diferentes tipos de dispositivos en los que se puede integrar.



Figura 2. Asterisk y sus diferentes integraciones

Fuente: (Ruiz, 2018)

Asterisk es un software de código abierto proporcionado por Digium bajo la licencia GPL. Este software, una vez instalado en un ordenador de propósito general, ofrece todas las funcionalidades típicas de las centralitas telefónicas convencionales, como Cisco, Alcatel o Siemens. Aunque Asterisk es ampliamente utilizado en entornos empresariales, su flexibilidad también lo hace adecuado para su implementación en hogares, siendo este sector el de mayor interés para el proyecto en cuestión. Las características básicas que ofrece Asterisk incluyen desvíos, capturas y transferencias de llamadas, identificación del número llamante, música en espera, autenticación, conexión con líneas de telefonía tradicional y configuración de bases de datos, entre otras (Castro, 2022).

Funcionalidades principales de Asterisk

Asterisk cuenta con funcionalidades más avanzadas, como buzones de voz, IVR, Call Detail Record (CDR), Automatic Call Distribution (ACD), sistema de audioconferencias e integraciones con reconocimiento de voz, entre otras. Los administradores del sistema tienen la capacidad de añadir nuevas funcionalidades de diversas formas, ya sea programando scripts en el lenguaje propio de Asterisk, agregando módulos personalizados escritos en C o desarrollando scripts Asterisk Gateway Interface (AGI) en Perl, Bash, PHP u otros lenguajes de su preferencia. Esta versatilidad ha convertido a Asterisk en la principal alternativa dentro de las plataformas de telefonía de código abierto en la actualidad. Aunque originalmente fue diseñado para el sistema operativo GNU/Linux, actualmente cuenta con versiones para otros sistemas operativos (Castro, 2022).

2.4. PostgreSQL

En la figura 3, se observa el sistema gestor de base de datos PostgreSQL y sus características.



Figura 3. Características de PostgreSQL.

Fuente: (PostgreSQL, 2024)

Este es un sistema de gestión de bases de datos de código abierto que destaca por su potencia y versatilidad. Utiliza una extensión del lenguaje SQL y ofrece una amplia gama de características para almacenar y gestionar eficientemente datos de diversa complejidad. Asimismo, dispone de más de 35 años de desarrollo activo y una sólida reputación en la industria, PostgreSQL se ha convertido en la elección preferida para muchas organizaciones y profesionales en busca de una solución fiable y flexible para sus necesidades de almacenamiento de datos. Además de su arquitectura probada, confiabilidad y extensibilidad, este se distingue por su compatibilidad con todos los principales sistemas operativos, su cumplimiento con ACID desde hace más de dos décadas y su capacidad para integrar poderosos complementos como PostGIS para la gestión de datos geoespaciales (PostgreSQL, 2024).

Características principales de PostgreSQL

Este ofrece una serie de características distintivas que lo hacen destacar como un sistema de gestión de bases de datos robusto y versátil. Algunas de sus características principales incluyen:

- **Arquitectura Orientada a Objetos:** PostgreSQL permite el modelado de datos orientado a objetos, lo que habilita la definición de tipos de datos personalizados, funciones y operadores, ofreciendo flexibilidad en el diseño de la base de datos (IONOS, 2022).
- **Transacciones ACID:** Garantiza la integridad y consistencia de los datos mediante transacciones Atomicidad, Consistencia, Aislamiento y Durabilidad (ACID), lo que asegura que las operaciones se realicen de manera segura y confiable.
- **Extensibilidad:** Los usuarios pueden definir sus propias funciones y tipos de datos, lo que permite adaptar PostgreSQL a requisitos específicos y ampliar su funcionalidad según las necesidades del proyecto.
- **Soporte para Consultas Complejas:** PostgreSQL cuenta con un optimizador de consultas potente que puede manejar eficientemente consultas complejas y grandes

conjuntos de datos, garantizando un rendimiento óptimo incluso en escenarios exigentes.

- **Replicación y Alta Disponibilidad:** PostgreSQL ofrece opciones integradas para la replicación de datos y configuraciones de alta disponibilidad, lo que garantiza la continuidad del servicio y la disponibilidad de los datos en caso de fallos del sistema.

2.5. IPV6

Protocolo de Internet versión 6, se presenta como el sucesor del ampliamente utilizado protocolo IPv4. Su principal función es permitir la conexión de dispositivos fijos o móviles a Internet mediante el uso de direcciones IP de 128 bits. Estas direcciones únicas facilitan la identificación inequívoca de cada dispositivo en la red, lo que garantiza una comunicación sin contratiempos entre ellos (Aguirre, 2023).

Este nuevo estándar de red ofrece mejoras significativas en el rendimiento y la seguridad, gracias a su espacio de direcciones de 128 bits, que prácticamente garantiza una cantidad ilimitada de direcciones únicas. Además, IPv6 simplifica el enrutamiento y reduce la necesidad de técnicas de traducción de direcciones de red Network Address Translation (NAT), lo que contribuye a una comunicación más fluida entre dispositivos y mejora la eficiencia operativa de las redes. En un contexto de crecimiento exponencial de dispositivos IoT y la necesidad de direcciones únicas para cada uno, la transición hacia IPv6 se vuelve inevitable. Por lo tanto, su implementación efectiva es esencial para garantizar la continuidad y el crecimiento de las redes globales, adaptándose a las demandas y desafíos de un mundo digitalmente interconectado (Khasawneh & Alarmouty, 2019).

Características de IPv6

Esta presenta una serie de características fundamentales que lo distinguen y lo hacen esencial en el panorama de las redes de próxima generación:

- **Jerarquía eficiente en enrutamiento:** La implementación de múltiples niveles de jerarquía facilita la consolidación de rutas, promoviendo un enrutamiento eficiente y escalable en Internet (NIC México, 2024).
- **Amplio espacio de direcciones:** Con un esquema de direcciones de 128 bits, IPv6 ofrece una abundancia de direcciones IP, permitiendo asignar identificadores únicos globales a cada nuevo dispositivo conectado a la red.
- **Autoconfiguración simplificada:** Los nodos de la red IPv6 pueden configurar sus propias direcciones de manera autónoma, agilizando el proceso y facilitando su implementación.
- **Transición transparente entre proveedores:** El mecanismo de reenumerado garantiza una transición sin inconvenientes entre proveedores de IPv6, manteniendo la transparencia para los usuarios finales.
- **Uso de multicast en el link local:** IPv6 reemplaza la difusión ARP con el uso de multicast en el ámbito local de la red, mejorando la eficiencia en la comunicación.
- **Encabezado más eficiente:** El encabezado de IPv6 es más eficiente en comparación con IPv4, con menos campos y la eliminación de la suma de verificación del encabezado.

- **Diferenciación de tráfico:** IPv6 permite la diferenciación de tráfico mediante los campos del encabezado, brindando mayor control y flexibilidad en la gestión del tráfico de red.
- **Extensiones de encabezado innovadoras:** Las nuevas extensiones de encabezado reemplazan el campo de Opciones de IPv4, ofreciendo mayor flexibilidad y capacidades avanzadas.
- **Manejo eficiente de movilidad y seguridad:** IPv6 ha sido diseñado para gestionar de manera eficiente mecanismos de movilidad y seguridad en comparación con su predecesor, IPv4.
- **Múltiples mecanismos de transición:** Se han creado varios mecanismos junto con el protocolo IPv6 para asegurar una transición sin problemas de las redes IPv4 a IPv6, asegurando la continuidad y evolución de las infraestructuras de red.

2.6. Integración de plataformas

Se define como el proceso mediante el cual se permite que aplicaciones individuales, diseñadas con propósitos específicos, trabajen en conjunto de manera eficiente. Este proceso implica la fusión y optimización de datos y flujos de trabajo entre múltiples aplicaciones de software, lo que permite a las organizaciones modernizar sus infraestructuras y respaldar operaciones comerciales ágiles. Por lo tanto, la integración de plataformas es crucial para cerrar la brecha entre los sistemas locales existentes y las aplicaciones empresariales basadas en la nube de rápida evolución. Al interconectar procesos e intercambiar datos de manera fluida, esta integración permite a las empresas organizar diversas funciones en toda su infraestructura, lo que a su vez les permite operar de manera más efectiva y eficiente (IBM, 2023).

Características de la integración de plataformas

Algunas de las características fundamentales de la integración de plataformas son las siguientes:

- **Interoperabilidad:** Se refiere a la capacidad de las plataformas para comunicarse y compartir información entre sí, utilizando estándares comunes y protocolos establecidos. Esto garantiza una integración fluida y eficiente entre diferentes sistemas y aplicaciones (IBM, 2023).
- **Escalabilidad:** Esta característica se relaciona con la capacidad de las plataformas para adaptarse al crecimiento o disminución de la demanda sin que esto afecte su rendimiento o calidad del servicio. Las plataformas deben ser capaces de escalar vertical u horizontalmente según sea necesario para satisfacer las necesidades cambiantes del negocio.
- **Seguridad:** Se refiere a la capacidad de las plataformas para proteger la confidencialidad, integridad y disponibilidad de los datos y servicios frente a amenazas tanto internas como externas. Esto incluye medidas de seguridad como el cifrado de datos, la autenticación de usuarios y la gestión de accesos para prevenir cualquier vulnerabilidad.

- **Flexibilidad:** Esta característica hace referencia a la capacidad de las plataformas para incorporar nuevas funcionalidades, tecnologías o proveedores sin comprometer la compatibilidad o eficiencia del sistema. Por ello, las plataformas deben ser lo suficientemente flexibles para adaptarse a los cambios en el entorno empresarial y aprovechar las oportunidades emergentes sin interrupciones en las operaciones existentes.

2.7. WebServices

Es un sistema de información diseñado para facilitar la interoperabilidad entre diferentes máquinas a través de una red. En el ámbito de los sistemas de información, se emplea con el fin de garantizar la accesibilidad y aplicabilidad de los sistemas en diversas plataformas, como sitios web, aplicaciones de escritorio y móviles, entre otros. Este tipo de servicio utiliza métodos de Application Programming Interface (API) para procesar y presentar información de manera remota, posibilitando el acceso a los datos a través de múltiples plataformas (Kurniawan & Warlina, 2020). Además, contribuye al desarrollo y gestión eficiente de datos al ser altamente adaptable y escalable gracias al uso de API.

En la Figura 4 se observa el proceso de utilización de un WebService,

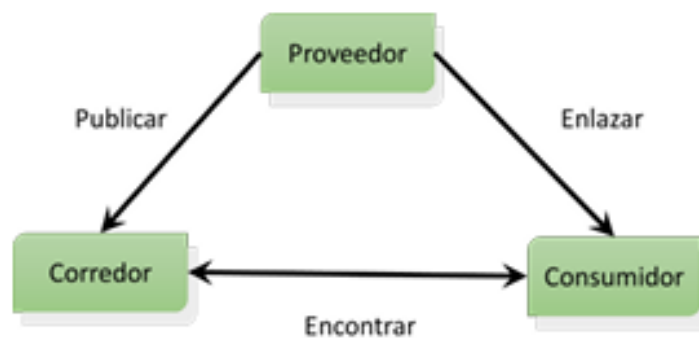


Figura 4. Procedimiento de comunicación de un WebService

Fuente: (Kurniawan & Warlina, 2020)

Características de un WebService

Entre las características de un WebService se pueden mencionar las siguientes:

- **Interoperabilidad:** Facilita la comunicación entre diversas plataformas y lenguajes de programación, permitiendo una integración efectiva de sistemas heterogéneos (Sivankalai & Virumandi, 2021).
- **Accesibilidad:** Puede ser utilizado desde cualquier dispositivo con conexión a Internet, lo que proporciona una amplia disponibilidad y accesibilidad a los servicios ofrecidos.
- **Escalabilidad:** Es capaz de manejar un gran número de solicitudes simultáneas, lo que garantiza un rendimiento óptimo incluso en entornos de alta demanda.

- **Capacidad de Reutilización:** Puede ser utilizado por múltiples aplicaciones y plataformas, lo que fomenta la reutilización de recursos y la eficiencia en el desarrollo de software.
- **Formato basado en texto:** Utilizan un formato basado en texto para la transmisión de datos, lo que facilita la interoperabilidad y el procesamiento de la información.
- **Fácil uso y acceso:** Son herramientas accesibles y fáciles de utilizar, lo que simplifica su implementación y adopción por parte de los desarrolladores y usuarios.
- **Intercambio de mensajes SOAP:** Permiten el intercambio de mensajes utilizando el protocolo Simple Object Access Protocol (SOAP), lo que garantiza la comunicación segura y confiable entre los sistemas.
- **Interfaz descrita en WSDL:** La interfaz del servicio web se describe utilizando WebService Description Language (WSDL), lo que proporciona una descripción detallada de sus operaciones y parámetros.
- **Se apoya en el formato HTTP:** Utilizan el protocolo Hypertext Transfer Protocol (HTTP) para la comunicación, lo que garantiza una comunicación eficiente y compatible con la infraestructura de Internet.

2.8. AGI (Asterisk Gateway Interface)

Es una Interfaz de Programación de Aplicaciones (API) diseñada para permitir que un programa externo controle la lógica de llamadas en Asterisk, el software de centralita telefónica de código abierto. Por ello, esta interfaz actúa como un puente entre Asterisk y aplicaciones externas, posibilitando la ejecución de comandos complejos y la integración con otros sistemas y tecnologías, destacando que AGI se utiliza especialmente para implementar Buzones de Voz Interactivos (IVR) en una red IMS, con el propósito de proporcionar servicios de comunicación mejorados para redes 4G o incluso 5G (Ndiaye et al., 2020).

Características de AGI

Entre las características fundamentales de AGI se pueden mencionar las siguientes:

- **Interacción con el Plan de Marcación:** AGI permite que un programa externo interactúe con el plan de marcación de Asterisk, posibilitando acciones como registrar información, validar datos o interactuar con servicios web (Deka & Kumari, 2019).
- **Sincronización:** La ejecución de AGI es sincrónica, lo que implica que no devuelve el control al plan de marcación hasta que la acción se completa, asegurando un flujo de ejecución ordenado y predecible.
- **Manipulación de Variables:** AGI puede leer y escribir variables de canal, facilitando la transferencia de información entre Asterisk y el programa externo, lo que aumenta la flexibilidad y la capacidad de personalización de las aplicaciones.
- **Acceso a Bases de Datos:** AGI proporciona la capacidad de registrar o validar información en bases de datos locales o externas, lo que amplía su utilidad y la integración con otros sistemas.

- **Integración con Servicios Web:** AGI puede interactuar con servicios web o APIs REST para obtener o procesar datos, lo que abre la puerta a la integración con una amplia gama de servicios y plataformas en línea.
- **Librerías y Frameworks:** Existen diversas librerías y frameworks disponibles en varios lenguajes de programación, como PHP, Python, Java, entre otros, que permiten implementar AGI de manera efectiva y eficiente, brindando a los desarrolladores opciones flexibles para trabajar con esta interfaz.

2.9. ODBC (Open Database Connectivity)

Es una tecnología de interfaz de base de datos desarrollada por Microsoft en sus primeros años. Su propósito es proporcionar a los programadores un método sencillo para acceder al contenido de las bases de datos de manera independiente del lenguaje de programación utilizado. Por ello, ODBC emplea un par de controladores: uno para el gestor de base de datos y otro para la interfaz común del lenguaje de programación, permitiendo así el acceso al contenido de la base de datos a través de una interfaz estandarizada mediante llamadas a funciones comunes. Esta tecnología se basa en una versión estandarizada del lenguaje de consulta estructurada (SQL), lo que posibilita la escritura de código de acceso a bases de datos que sea independiente de cualquier sistema de gestión de bases de datos específico, lo que resulta en programas modularizados.

Además, ODBC se presenta como una herramienta clave para el manejo del Big Data, ya que fue creada con el objetivo principal de facilitar el acceso a los datos como una API, siendo indiferente al sistema operativo y al lenguaje sobre el que se ejecuta. Esta característica de independencia otorga versatilidad y flexibilidad a las aplicaciones que utilizan ODBC, permitiendo su implementación en una amplia variedad de entornos tecnológicos (Chen, 2020).

Características de ODBC

Este presenta una serie de características distintivas que lo hacen una herramienta poderosa para el manejo de bases de datos. Entre estas características se pueden mencionar:

- **Estándar Abierto:** A diferencia de Java Database Connectivity (JDBC), ODBC se fundamenta en un estándar abierto, permitiendo a los usuarios acceder y modificarlo según sus necesidades (Chen, 2020).
- **Controladores y Administrador de Controladores:** ODBC incluye estas herramientas que optimizan y simplifican el procesamiento de datos.
- **Compatibilidad con Diversos Lenguajes de Programación:** ODBC no se restringe únicamente al lenguaje de programación Java; también es compatible con otros lenguajes.
- **Portabilidad de Aplicaciones:** Las aplicaciones desarrolladas con ODBC pueden ser trasladadas a otras plataformas sin dificultades.
- **Evolución Continua:** Desde su creación en 1992, ODBC ha seguido evolucionando con actualizaciones y mejoras constantes.

2.10. Rendimiento del sistema

El rendimiento del sistema se refiere a la capacidad de un sistema para mantener su funcionamiento dentro de los parámetros esperados. Esto implica la habilidad del sistema para responder efectivamente a perturbaciones externas o fluctuaciones internas, manteniendo así su viabilidad y cumpliendo con las expectativas de desempeño. El rendimiento está directamente relacionado con la capacidad regulatoria del sistema, que es una función de su diseño, ejecución y desarrollo continuo (Keating et al., 2019).

2.11. SIPp

En la figura 5, se puede visualizar el cómo es el comportamiento de SIPp al momento de generar un tráfico de llamadas.

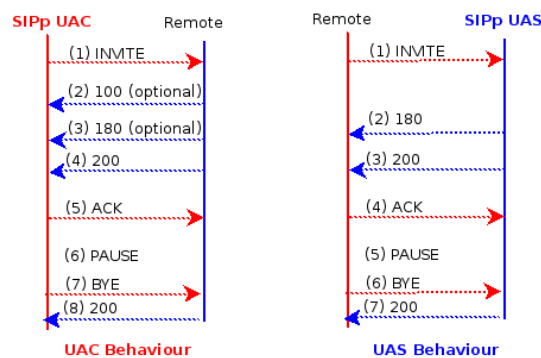


Figura 5. Tráfico de llamadas entre UAC y UAS de SIPp

Fuente: (SIPp, 2014)

Es una herramienta de prueba de rendimiento para el protocolo SIP. Incluye algunos escenarios básicos de agentes de usuario de SipStone (UAC y UAS) que establece y libera múltiples llamadas con los métodos INVITE y BYE. También, puede leer archivos de escenarios XML que describen cualquier configuración de prueba de rendimiento (SIPp, 2014).

Características de SIPp

Este presenta varias características clave que lo distinguen como una herramienta eficaz para la priorización y gestión de casos en la recolección de datos:

- **Priorización de Casos:** Se refiere al esfuerzo concertado para alcanzar una alta calidad de datos durante la recolección, reasignando recursos escasos a los casos más necesitados (Tolliver et al., 2023).
- **Diseño Adaptativo:** Implica la implementación de estrategias modificadas durante la recolección de datos para abordar la disminución en las tasas de respuesta y mejorar las medidas de calidad de datos.

2.12. Softphone

Este se define como una aplicación de software que permite realizar llamadas de voz a través de Internet mediante la tecnología de Voz sobre IP (VoIP). En el ámbito empresarial, los Softphones forman parte de las soluciones IP-PBX, que han evolucionado para aprovechar

las redes de datos en lugar de depender de infraestructuras tradicionales como Private Branch Exchange (PBX). Estos Softphones se integran en el ecosistema corporativo para mejorar la eficiencia de las comunicaciones empresariales, aprovechando la ubicuidad de la red de datos en las organizaciones (Tanutama et al., 2021).

Características de Softphone

Las características del Softphone incluyen una funcionalidad completa que equipara su desempeño con el de un teléfono fijo de escritorio. Esto implica la capacidad de realizar y recibir llamadas, gestionar contactos, enviar mensajes de texto y correo de voz, entre otras funcionalidades básicas de comunicación.

Uno de los aspectos destacados es el ahorro de costos asociado con los Softphones, ya que muchos de ellos son gratuitos, eliminando así la necesidad de incurrir en gastos iniciales en la compra de teléfonos IP dedicados. Esta característica resulta especialmente atractiva para las empresas que buscan reducir sus gastos de infraestructura de comunicación.

Además, los Softphones ofrecen una notable movilidad al permitir a los usuarios hacer y recibir llamadas desde cualquier lugar con acceso a Internet, utilizando dispositivos como teléfonos móviles, tablets, computadoras portátiles y PCs. Esta flexibilidad en la ubicación mejora la productividad y la accesibilidad de las comunicaciones empresariales.

Otra característica relevante es la compatibilidad con diferentes plataformas, lo que significa que existen Softphones disponibles para una amplia variedad de sistemas operativos, incluyendo iOS, Android, Windows Phone, macOS, Windows y Linux. Esta diversidad asegura que los usuarios puedan acceder a las funcionalidades del Softphone independientemente del dispositivo que utilicen.

Finalmente, el Softphone facilita el trabajo remoto al permitir a los empleados utilizarlo desde sus hogares u otras ubicaciones geográficas. Esta capacidad de trabajar de forma remota es cada vez más importante en el entorno laboral actual, donde la flexibilidad y la movilidad son esenciales para mantener la eficiencia y la colaboración entre equipos distribuidos.

Zoiper Softphone VoIP

En la figura 6, se puede visualizar en el entorno del software Softphone Zoiper en el cual se está realizando una llamada y en la barra superior se puede encontrar las diferentes funcionalidades.

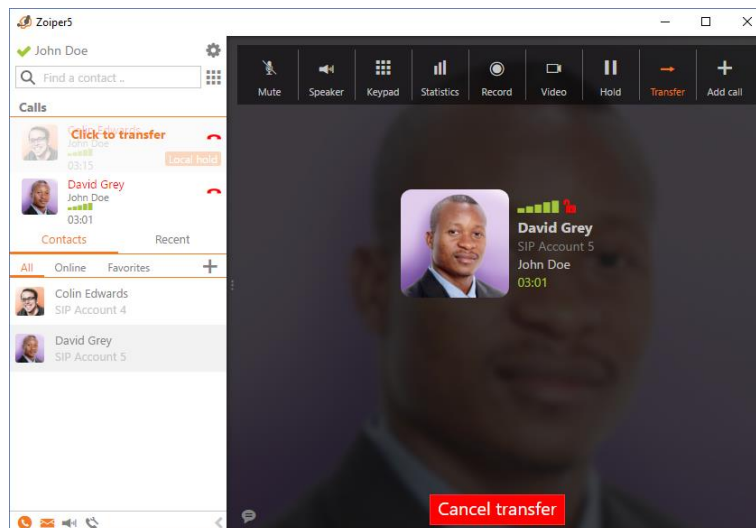


Figura 6. Entorno del software Softphone Zoiper

Fuente: (Nubelfon, 2023)

Es un softphone que permite hacer y recibir llamadas telefónicas a través de internet. Se trata de una aplicación de comunicaciones unificadas que se ha convertido en una alternativa popular para empresas y usuarios individuales que buscan una solución flexible y eficiente para sus necesidades de comunicación (Restoy, 2023).

Funcionalidades

Zoiper es una herramienta de comunicación versátil y eficiente con las siguientes funcionalidades principales:

- **Integración de cuentas SIP:** Permite integrar diferentes cuentas SIP para mayor flexibilidad en la elección de proveedores de telefonía IP.
- **Calidad de llamadas:** Ofrece una excelente calidad de sonido con tecnología de compresión de audio de alta calidad y soporte para el códec de audio G.711.
- **Registro de llamadas:** Ofrece una función de registro detallado de todas las llamadas realizadas y recibidas.
- **Integración con otros servicios:** Permite la integración con servicios como Google Voice, Microsoft Teams y SIP URI para una comunicación más eficiente (Nubelfon, 2023).

2.13. Máquinas virtuales (MV)

Son entidades de software que replican el funcionamiento del hardware de una computadora física, lo que permite la ejecución simultánea de múltiples sistemas operativos en un único dispositivo físico. Estas VM son administradas por un programa conocido como hipervisor, el cual actúa como una capa de aislamiento entre el hardware físico y las máquinas virtuales. Por ello, esta capa de virtualización proporciona seguridad y eficiencia al asignar recursos de hardware a cada máquina virtual de manera controlada y aislada (Aalam et al., 2021).

Características de las máquinas virtuales

Entre las principales características de las Máquinas Virtuales se pueden mencionar:

- Las máquinas virtuales son entidades de software que emulan el hardware de una computadora física, permitiendo que múltiples sistemas operativos se ejecuten simultáneamente en un solo dispositivo físico.
- Las máquinas virtuales son gestionadas por un programa llamado hipervisor, que actúa como una capa de aislamiento entre el hardware físico y las máquinas virtuales, proporcionando seguridad y eficiencia en la asignación de recursos.

2.14. Virtual Box

Es una aplicación de virtualización de código abierto que posibilita la ejecución de múltiples sistemas operativos de manera simultánea en una única máquina física. Por lo tanto, este software resulta especialmente útil para desarrolladores, administradores de sistemas y cualquier persona que requiera probar aplicaciones o sistemas en diversos entornos sin la necesidad de contar con hardware adicional (Khan et al., 2022).

Características de Virtual Box

Este presenta una serie de características destacadas que lo hacen una herramienta versátil y útil en el ámbito de la virtualización:

- **Multiplataforma:** Es compatible con una amplia variedad de sistemas operativos de 32 y 64 bits, incluyendo Windows, GNU/Linux, Mac OS X y Solaris (Khan et al., 2022).
- **Multihuéspedes:** Permite la virtualización de diversos sistemas operativos, abarcando desde Windows 8 y Windows 7 hasta Debian, Ubuntu, OS/2, Mac OS X, DOS, Solaris, entre otros.
- **Portabilidad:** Ofrece una experiencia consistente en todas las plataformas en las que se ejecuta, lo que posibilita la creación de máquinas virtuales en un sistema y su posterior ejecución en otro.
- **Compatibilidad sin virtualización asistida por hardware:** En muchos casos, VirtualBox puede operar sin la necesidad de características específicas de hardware, lo que lo convierte en una opción viable incluso para equipos más antiguos.
- **Importación y exportación de máquinas virtuales:** Utiliza el estándar Open Virtualization Format (OVF) para facilitar el proceso de importación y exportación de máquinas virtuales, permitiendo la interoperabilidad entre diferentes plataformas de virtualización.

CAPÍTULO III. METODOLOGIA

3.1. Tipo de Investigación

La investigación propuso una metodología de investigación cuantitativa para alcanzar los objetivos específicos. En primer lugar, se llevó a cabo una investigación bibliográfica a través de una revisión y análisis de documentación técnica y literaria sobre la integración de Asterisk y PostgreSQL sobre protocolo IPv6. Este enfoque permitió determinar las opciones en la integración de estas tecnologías. Posteriormente, se desarrolló un prototipo simulado de red en el entorno de GNS3, sobre el cual se despliega las máquinas virtuales con los servidores de Asterisk y PostgreSQL sobre IPv6; así como los equipos activos de red como routers y switches para implementar los diferentes escenarios de integración y disponer del sistema funcional para ejecutar las pruebas de rendimiento y recopilar datos cuantitativos sobre la latencia, el tiempo de respuesta y el uso de recursos, comparando así las métricas entre los diferentes escenarios implementados. Finalmente, se analizaron los datos cuantitativos obtenidos mediante el software de esfuerzo SIPp para interpretar los resultados, identificar tendencias y patrones, y generar conclusiones y recomendaciones; así como posibles áreas de investigación futura del presente proyecto.

3.2. Diseño de la Investigación

El diseño de investigación metodológica propuesto para el presente estudio integró observación y experimentación como componentes fundamentales. La observación se llevó a cabo a través de la revisión de la literatura, permitiendo una comprensión profunda de las soluciones para la integración de Asterisk y PostgreSQL sobre IPv6. Esta fase estableció los procedimientos y bases tanto teóricas como prácticas indispensables para el desarrollo del prototipo. Por otro lado, la experimentación se llevó a cabo en el entorno de GNS3, donde se implementaron dos escenarios de prueba, mismos que fueron evaluados utilizando el software de generación de tráfico SIPp para medir indicadores de latencia, uso de recursos y tiempo de respuesta. El primer escenario consistió en una conexión directa entre la base de datos PostgreSQL y Asterisk, mientras que el segundo escenario implicó una conexión mediante un Webservice utilizando AGI. Se recopilaron los datos cuantitativos de rendimiento, lo que permitió analizar comparativamente los resultados y obtener conclusiones significativas.

3.3. Técnicas de recolección de Datos

Se empleó la técnica de recolección de datos cuantitativas, para lo cual se realizó una revisión exhaustiva de la literatura académica y técnica relacionada con la integración de Asterisk y PostgreSQL sobre IPv6, complementada con el análisis de documentación técnica y recursos en línea. La experimentación se llevó a cabo en el entorno de simulación GNS3, donde se realizaron pruebas de rendimiento para evaluar el desempeño de la integración propuesta. Se recopilaron datos cuantitativos sobre latencia, uso de recursos y tiempo de respuesta utilizando herramientas de esfuerzo como SIPp, por lo que las técnicas de recolección de datos fueron en su gran mayoría computacionales.

3.4. Población de estudio y tamaño de muestra

La población objetivo de este estudio es infinita, puesto que se utilizó software de esfuerzo con distintos parámetros de concurrencia sobre los sistemas informáticos que componen el prototipo, para medir su rendimiento en términos de tiempo de respuesta, latencia y uso de recursos, por lo tanto, el concepto de tamaño de muestra tradicional no se aplicó de manera directa. Por esta razón, el enfoque se centró en realizar pruebas exhaustivas y representativas sobre dos escenarios de integración de Asterisk y PostgreSQL sobre IPv6. Se buscó garantizar que las pruebas fueran lo suficientemente completas como para obtener resultados significativos y relevantes que pudieran informar sobre el rendimiento del prototipo implementado.

3.5. Métodos de análisis y procesamiento de datos

En la presente investigación se recurrieron a diversos métodos de análisis y procesamiento de datos adaptados a la naturaleza cuantitativa de la información obtenida, además de realizar un análisis estadístico descriptivo para identificar las relaciones entre variables cuantitativas y determinar la relación con el rendimiento del sistema durante las pruebas de desempeño. El procesamiento de datos incluyó la visualización de resultados mediante tablas para una presentación clara y comprensible de la información obtenida.

En la Figura 7 se observa que para el caso del primer escenario de integración se desplegaron sobre el prototipo tres servidores y una máquina de cliente. Dentro de este escenario se implementó la conexión directa mediante entre Asterisk y PostgreSQL utilizando ODBC y se aplicaron las pruebas respectivas para la comprobación del servicio mediante la herramienta de esfuerzo SIPp.

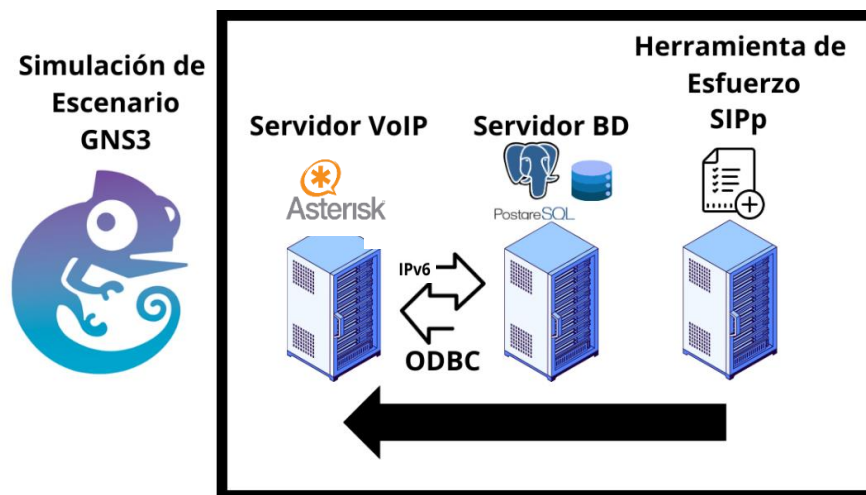


Figura 7. Diagrama de pruebas del escenario 1 de integración utilizando ODBC

En la Figura 8 se observa que, para el segundo escenario de integración, se desplegaron sobre el prototipo tres servidores y una máquina de cliente. Dentro de este escenario se implementó la conexión entre Asterisk y PostgreSQL utilizando Webservice a través de la tecnología AGI y se aplicaron las pruebas respectivas para la comprobación del servicio mediante la herramienta de esfuerzo SIPp.

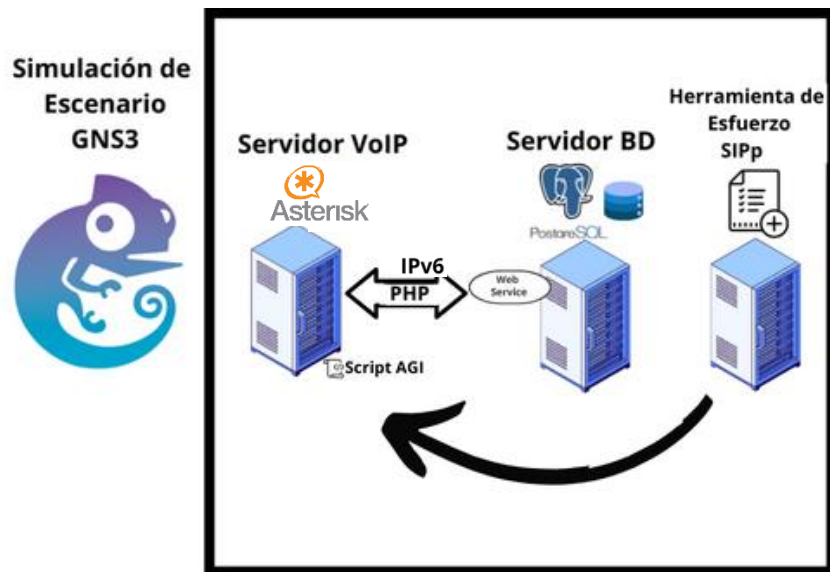


Figura 8. Diagrama de pruebas del escenario 2 de integración utilizando WebService y AGI

3.6. Identificación de variables

Las variables identificadas en el presente trabajo de investigación fueron fundamentales para evaluar el rendimiento sobre la integración de las diversas tecnologías empleadas como Asterisk y PostgreSQL sobre IPv6 en el proceso de consulta académica, utilizando el simulador GNS3 e implementando un prototipo funcional.

3.6.1. Variable Dependiente

Rendimiento del servicio implementado.

3.6.2. Variable Independiente

Prototipo con la Integración de Asterisk y PostgreSQL sobre IPv6.

3.7. Operacionalización de Variables

En la Tabla 1 se describen el problema, objetivos y variables del tema, además de la conceptualización, dimensiones e indicadores para cada variable.

Tabla 1. Operacionalización de las Variables

PROBLEMA	TEMA	OBJETIVOS	VARIABLES	CONCEPTUALIZACION	DIMENSION	INDICADORES
¿Cuál será la importancia de evaluar el rendimiento en la integración de Asterisk y PostgreSQL con IPv6 sobre un prototipo en GNS3?	Prototipo de un Sistema en GNS3 con la Integración de Asterisk y PostgreSQL sobre IPv6 para consulta de Notas Académicas.	General	Independiente	Escenario simulado con equipos activos de red y sistemas de telefonía IP y base de datos integrados utilizando protocolo IPv6 en la capa de red del modelo TCP/IP	Prototipo	<ul style="list-style-type: none"> • PBX Asterisk funcional. • Base de datos académica funcional con PostgreSQL • Asterisk y PostgreSQL integrados sobre ipv6
		<ul style="list-style-type: none"> • Implementar un Prototipo de un Sistema en GNS3 con la Integración de Asterisk y PostgreSQL sobre IPv6 para Consulta de Notas Académicas. 	Prototipo con la Integración de Asterisk y PostgreSQL sobre IPv6.			
		Específicos	Dependiente	Capacidad de respuesta y funcionalidad de las tecnologías aplicadas para integración de Asterisk y PostgreSQL sobre IPv6	Rendimiento	<ul style="list-style-type: none"> • Tiempo de respuesta. • Latencia. • Uso de recursos.
		<ul style="list-style-type: none"> • Investigar las tecnologías que permitan la integración de Asterisk y PostgreSQL a través del IPv6. • Desarrollar el prototipo con la integración de Asterisk y PostgreSQL con IPv6 sobre GNS3. • Evaluar el rendimiento de las tecnologías utilizadas para la integración de Asterisk y PostgreSQL sobre IPv6. 	Rendimiento de las tecnologías utilizadas para la integración			

3.8. Procedimiento de implementación

1. Investigar las tecnologías de integración de Asterisk y PostgreSQL sobre protocolo IPv6 y realizar un análisis comparativo para seleccionar las mejores alternativas de integración a implementarse sobre un escenario simulado utilizando GNS3.
2. Implementación del Prototipo de Integración de Asterisk y PostgreSQL sobre IPv6 utilizando GNS3.

2.1. Instalación y Configuración de GNS3 sobre el Sistema Operativo Windows 10

2.2. Creación de 4 Máquinas Virtuales utilizando VirtualBox versión 7.0, para la instalación de:

- Windows 7 64 bits. - Donde se instala el Softphone Zoiper para realizar las llamadas y consultar las notas del estudiante.
- Linux Sangoma 7.0. – Sistema operativo donde se encuentra pre instalado la PBX Asterisk versión 13.38.3 donde se configura los canales SIP, plan de marcado, IVR y se configura las alternativas de integración con PostgreSQL utilizando ODBC y Webservice con AGI PHP.
- Ubuntu Desktop versión 18.04.- Donde se instala la base de datos PostgreSQL con soporte del protocolo IPv6 y se implementa además la capa de WebService a ser consumida desde AGI de Asterisk.
- Ubuntu Desktop versión 22.04.- Donde se instala la herramienta de esfuerzo SIPp para realizar las pruebas de rendimiento del sistema.

2.3. Diseño de los escenarios de red para las alternativas de integración y pruebas de rendimiento

- Establecimiento del direccionamiento de red
- Diseño Lógico
- Despliegue físico de máquinas virtuales y componentes activos de red para cada escenario de integración.
- Configuración de canales SIP y plan de marcado en la PBX Asterisk
- Configuración de /etc/asterisk/ manager.conf para la detección de Softphone por IPv6 en la línea bindaddr.
- Configuración de la extensión 1888 para registrar el Softphone y probar las dos alternativas de integración con el sistema académico.
- Comprobación de conexión del Softphone con el servidor de Asterisk.
- Configuración del IVR a través del número 2001 para conexión directa a mediante el uso de ODBC con la base de datos de PostgreSQL.
- Configuración del IVR a través del número 2002 para conexión mediante Webservice a la base de datos de PostgreSQL.

2.4. Implementación de Base de Datos mediante PostgreSQL

- Instalación de PostgreSQL 10.23 dentro de la máquina virtual de Ubuntu Desktop versión 18.04,
- Configuración de archivo para escuchar las direcciones IPv6 configuradas dentro del archivo /etc/postgresql/10/main/ pg.hba.conf.
- Configuración de archivo /etc/postgresql/10/main/ postgresql.conf. para la selección del puerto y permisos necesarios para la comunicación.

- Creación de la Base de Datos Sistema_Academico dentro de PostgreSQL.
- Creación de la tabla Notas_Estudiantes dentro de base de datos Sistema_Academico con las columnas ID, nombre, apellido, CI, Nota1, Nota2, Nota3, Nota4, Nota5.
- Creación de la tabla Materias dentro de la tabla Notas_Estudiantes con las columnas ID, código, materia, asignatura.
- Registro de las asignaturas dentro de la tabla Materias.
- Registro de los alumnos y notas dentro de la tabla Notas_Estudiantes.

Implementación de servidor festival TTS

- Instalación de archivos del sistema Festival mediante el comando `sudo yum install festival`.
- Configuración de Puerto e IP dentro del archivo `/etc/Asterisk/festival.conf`, para la ejecución dentro de Asterisk

2.5. Configuración de softphone Zoiper

- Instalación del softphone zoiper dentro de la máquina virtual de Windows 7 64 bits.
- Colocar las SIP CREDENTIALS con la dirección IP del servidor, el usuario o extensión y el password.
- Registrar la extensión y verificar conexión.
- Integración del sistema mediante ODBC (Escenario 1)
- Instalación del socket de conexión `sudo yum install odbc-postgresql`.
- Comprobación del Driver ODBC dentro del archivo `/etc/odbcinst.ini`
- Crear la instancia SISTEMA_ACADEMICO_PSQL para el llamado de las funciones ODBC dentro del archivo `/etc/odbc.ini`.
- Configurar la instancia con usuario: admin, contraseña: 1234, puerto de comunicación: 5432, Dirección IP del servidor: 2003:db8:c::10 y nombre de la base de datos: Sistema_Academico.
- Comprobar la Conexión con la Base de Datos mediante el comando `isql -v Sistema_Academico_PSQL`.
- Configurar el archivo `res_odbc_custom.conf` para habilitar la instancia SISTEMA_ACADEMICO_PSQL.
- Configurar el archivo `func_odbc_conf` con los QUERY necesarios para la búsqueda dentro de PostgreSQL.
- Configurar el archivo `extensions_custom.conf`, para determinar el Dialplan de la extensión 2001 para la implementación del IVR de consulta de notas.
- Realizar pruebas de funcionamiento.

2.6. Integración del sistema mediante WebService AGI PHP (Escenario 2)

- Crear el archivo `prueba.php` para el consumo desde Asterisk incluyendo la librería `phpagi`.
- Configurar los QUERY que permitan la lectura de los datos y de los alumnos en la base de datos.
- Agregar la dirección url `http://[2003:db8:c::10]:80/pruebacon.php` para que exista conexión con la Base de Datos.

- Programar el código para que realice la búsqueda y consulta de la nota tanto por todas las materias como por asignatura.
- Configurar el archivo `extensions_custom.conf`, para determinar el Dialplan de la extensión 2002 para la implementación del IVR de consulta de notas por Webservice.
- Colocar dentro del dialplan la dirección `/var/lib/Asterisk/agi-bin/prueba.php`, para la lectura mediante AGI.
- Realizar pruebas de funcionamiento.

3. Pruebas de esfuerzo del prototipo del sistema

- Instalar el Software SIPp dentro de la máquina virtual Ubuntu 22.04.
- Crear el archivo XML con los parámetros para las pruebas de esfuerzo.
- Ejecutar el archivo XML con SIPp
- Verificar los archivos LOG para verificar los resultados obtenidos en cada uno de los escenarios.
- Determinar los parámetros de uso de recursos, latencia y tiempo de respuesta.

3.9. Desarrollo

Para lograr el desarrollo y evaluación del prototipo en GNS3 se integró Asterisk y PostgreSQL sobre IPv6 para la consulta de notas académicas, se realizó una serie de procedimientos que se detallan a continuación. En primer lugar, se realizó una exhaustiva revisión de la literatura técnica y científica relacionada con la integración de Asterisk y PostgreSQL, así como con el despliegue de IPv6 en entornos de comunicaciones. Esta revisión proporcionó un marco conceptual concreto y permitió identificar los mejores recursos y métodos que permitían la integración de estas tecnologías. Posteriormente, se procedió a la configuración y puesta en marcha del entorno de simulación GNS3, donde se desarrolló el prototipo del sistema de red. Este proceso incluyó la instalación y configuración de Asterisk y PostgreSQL sobre IPv6, así como la implementación de los archivos, librerías, programas y complementos necesarios para la consulta de notas académicas. Además, se establecieron dos escenarios de prueba distintos para medir el rendimiento del sistema: uno que implicaba una conexión directa entre la base de datos y Asterisk, y otro que utilizaba un servicio web mediante AGI para la conexión. Una vez generado el sistema mediante los escenarios planteados se procedió a ejecutar las pruebas de rendimiento mediante el Software SIPp en donde se evaluaron variables como el tiempo de respuesta, el uso de recursos y la latencia con el objetivo de evaluar el sistema que mejor se adaptase a las condiciones de un entorno educativo. Finalmente se analizaron los resultados para poder determinar las conclusiones y trabajos futuros surgidos a través del presente trabajo. El desarrollo e implementación del prototipo se encuentra detallado en el Anexo 1.

3.9.1. Arquitectura funcional del prototipo

En la figura 9, se muestra el escenario de red con protocolo IPv6 simulado en GNS3, donde se encuentra el softphone Zoiper el cual va a permitir realizar una llamada y conectar mediante una conexión directa por OBDC o por una conexión indirecta por Webservice con Asterisk para realizar la consulta al gestor de base de datos PostgreSQL donde se encuentra la base de datos del Sistema de Notas Académicas.

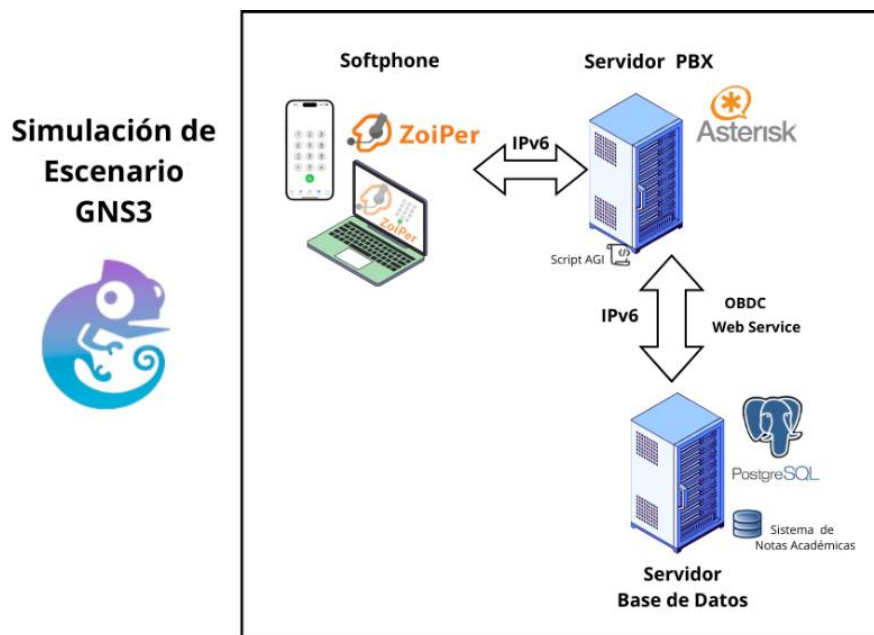


Figura 9. Arquitectura funcional del prototipo

3.9.2. Topología de red IPv6 del prototipo en GNS3

En la siguiente topología de red, se utilizó equipos activos de red con protocolo IPv6, la cual está compuesta fundamentalmente por cuatro segmentos.

En la Figura 10 se observa, el primer segmento se encuentra ubicados los clientes, mientras que en el segundo segmento está el servidor de Asterisk. El tercer segmento aloja el gestor de base de datos y, finalmente, en el cuarto segmento se sitúa el software de esfuerzo. Posteriormente, se asignó direccionamiento IPv6 a todas las interfaces de los routers, así como a los clientes y servidores, con el objetivo de probar conectividad entre todos y cada uno de los elementos de red.

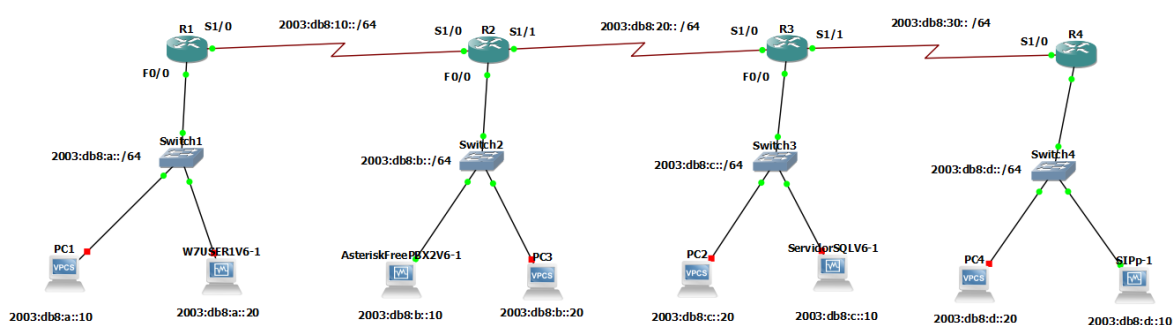


Figura 10. Topología de red ipv6 del prototipo

En la Figura 11 se observa que se asignaron direcciones IP estáticas con IPv6 sobre los cliente Zoiper y los servidores como: Asterisk, PostgreSQL y la herramienta de esfuerzo SIPp.

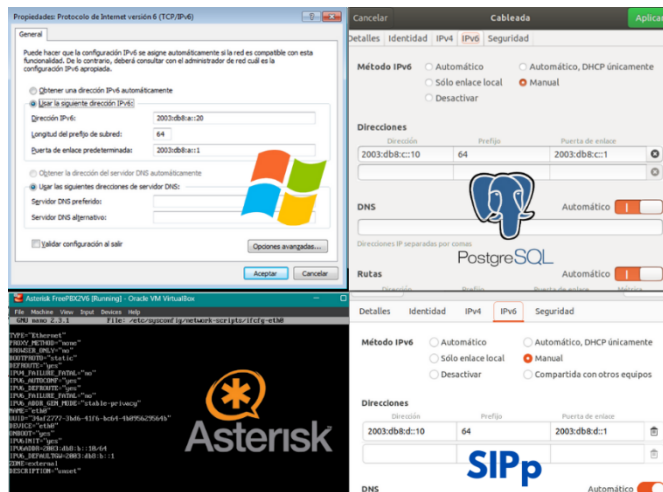


Figura 11. Configuración de direcciones de red estática con IPv6

3.9.3. Implementación de servidor PostgreSQL con IPv6

Para la implementación del servidor PostgreSQL se seleccionó una máquina virtual con sistema operativo Ubuntu 18.04, donde se instaló PostgreSQL versión 10.23.

En la Figura 12 se divide en dos imágenes, en la primera se observa al archivo pg.hba.conf, configurado el método y la dirección IPv6 del servidor de Asterisk para que exista interacción entre estos dos servidores. Mientras que en el archivo postgresql.conf que está en la segunda imagen se configuró el puerto, permisos para el socket de dominio Unix y la dirección IP del servidor de PostgreSQL con su dirección IPv6.

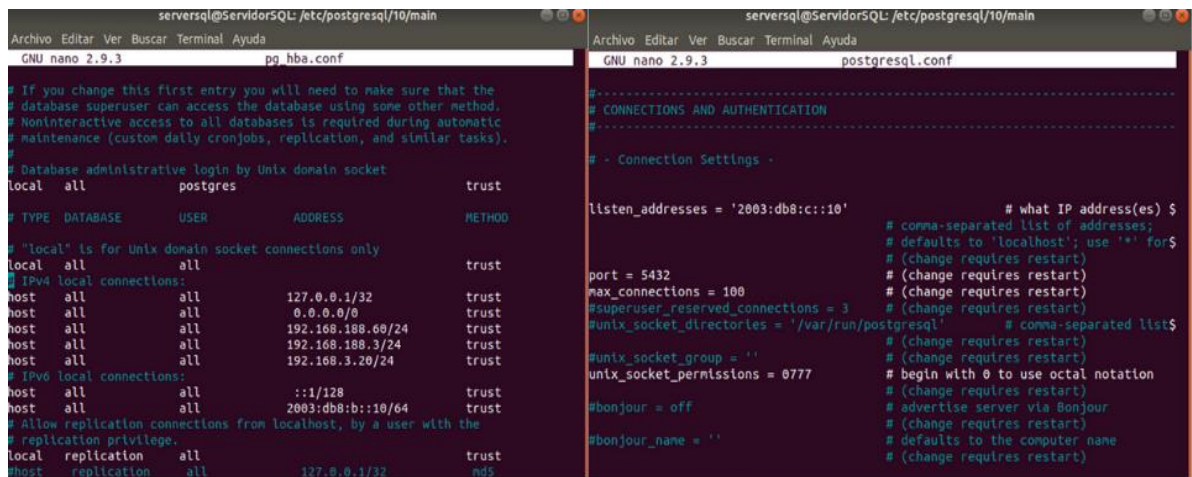
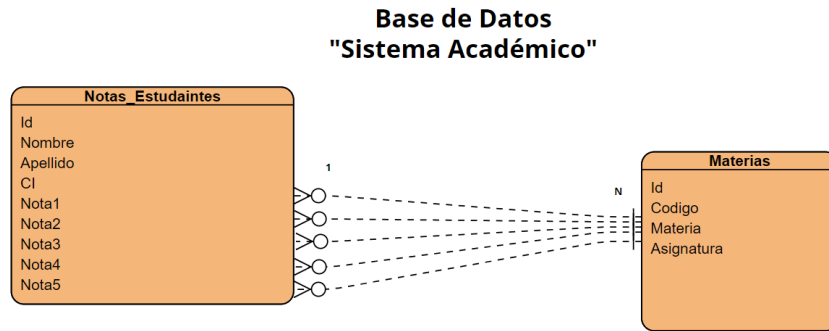


Figura 12. Configuración de PostgreSQL con IPv6

Diseño de la base datos



En la

Figura 13 se observa el diseño de la base de datos denominado “Sistema Académico” se utilizó dos tablas, la tabla denominada “Notas_Estudiantes” la cual tiene los siguientes atributos: id, nombre, apellido, CI, Nota1, Nota2, Nota3, Nota4 Nota5. Mientras que en la segunda tabla denominada “Materias” tiene como atributo: id, código, materia y asignatura. El atributo materia en la tabla Notas_Estudiantes es una clave externa y hace referencia al atributo "id" en la tabla Materias. Esto permitió enlazar las dos tablas y obtener información completa sobre las notas de cada alumno en cada materia existiendo una relación de 1: N, como se observa

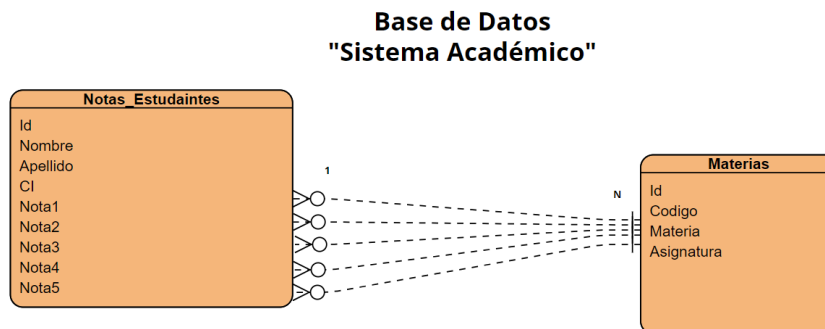


Figura 13. Diseño de base de datos relacional sistema académico.

3.10. Implementación de Servidor Asterisk

En la Figura 14 se compone de dos imágenes, la implementación del servidor de Asterisk para la cual seleccionó una máquina virtual de Sangoma con sistema operativo Linux 7 en el cual viene preinstalado Asterisk. Una vez instalado se procedió a ingresar al archivo donde la primera muestra el archivo manager.conf, donde se debe configurar la línea bindaddr colocando '::' para establecer conexión con direcciones IPv6. En la segunda imagen, correspondiente al archivo sip_general_additional.conf, se editó la línea tlsbindaddr y se colocó ':::5161'. Además, en la misma imagen se asigna la dirección IPv6 en localnet.

```

File Machine View Input Devices Help
GNU nano 2.3.1 File: manager.conf

AMI - Asterisk Manager interface - Generated at 2023-12-28T00:14:11+00:00
FreePBX needs this to be enabled. Note that if you enable it on a different IP, you need
to assure that this can't be reached from un-authorized hosts with the ACL settings (permit/deny).
Also, remember to configure non-default port or IP-addresses in amportal.conf.

The AMI connection is used both by the portal and the operator's panel in FreePBX.

FreePBX assumes an AMI connection to localhost:5038 by default.

[general]
enabled = yes
port = 5038
bindaddr = ::
displayconnects=no :only effects 1.6+

[admin]
secret = aBk/vhzeZfIt
deny=0.0.0/0.0.0.0
permit=127.0.0.1/255.255.255.0
read = system,call,log,verbose,command,agent,user,config,command,dtmf,reporting,cdr,dialplan,originate
write = system,call,log,verbose,command,agent,user,config,command,dtmf,reporting,cdr,dialplan,originate
writeprotect = 0
writeprotecttimeout = 5000

include manager_additional.conf
include manager_custom.conf

GNU nano 2.3.1 File: sip_general_additional.conf

bindport=5160
jbenable=no
checksum=10
maxexpiry=3600
minexpiry=60
srvlookup=no
allowguest=yes
notifyhold=yes
rtptimeout=30
canreinvite=no
rtptimeout=30
videosupport=no
defaultexpiry=120
notifying=yes
maxcallbitrate=304
rtptimeout=300
g726nonstandard=no
registertimeout=20
registerattempts=0
nat=force_rport,comedia
ALLOW_SIP_ANON=no
bindaddr=::1:5161
tlscafile=/etc/pki/tls/certs/ca-bundle.crt
localnet=192.168.100.0/24
localnet=192.168.3.0/24
localnet=12803:4001::91/64
accept_outofcall_message=yes
auth_message_requests=no
outofcall_message_context=dpmo_message_context

```

Figura 14. Configuraciones de Asterisk con IPv6

3.10.1. Instalación de TEXT TO SPEACH (TTS)

En la Figura 15 se observa que para poder ejecutar instrucciones de lectura de base de datos se requería una aplicación que permitiera que el texto generado en tiempo real se convierta en audio, es decir, un sistema de Text to Speech (TTS). Para esto se ejecutó el comando `sudo yum install festival festival-devel` dentro del servicio de Asterisk. Se configuró el archivo `festival.conf` y se verificó la configuración del comando y el host al cual estaba direccionado el servicio.

```

[general]
host=localhost
port=1314
festivalcommand=(tts_textasterisk "%s" 'file')(quit)\n

```

Figura 15. Configuración básica de servidor festival

3.10.2. Creación de los canales SIP

En la Figura 16 se observa que habilita el canal 1888 para permitir al cliente Softphone Zoiper efectuar la consulta de notas. En este canal, dentro del contexto `ivr_academico` en el plan de marcado, se encuentra habilitado el IVR que permitirá el acceso a los dos escenarios de conexión hacia la base de datos PostgreSQL utilizando IPv6.

```

GNU nano 2.3.1 File: pjsip.endpoint.conf
-----
;
; Do NOT edit this file as it is auto-generated by FreePBX.
;
; For information on adding additional paramaters to this file, please visit
; FreePBX.org wiki page, or ask on IRC. This file was created by the new F
; BMO - Big Module Object. Any similarity in naming with BMO from Adventure
; is totally deliberate.
;
-----
#include pjsip.endpoint_custom.conf

[1888]
type=endpoint
aors=1888
auth=1888-auth
tos_audio=ef
tos_video=af41
cos_audio=5
cos_video=4
allow=ulaw,alaw,gsm,g726,g722
context=ivr_academico
callerid=Usuario1 <1888>
dtmf_mode=auto
aggregate_mwi=yes
use_avpf=no
rtcp_mux=no
ice_support=no
media_use_received_transport=no
trust_id_inbound=yes
user_eq_phone=no
send_connected_line=yes
media_encryption=no
timers=yes

```

Figura 16. Configuración de extensiones SIP en Asterisk 13.38

3.10.3. Plan de marcado

En la Figura 17 se observa se genera un Dialplan en el archivo extensions_custom.conf, que define el flujo y las acciones a tomar durante una llamada. Se ha creado dos planes de llamada [consulta_bd].

```

GNU nano 2.3.1 File: /etc/asterisk/extensions_custom.conf

[ivr_academico]
include => consulta_bd
include => consulta_webservice

[consulta_bd]
exten=> 2001,1,Answer()
exten=> 2001,2,Playback(Bienvenida)
exten=> 2001,3,Playback(Cedula)
exten=> 2001,4,Read(CI)
exten=> 2001,5,Set(nombre=${ODBC_nombre(${CI}})
exten=> 2001,6,GotoIf($[${nombre}=""?]fca1:cca1)
exten=> 2001,7(cca1),Set(apellido=${ODBC_apellido(${CI}})
exten=> 2001,8(cca1),Festival(ESTUDIANTE)
exten=> 2001,9(cca1),Festival(${nombre})
exten=> 2001,10(cca1),Festival(${apellido})
exten=> 2001,11(cca1),Playback(Opcion1)
exten=> 2001,12(cca1),Playback(Opcion2)
exten=> 2001,13(cca1),WaitExten(3)
exten=> 2001,14(cca1),Goto(2001,11)
;CEDULA INCORRECTA
exten=> 2001,n(fca1),Festival(Numero de Cedula Incorrecto)
exten=> 2001,n(fca1),Festival(Saliendo del Sistema)
exten=> 2001,n(fca1),Hangup
;SI SELECCIONA 1 ENTRA A ESTE CODIGO
exten=> 1,1,Answer()
exten=> 1,n,Festival(INGRESE EL CODIGO DE ASIGNATURA)
exten=> 1,n,Read(codigo)
exten=> 1,n,Festival(CUN MOMENTO PORFAVOR)
exten=> 1,n,Set(materia=${ODBC_codasignatura(${CI},${codigo}})
exten=> 1,n,GotoIf($[${materia}=""?]nmat:smat)
exten=> 1,n(smat),Festival(BUSCANDO INFORMACION)
exten=> 1,n(smat),Set(asignatura=${ODBC_asignatura(${CI},${codigo}})

```

Figura 17. Archivo del plan de marcado consulta_db

En la Figura 18 se observa se genera un Dialplan, que define el flujo y las acciones a tomar durante una llamada, que contiene la conexión por webservice [consulta_webservice] que tiene el plan de llamada del WebService.

```

[consulta_webservice]
exten => 2002,1,Answer()
exten => 2002,2,Playback(Bienvenida)
exten => 2002,3,Playback(Cedula)
exten => 2002,4,Read(CI)
exten => 2002,5,Festival(Buscando Informacion, Un momento Porfavor)
exten => 2002,6,Agi(/var/lib/asterisk/agi-bin/prueba.php,${CI})
exten => 2002,7,Gotoif(${I}${validacion}"=No existe"?cal1:cal2)
exten => 2002,8(cal2),Festival('Estudiante ${nombre} ${apellido}')
exten => 2002,9(cal2),Festival('Presione tres si desea conocer la nota por codigo')
exten => 2002,10(cal2),Festival('Presione cuatros si de desea conocer todas las notas')
exten => 2002,11(cal2),WaitExten(3);
exten => 2002,12(cal2),Goto(2002,9);
:CEDULA INCORRECTA
exten => 2002,n(cal1), Festival(Numero de cedula incorrecta)
exten => 2002,n(cal1), Festival(Saliendo del Sistema)
exten => 2002,n(cal1), Hangup

;SI INGRESA POR LA OPCION 3
exten => 3,1,Answer()
exten => 3,n, Festival(Ingrese el codigo de la asignatura)
exten => 3,n, Read(CODIGO)
exten => 3,n, Festival(Buscando informacion)

```

Figura 18. Archivo del plan de marcado consulta_webservice

3.11. Integración del Sistema con ODBC (Escenario I)

En la

```

[SISTEMA_ACADEMICO_PSQL]
Description = PostgreSQL connection to Alumnos
Driver      = PostgreSQL
Database    = Sistema_Academico
Servername  = 2003:db8:c::10
UserName    = admin_
Password    = 1234_
Port        = 5432

```

Figura 19 se muestra la configuración para establecer comunicación entre los servidores de Asterisk y PostgreSQL se utilizó un socket ODBC. Para ello, se instaló el driver utilizando el comando "sudo yum install odbc-postgresql". En el archivo /etc/odbc.ini se configuró la conexión con la base de datos como: usuario, contraseña, puerto, dirección IPv6 y nombre de la base de datos

```

[SISTEMA_ACADEMICO_PSQL]
Description = PostgreSQL connection to Alumnos
Driver      = PostgreSQL
Database    = Sistema_Academico
Servername  = 2003:db8:c::10
UserName    = admin_
Password    = 1234_
Port        = 5432

```

Figura 19. Conexión de PostgreSQL con Asterisk a través de ODBC

En la Figura 20 se muestra el contenido específico del archivo res_odbc_custom.conf, donde se realizó la configuración para la inicialización del socket para establecer conexión con el driver correspondiente.


```
GNU nano 2.3.1 File: res_odbc_custom.conf
[ALUMNOSPSQL]
enabled=>yes
dsn=>SISTEMA_ACADEMICO_PSQL
pre-connect=>yes
```

Figura 20. Configuración para la conexión de una base de datos con ODBC

En la Figura 21, se visualiza el archivo func_odbc_conf, donde se lleva a cabo la programación de funciones para la búsqueda de información y extracción de variables desde la base de datos.

```
GNU nano 2.3.1 File: func_odbc.conf
[nota1]
dsn=SISTEMA_ACADEMICO_PSQL
readsql=select nota1 from Notas_Estudaintes where CI='${ARG1}';
[nota2]
dsn=SISTEMA_ACADEMICO_PSQL
readsql=select nota2 from Notas_Estudiantes where CI='${ARG1}';
[nota3]
dsn=SISTEMA_ACADEMICO_PSQL
readsql=select nota3 from Notas_Estudaintes where CI='${ARG1}';
[nota4]
dsn=SISTEMA_ACADEMICO_PSQL
readsql=select nota4 from Notas_Estudiantes where CI='${ARG1}';
[nota5]
dsn=SISTEMA_ACADEMICO_PSQL
readsql=select nota5 from Notas_Estudiantes where CI='${ARG1}';
[nombre]
dsn=SISTEMA_ACADEMICO_PSQL
readsql=select nombre from Notas_Estudiantes where CI='${ARG1}';
[apellido]
dsn=SISTEMA_ACADEMICO_PSQL;
readsql=select apellido from Notas_Estudiantes where CI='${ARG1}';
[codasignatura]
dsn=SISTEMA_ACADEMIC_PSQL
readsql=select materia from Materias where codigo='${ARG2}';
[asignatura]
dsn=SISTEMA_ACADEMICO_PSQL
readsql=select asignatura from Materias where codigo='${ARG2}';
```

Figura 21. Funciones para la extracción de registros de la base de datos con ODBC

3.12. Integración del Sistema con Webservice (Escenario II)

En la Figura 22 se muestra la creación de un archivo llamado prueba.php en var/lib/asterisk/agi-bin. Este archivo ejecutará la función AGI y contiene el algoritmo para obtener y extraer información, así como la dirección URL para establecer conectividad con la base de datos.

```

GNU nano 2.3.1      File: /var/lib/asterisk/agi-bin/prueba.php
#!/usr/bin/php -q
<?php
require_once(__DIR__ . '/phpagi/phpagi.php');
include('http://[2003:db8:c::10]:80/pruebacon.php');
// TimeZone
date_default_timezone_set('America/Bogota');
ob_implicit_flush(true);
// Execution Timeout
set_time_limit(30);
// New AGI Object
$agi = new AGI();
$CI=$SERVER['argv'][1];
$agi->verbose($CI);
$query="SELECT * FROM NOTAS1 WHERE ci='$CI'";
$result=pg_query($query);
$row=pg_fetch_array($result);
if ($row!=NULL)
{
    $nombre=$row[1];
    $apellido=$row[2];
    $nota1=$row[4];
    $nota2=$row[5];
}
else
{
    $validacion="No existe";
}
$agi->set_variable("validacion",$validacion);
$agi->set_variable("nombre",$nombre);
$agi->set_variable("apellido",$apellido);
$agi->set_variable("nota1",$nota1);
$agi->set_variable("nota2",$nota2);

```

Figura 22. Archivo PHP AGI para la búsqueda dentro de la base de datos

3.13. Implementación de Cliente Softphone

En la Figura 23 se observa para la implementación de Softphone Zoiper se instaló en el Sistema Operativo de Windows 7, mismo que permita la ejecución del Softphone que servirá como usuario para la consulta de notas académicas. Se instaló el software denominado Zoiper5 donde se establece como dominio del servidor la dirección IPv6 del servidor Asterisk, también se agrega el Username y el Password, se da clic en el botón Register y se espera que se establezca la conexión con el servidor.

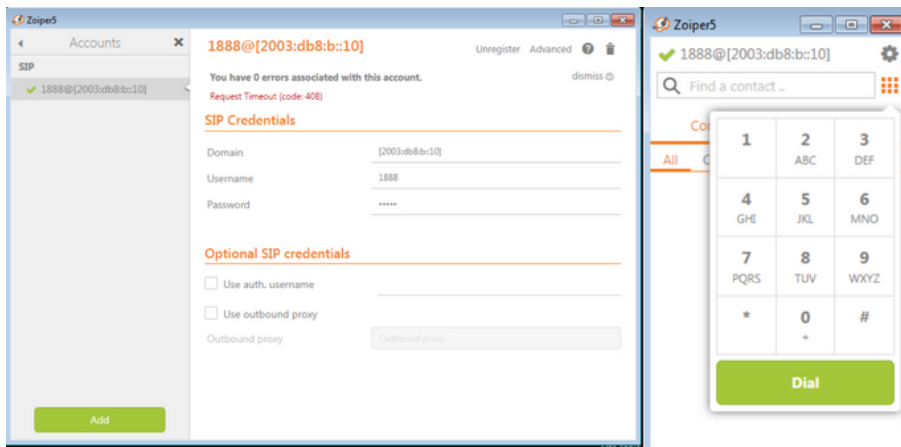


Figura 23. Configuración de extensión y dirección del dominio del servicio

CAPÍTULO IV. RESULTADOS Y DISCUSIÓN

4.1. Resultados de Integración de Plataformas

Antes de la implementación del prototipo mediante el uso del simulador GNS3, se realizó una investigación bibliográfica sobre las tecnologías para la integración de Asterisk y PostgreSQL a través del IPv6, de la cual se obtuvo información relevante que permitió identificar las alternativas respectivas para establecer una comunicación entre Asterisk y PostgreSQL.

En la Tabla 2 se pueden observar las diferentes tecnologías que permiten la integración de servicios entre la base de datos PostgreSQL y Asterisk. Para el caso del primer escenario existen opciones de driver de conexión ODBC y JDBC, siendo el más adecuado para el prototipo la primera opción, debido a que únicamente requería la instalación del Driver correspondiente a la base de datos PostgreSQL, mientras que el otro requería de una serie de componentes pertenecientes a JAVA que dificultarían la implementación por compatibilidad. Así también se observó que para el escenario 2 o WebService se encontraron igualmente dos alternativas para la integración entre los servicios. El primero era AGI, mismo que se encontraba incluido en las librerías de Asterisk y que permitía la lectura de código PHP para la lectura de los datos dentro de PSQL, siendo esta la más adecuada para el proyecto. Por otro lado, la opción de FastAGI tenía comandos diferentes a la primera opción y requería la instalación y descarga de archivos que permitieran la vinculación de esta interfaz con Asterisk.

Tabla 2. Comparativa de Tecnologías de Integración

Escenario	Tecnologías de integración	Observaciones
Conexión directa a base de datos	Driver ODBC	Requiere un Driver específico para cada base de datos
	Driver JDBC	Requiere de Servicios y Aplicaciones compatibles con JAVA
Conexión mediante WebService	AGI	Interfaz más utilizada para la integración con otros servicios que puede utilizar PHP, Python.
	FAST AGI	Requiere instalación de librerías adicionales y los comandos son diferentes con respecto a AGI.

4.2. Resultados de la Implementación del Prototipo

En la Figura 24 se observa una vez establecidos los métodos de integración a utilizarse para los dos escenarios de prueba, se procedió a comprobar el funcionamiento del prototipo

ejecutando pruebas de búsqueda en ambos escenarios. Para ejecutar las pruebas del escenario 1 se marcó dentro del Softphone la línea 2001. Se obtuvo la respuesta por parte del Asterisk del DialPlan diseñado, para lo cual se ingresó el número de cédula del estudiante registrado en la base de datos y se devolvió la nota de la materia seleccionada, comprobando así el correcto funcionamiento del sistema mediante ODBC o conexión directa.

En la Figura 24 se observa que para el caso del escenario 2 o Webservice, el sistema requería que el usuario marcara la línea 2002. Se obtuvo la respuesta por parte del servidor Asterisk verificando los comandos y ejecución del script externo implementado mediante PHP por medio de AGI. Dentro de esta prueba de funcionamiento se seleccionó un número de cédula aleatorio de los alumnos registrados en PSQL y se solicitó al sistema la entrega de todas las notas registradas del usuario, obteniendo así la correcta respuesta del sistema.

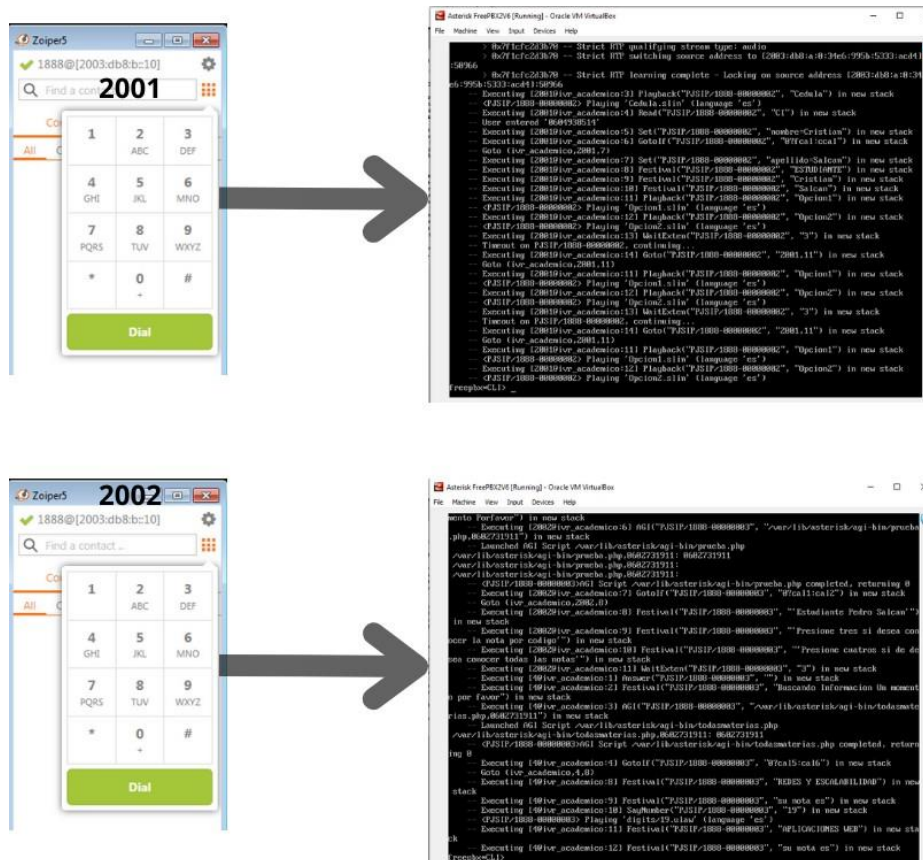


Figura 24. Pruebas de funcionamiento del sistema

4.3. Pruebas de Desempeño

Una vez que se desarrolló y comprobó el funcionamiento del prototipo, se procedió a la evaluación del desempeño para determinar cuál de los dos escenarios implementados tenía mejor respuesta ante una prueba de estrés. Mediante el software de esfuerzo SIPp se implementaron pruebas que permitieron determinar cuál tenía mejor rendimiento.

En la Figura 25 se pudo observar que se realizó mediante la implementación de un nuevo segmento de red en la topología original definida. Dentro de este nuevo segmento de red se colocó un servidor de Ubuntu versión 22.04 que contendrá los archivos del programa SIPp

que evaluará el sistema. Esta herramienta determino y forzó los escenarios implementados tanto de WebService como de conexión directa a la base de datos; mediante varias llamadas concurrentes.

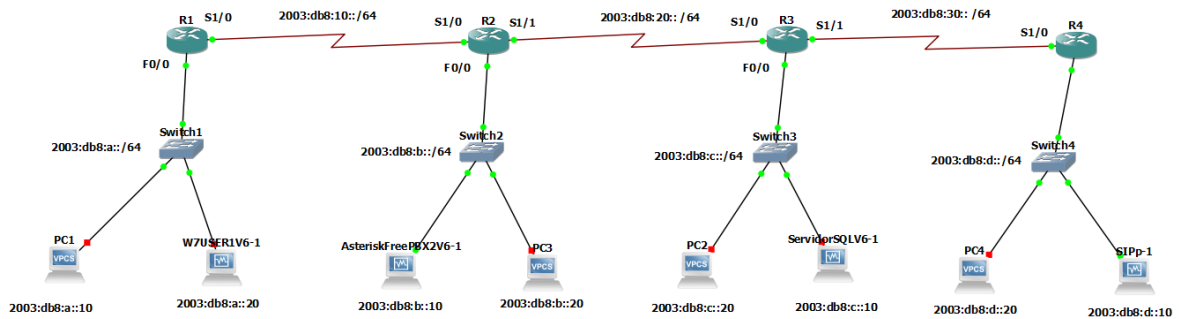


Figura 25. Esquema de red para pruebas de estrés con SIPp

Para poder determinar los parámetros de medición de latencia y tiempo de respuesta con los datos que SIPp proporciona, para evaluarlos y compararlos se realizó una búsqueda de información sobre la interpretación y estimación de cada una de las variables. En primer lugar, para lograr estimar el valor de latencia, se analizan los paquetes recibidos y paquetes transmitidos por parte del servidor (Minda & Pacheco, 2019). Mediante el uso de los archivos de respaldo de SIPp se puede determinar el tiempo de ida y vuelta de los paquetes SIP que son devueltos del servidor de Asterisk mediante la siguiente fórmula:

$$l = T_{\text{envío}} - T_{\text{recepción}}$$

Por otro lado, para poder determinar el tiempo de respuesta utilizando SIPp se identifica el parámetro CPS que indica la cantidad de llamadas por segundo que puede ejecutar el prototipo. Con esto en mente, dentro del documento se indica que para determinar el tiempo de respuesta entre llamadas se debe utilizar el promedio de CPS y estimar el tiempo con respecto a una llamada (Voznak & Rozhon, 2010).

4.3.1. Pruebas y Resultados Escenario I

En la Figura 26 se procedió a establecer una cantidad de 1000 llamadas por prueba hasta evidenciar problemas de comunicación o llamadas que no puedan ser recibidas. Una vez establecido el protocolo de pruebas planteado se procede a la verificación del estado de recursos antes de realizar el test de estrés, mediante el uso del comando TOP. Se muestra que el servidor de Asterisk tiene un consumo del 2.3% del CPU y un 5.2% de la memoria.

PID	USER	PR	NI	UIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2851	asterisk	20	0	1680416	52756	18628	S	2.3	5.2	0:02.55	asterisk
2162	mongodb	20	0	478856	54580	20604	R	1.0	5.4	0:02.55	mongod
9	root	20	0	0	0	0	S	0.3	0.0	0:00.87	rcu_sched
25	root	20	0	0	0	0	S	0.3	0.0	0:01.73	kworker/0:1
1368	root	20	0	0	0	0	S	0.3	0.0	0:00.42	xfsaild/dm-0
2154	mysql	20	0	1171480	96320	8492	S	0.3	9.5	0:01.57	mysqld
3322	asterisk	20	0	918372	26364	11676	S	0.3	2.6	0:00.52	PM2 v2.10.7: Go

Figura 26. Uso de recursos pretest escenario 1

Para efectuar la prueba de estrés para 100 llamadas se utiliza el siguiente comando dentro de la terminal del servidor de SIPp: `sipp -sf prueba4.xml -s 2001 -m 1000 -r 40 -l 80 -i [2003:db8:d::10] [2003:db8:b::10] -trace_msg`. Dentro de este comando se especifica el archivo XML asignado para todo el protocolo de pruebas, se selecciona la extensión del servidor a la que se va a realizar la llamada, la cantidad de llamadas a realizarse (-m 1000), el número de llamadas por segundo (-r 40) y las direcciones de IP tanto del servidor como del cliente.

En la Figura 27 se puede observar la respuesta del software SIPp con una cantidad de 1000 llamadas en dónde se muestra que se realizaron 606 llamadas efectivas, con un promedio de 2.386 llamadas por segundo. Con este valor podemos obtener el tiempo de respuesta entre cada llamada, ya que una se quiere conocer cuánto tiempo es el intervalo entre llamadas que tarda en responder el servidor, generando un valor de 41.910 ms entre cada llamada.

Counter Name	Periodic value	Cumulative value
Last Reset Time	2024-04-23 11:15:27.943704 1713888927.943704	
Current Time	2024-04-23 11:15:28.540946 1713888928.540946	
Elapsed Time	00:00:00:597000	00:00:00:597000
Call Rate	0,000 cps	2,386 cps
Incoming calls created	0	0
Outgoing calls created	0	1000
Total Calls created		1000
Current Calls	3	
Successful call	0	606
Failed call	0	391
Call Length	00:00:00:000000	00:00:00:000000

----- Test Terminated -----

Figura 27. Respuesta software SIPp en 1000 llamadas escenario 1

Posteriormente se procede a observar los archivos de logs que genera la simulación, en dónde se muestran los encabezados de los paquetes transmitidos y recibidos entre el simulador con el servidor de Asterisk. Esta información de tiempo que genera estos paquetes permite el cálculo de la latencia existente en el sistema.

En la Figura 28 para estas pruebas existe una latencia que se puede obtener restando la hora del paquete recibido menos la hora del paquete enviado que para este caso es de 2.4955 ms.

```

94
95 ----- 2024-04-23 11:08:29.617354
96 UDP message sent (380 bytes):
97
98 INVITE sip:2001@[2003:db8:b::10]:5060 SIP/2.0
99 Via: SIP/2.0/UDP [2003:db8:d::10]:5060
100 From: 1888 <sip:1888@[2003:db8:d::10]:5060>;tag=4386SIPpTag004
101 To: 2001 <sip:2001@[2003:db8:b::10]:5060>
102 Call-ID: 4-4386@2003:db8:d::10
103 CSeq: 1 INVITE
104 Contact: sip:1888@[2003:db8:d::10]:5060
105 Max-Forwards: 70
106 Subject: Performance Test
107 Content-Type: application/sdp
108 Content-Length: 0
109
110
111 ----- 2024-04-23 11:08:29.620622
112 UDP message received [449] bytes :
113
114 SIP/2.0 401 Unauthorized
115 Via: SIP/2.0/UDP [2003:db8:d::10]:5060;rport=5060;received=2003:db8:d::10
116 Call-ID: 2-4386@2003:db8:d::10
117 From: "1888" <sip:1888@[2003:db8:d::10]>;tag=4386SIPpTag002
118 To: "2001" <sip:2001@[2003:db8:b::10]>
119 CSeq: 1 INVITE
120 WWW-Authenticate: Digest realm="asterisk",nonce="1713888511/
d55b7096945c21abcf185aa0c841b420",opaque="6cedf38913bea3ae",algorithm=md5,qop="auth"
121 Server: FPRX-14.0-17(13.38.3)

```

Figura 28. Respuesta de log de SIPp para calcular la latencia del escenario 1

En la Figura 29 se observa finalmente que se requiere evaluar el comportamiento de los recursos utilizados por el servidor mediante el uso del comando top. que el porcentaje del uso del CPU aumentó a 7.7% y de la Memoria a un 6.2%.

PID	USER	PR	NI	UIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2149	mysql	20	0	1171480	102332	8492	S	7.7	10.1	0:02.88	mysql
2884	asterisk	20	0	1680816	62620	19056	S	7.7	6.2	0:03.32	asterisk
3	root	20	0	0	0	0	S	1.0	0.0	0:01.86	ksoftirqd/0
2157	mongod	20	0	478860	56708	20624	S	1.0	5.6	0:02.22	mongod
3190	asterisk	20	0	1075780	44048	14432	S	1.0	4.3	0:01.19	node /var/www/h

Figura 29. Prueba de uso de recursos con 1000 llamadas en escenario 1

En la Tabla 3 se observa de la misma manera y como se mencionó anteriormente, se procede a evaluar todos los parámetros de Tiempo de Respuesta, Uso de Datos y Latencia para cada una de las diferentes cantidades de llamadas, en donde se aumentó de 200 en 200 hasta generar problemas dentro del servidor. Se recopiló toda la información de las pruebas obtenidas identificando que para una cantidad de 1000 llamadas el sistema ya genera problemas con únicamente 606 llamadas efectuadas, es decir se tiene un 39.4% de errores.

Tabla 3. Recopilación de resultados obtenidos del escenario 1 hasta 1000 llamadas

ESCENARIO DE PRUEBA 1 - CONEXIÓN DIRECTA										
N° Prueba	Cantidad de Llamadas	Llamadas Correctas	% Llamadas Correctas	Llamadas por Segundo	Tiempo de Respuesta Llamadas (ms)	Paquete 1	Paquete 2	Latencia (ms)	CPU %	RAM %
1	200	195	97,5	5	20	41,4575	41,4402	1,73	7,65%	6,56%
2	400	313	78,25	5,347	18,7020	9,2569	9,2468	1,0118	9,60%	6,60%
3	600	428	71,3333	5,242	19,0766	51,7612	51,7565	0,4727	9,30%	6,60%
4	800	532	66,5	2,339	42,7533	1,1009	1,05374	4,7208	8,00%	6,60%
5	1000	606	60,6	2,386	41,9111	29,5655	29,5405	2,4955	7,70%	6,70%

4.3.2. Pruebas y Resultados Escenario II

En la Figura 30 se muestra que, de manera similar al escenario 1, se realizan la visualización del estado de recursos del servidor previo a realizar el test de estrés, mediante el uso del comando TOP. Se muestra que el servidor de Asterisk tiene un consumo del 7.7% del CPU y un 6.7% de la memoria. Posteriormente se procedió a establecer una cantidad de 1000 llamadas por prueba que exista problemas en la recepción de llamadas o existan algunas que no se efectúen.

PID	USER	PR	NI	UIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2884	asterisk	20	0	1680816	67500	19108	S	7.7	6.7	0:11.81	asterisk
2149	mysql	20	0	1171480	103428	8496	S	6.7	10.2	0:09.93	mysqld
3190	asterisk	20	0	1076804	58068	14436	S	1.3	5.7	0:02.47	node /var/www/h
2157	mongodb	20	0	478860	56708	20624	S	1.0	5.6	0:03.55	mongod
3	root	20	0	0	0	0	S	0.7	0.0	0:02.63	ksoftirqd/0
93	root	20	0	0	0	0	S	0.7	0.0	0:02.78	kworke/0:3
9	root	20	0	0	0	0	S	0.3	0.0	0:01.08	rcu_sched

Figura 30. Estado de recursos del servidor pretest escenario 2

Para realizar el test de estrés en el escenario de prueba de WebService con una cantidad inicial de 200 llamadas se utiliza el siguiente comando dentro de la terminal del servidor de SIPp: `sipp -sf prueba4.xml -s 2001 -m 1000 -r 40 -l 80 -i [2003:db8:d::10] [2003:db8:b::10] -trace_msg`. Dentro de este comando se especifica el archivo prueba4.XML, se selecciona la extensión del servidor a la que se va a realizar la llamada en este caso 2002 la que corresponde al WebServices implementado, la cantidad de llamadas a realizarse (-m 1000), el número de llamadas por segundo (-r 40) y las direcciones IPv6 tanto del servidor como del cliente.

En la Figura 31 se observa que una vez ejecutado y culminado el test de estrés para 1000 llamadas en el escenario 2, se procede a visualizar los resultados obtenidos por el simulador. Dentro de estos se muestra qué se tiene un promedio de 2,953 de llamadas por minuto, lo que permite estimar el tiempo de respuesta para cada una de las llamadas, en este caso, corresponde a 33,86 ms.

Counter Name	Periodic value	Cumulative value
Last Reset Time	2024-04-23 11:53:22.744566	1713891202.744566
Current Time	2024-04-23 11:53:23.345265	1713891203.345265
Elapsed Time	00:00:00:600000	00:00:00:600000
Call Rate	0,000 cps	2,953 cps
Incoming calls created	0	0
Outgoing calls created	0	1000
Total calls created		1000
Current Calls	2	
Successful call	0	546
Failed call	0	452
Call Length	00:00:00:000000	00:00:00:000000

Test Terminated

Figura 31. Resultado de prueba estrés 100 llamadas en escenario 2

En la Figura 32 se observa que de igual forma que se lo realizó en el escenario 1, para poder estimar la latencia existente en el servicio con estas pruebas de estrés se visualizan los

archivos logs generados por el simulador, en donde se toma la hora del paquete recibido y se resta de la hora del paquete transmitido obteniendo un valor de latencia de 0.2661 ms.

```

41072 Content-Length: 0
41073
41074
41075 ----- 2024-04-23 11:48:01.275311
41076 UDP message sent (384 bytes):
41077
41078 INVITE sip:2002@[2003:db8:b::10]:5060 SIP/2.0
41079 Via: SIP/2.0/UDP [2003:db8:d::10]:5060
41080 From: 1888 <sip:1888@[2003:db8:d::10]:5060>;tag=5068SIPpTag00133
41081 To: 2002 <sip:2002@[2003:db8:b::10]:5060>
41082 Call-ID: 133-5068@2003:db8:d::10
41083 CSeq: 1 INVITE
41084 Contact: sip:1888@[2003:db8:d::10]:5060
41085 Max-Forwards: 70
41086 Subject: Performance Test
41087 Content-Type: application/sdp
41088 Content-Length: 0
41089
41090
41091 ----- 2024-04-23 11:48:01.277972
41092 UDP message received [961] bytes :|
41093
41094 SIP/2.0 200 OK
41095 Via: SIP/2.0/UDP [2003:db8:d::10]:5060;rport=5060;received=2003:db8:d::10
41096 Call-ID: 103-5068@2003:db8:d::10
41097 From: "1888" <sip:1888@[2003:db8:d::10]>;tag=5068SIPpTag00103
41098 To: "2002" <sip:2002@[2003:db8:b::10]>;tag=115a152b-5639-412b-9039-b19e03133b10
41099 CSeq: 2 INVITE
41100 Server: FPBX-14.0.17(13.38.3)

```

Figura 32. Respuesta de log de SIPp para calcular la latencia del escenario 2

En la Figura 33 se observa que, al realizar un análisis de la cantidad de recursos utilizados durante esta prueba dentro del servidor de Asterisk, y se obtuvo un porcentaje del 6.3% para el uso del CPU y un 6.0% en uso de memoria debido a que no existe una gran cantidad de llamadas; no existe un aumento significativo en estos parámetros.

PID	USER	PR	NI	UIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2892	asterisk	20	0	1681060	61076	19096	S	6.3	6.0	0:03.85	asterisk
2153	mysql	20	0	1171468	100944	8616	S	4.7	9.9	0:02.83	mysqld
2159	mongodb	20	0	478860	53912	17492	R	1.0	5.3	0:02.49	mongod
3232	asterisk	20	0	1075776	44776	14424	S	1.0	4.4	0:01.14	node /var/www/h
25	root	20	0	0	0	0	S	0.7	0.0	0:01.52	kworker/0:1
3	root	20	0	0	0	0	S	0.3	0.0	0:02.17	ksoftirqd/0
1867	redis	20	0	143060	7916	1520	S	0.3	0.8	0:00.26	redis-server
3176	asterisk	20	0	917388	34212	11684	S	0.3	3.4	0:00.66	PM2 v2.10.7: Go

Figura 33. Prueba de Uso de recursos del servidor Asterisk durante la prueba de estrés del escenario 2

En la **¡Error! La autoreferencia al marcador no es válida.** se registra la recopilación de todos los datos obtenidos con el incremento de 200 llamadas para cada prueba. Se puede observar los resultados recopilados en cada uno de estos escenarios de prueba, donde se identifica que, al llegar a 600 llamadas, el WebService comienza a experimentar problemas de comunicación y genera una alta cantidad de llamadas rechazadas, equivalentes en este caso al 55.16% del total de llamadas generadas. Además, se observa un promedio de tiempo de respuesta de 27.225 ms y una latencia de 0.017 ms."

Tabla 4. Recopilación de resultados obtenidos del escenario 2 hasta 1000 llamadas

ESCENARIO DE PRUEBA 2 – WEBSERVICE										
N° Prueba	Cantidad de Llamadas	Llamadas Correctas	% Llamadas Fallidas	Llamadas por Segundo	Tiempo de Respuesta de Llamadas (ms)	Paquete 1	Paquete 2	Latencia (ms)	CPU %	RAM %

1	200	194	97	1,509	66,2690	10,5393	10,5391	0,018	6,30%	6,00%
2	400	317	79,25	5,115	19,5503	25,4844	25,4783	0,615	7,00%	6,60%
3	600	269	44,8333	3,673	27,2257	59,5055	59,5053	0,017	6,60%	6,60%
4	800	401	50,125	2,424	41,2541	21,9122	21,90558	0,666	6,60%	6,60%
5	1000	546	54,6	2,953	33,8638	1,27797	1,275311	0,266	6,60%	6,60%

4.4. Discusión

Las pruebas de estrés evidenciaron la diferencia de rendimiento obtenidas con las pruebas de esfuerzo entre cada uno de los escenarios propuestos e implementados. En el caso del escenario con conexión directa de la base de datos, se evidencia una capacidad de 606 llamadas concurrentes sin inconvenientes puesto que de las 1000 llamadas establecidas en SIPp el 33.5% presentan fallos. En el caso del escenario con Webservice, se evidencia una capacidad de 546 llamadas concurrentes sin inconvenientes considerando que de las 1000 llamadas establecidas en SIPp el 54.6% presentan fallos. Otro de los parámetros considerados a medir en la presente investigación es el tiempo de respuesta, evidenciando en el escenario con conexión directa a la base de datos un promedio de 28.48 ms mientras que para conexión con Webservice se obtienen en promedio un 37.63 ms, lo que representa una diferencia de 9.15 ms entre ambos escenarios.

Para el caso de la latencia se evidencia para el escenario con conexión directa a la base de datos se obtiene una latencia promedio de 2.08 ms, mientras que para el escenario con Webservice se obtiene una latencia de 0.31 ms, lo que representa una diferencia mínima de 1.8 ms entre ambos escenarios. Según el trabajo realizado por Cedeño, Zambrano y Zambrano (2021), indican que para percibir una calidad de voz de excelencia se requiere poseer una latencia entre 0 a 150 ms, lo que indica que el sistema cumple con este estándar para ambos escenarios.

Finalmente considerando el uso de recursos la capacidad del CPU y memoria RAM, se han obtenido resultados muy cercanos para ambos escenarios. Para el caso del escenario con conexión directa a la base de datos se ha obtenido un promedio de 8.37% de consumo de CPU y de memoria RAM un 6.59%, mientras que para el escenario con Webservice se obtuvo un porcentaje de uso del CPU de 6.63% y de memoria RAM un 6.45%. Existiendo una diferencia de 1.74% en el uso del CPU mientras que en el uso de memoria esto se reduce al 0.04%.

CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones

- En esta investigación se pudo identificar varias opciones que permiten la integración entre Asterisk y la base de datos PostgreSQL en protocolo IPv6, determinando dos alternativas e integración para una conexión directa a la base de datos ODBC y JDBC, escogiendo ODBC por razones técnicas de compatibilidad. Por otro lado, para la integración por Webservice se determina como mejor alternativa para la integración AGI en lugar de FastAGI, ya que permite ejecutar comandos dentro del Dialplan implementado con códigos externos como PHP, Python entre otros.
- El desarrollo del prototipo en GNS3 permitió integrar un Servidor Asterisk con la base de datos PostgreSQL con protocolo IPv6, a través del cual el Softphone Zoiper con dirección IPv6 puede realizar satisfactoriamente la consulta de notas académicas, a través de una interacción con un IVR implementado en Asterisk; que habilita dos opciones; la consulta por materia o la consulta de todas las materias. Para la funcionalidad de la integración con conexión directa a la base de datos se habilita la extensión 2001 y para la funcionalidad con integración por Webservice se habilita la extensión 2002.
- El rendimiento de la consulta de notas académicas utilizando la primera opción de integración con conexión directa a la base de datos utilizando ODBC, respecto a la segunda opción de integración con Webservice utilizando AGI, referente a tiempo de respuesta es mejor con una diferencia de 9.15 ms, respecto a latencia el primer escenario es mejor con una diferencia de 1.8 ms y respecto al uso de CPU y memoria RAM existe una diferencia de 1.74% en el uso del CPU mientras que en el uso de memoria esto se reduce al 0.04%.
- Los resultados de rendimiento de las dos alternativas de integración utilizadas en esta investigación, permiten dar una visión técnica para la implementación de esta solución en otros campos de aplicación que no sean la consulta de notas, teniendo una visión clara de la concurrencia que cada una de las tecnologías puede soportar y la cantidad de usuarios finales que organizaciones o instituciones tengan y requieran.

5.2. Recomendaciones

- Se recomienda la verificación de los Drivers y Sockets ODBC dentro de Asterisk previo a tratar de comunicar con cualquier base de datos, ya que una errónea configuración de estos drivers puede provocar que no exista ningún tipo de conexión con la base de datos requerida. Así también se debe considerar que dentro del servidor base de datos se debe configurar el puerto de escucha y la dirección del servidor de Asterisk para que pueda comunicarse libremente.
- Se recomienda verificar la existencia de todos los drivers en cada uno de los servidores para que pueda haber una interacción libre entre los servicios. Dentro del servidor de PostgreSQL se requiere obtener el socket PHP-PSQL para poder ejecutar comandos con la interfaz AGI, de igual forma se lo aplica en el servidor de Asterisk.
- Se recomienda la implementación de esta investigación en otros campos distintos al educativo, con el objetivo de validar que el prototipo puede implementarse en producción en cualquier otra área o campo.

BIBLIOGRAFÍA

- Aalam, Z., Kumar, V., & Gour, S. (2021). A review paper on hypervisor and virtual machine security. *Journal of Physics*, 19(5), 1-9. doi:10.1088/1742-6596/1950/1/012027
- Aguirre, V. (2023). Diagnóstico y perspectivas de la implementación de IPv6 en el Ecuador. [Tesis de pregrado, Escuela Politécnica Nacional], Repositorio Institucional epn. Obtenido de <https://bibdigital.epn.edu.ec/handle/15000/23618>
- AJPD soft. (2020). Primer proyecto de laboratorio de red virtual con GNS3 en Windows e instalación de router Cisco 7200. Obtenido de https://proyectoa.com/primer-proyecto-de-laboratorio-de-red-virtual-con-gns3-e-instalacion-de-router-cisco-7200/#google_vignette
- Arranz, U. (2016). Propuesta de infraestructura de integración de aplicaciones basado en herramientas para la Universidad de las Ciencias Informáticas. Tesis de Grado, Repositorio UCI.
- Bjarnason, E., Lang, F., & Mjöberg, A. (2023). An empirically based model of software prototyping: a mapping study and a multi-case study. *Empirical Software Engineering*, 28(115), 1-47. Obtenido de <https://link.springer.com/article/10.1007/s10664-023-10331-w>
- Bustos, S. (2023). Desarrollo de un plan de migración de una red de área extensa de un proveedor de servicios de internet a una red definida por software SD-WAN. [Tesis de pregrado, Universidad Técnica del Norte], Repositorio Institucional utn. Obtenido de <http://repositorio.utn.edu.ec/bitstream/123456789/15328/2/04%20RED%20367%20TRABAJO%20GRADO.pdf>
- Castro, D. (2022). Control de un hogar domotizado a través de Asterisk. [Tesis de maestría, Universidad de Oviedo], Repositorio Institucional uniovi. Obtenido de <https://digibuo.uniovi.es/dspace/handle/10651/64298?locale-attribute=en>
- Cedeño, S., Zambrano, D., & Zambrano, W. (2021). Revisión sistemática de Comunicaciones Unificadas de VoIP en redes CAN. *Revista de Tecnologías de la Informática y las Telecomunicaciones*, 5(1), 17-34. doi:<https://doi.org/10.33936/isrtic.v5i1.3569>
- Chen, Y. (2020). Research and Implementation of Remote Sensing Image Database Technology Based on Oracle9i. *Journal of Physics*, 1(3), 1-5. doi:10.1088/1742-6596/1533/4/042002
- Deka, A., & Kumari, B. (2019). A DTMF based Railway Enquiry System. *Computing for Sustainable Global Development*, 1(2), 51-54. Obtenido de <https://ieeexplore.ieee.org/document/8991235>

- Fernández, Y. (2023). IPv6: qué es, para qué sirve y qué ventajas tiene . Obtenido de <https://www.xataka.com>: <https://www.xataka.com/basics/ipv6-que-sirve-que-ventajas-tiene>
- formaciongcc.com. (2020). Centralita Asterisk. Solución VoIP para call center. Obtenido de <https://www.formaciongcc.com>: <https://www.formaciongcc.com/centralita-asterisk/>
- GNS3. (2024). The software that empowers network professionals. Obtenido de <https://www.gns3.com/>
- González, G. (2022). GNS3 como herramienta para el diseño y desarrollo de prácticas de laboratorio de redes. [Tesis de pregrado, Universidad de Valladolid], Repositorio Institucional uva. Obtenido de <https://uvadoc.uva.es/bitstream/handle/10324/57310/TFG-G5812.pdf?sequence=1&isAllowed=y>
- Guerrero, J., & Guerrero, J. E. (2010). Definición de un Procedimiento de Pruebas para Definir la Capacidad, Disponibilidad y QoS de un Servidor de Asterisk. [Tesis de Grado], Repositorio ESPOL.
- IBM. (2023). ¿Qué es la integración de aplicaciones? Obtenido de <https://www.ibm.com/mx-es/topics/application-integration>
- IONOS. (2022). PostgreSQL: el gestor de bases de datos a fondo. Obtenido de <https://www.ionos.es>: <https://www.ionos.es/digitalguide/servidores/know-how/postgresql/>
- Keating, C., Katina, P., Jaradat, R., & Bradley, J. (2019). Framework For Improving Complex System Performance. *Engineering Management & Systems Engineering*, 29(1), 1218-1232. doi:<https://doi.org/10.1002/j.2334-5837.2019.00664.x>
- Khan, R., Alharbi, N., Alghamdi, G., & Berriche, L. (2022). Virtualization Software Security: Oracle VM VirtualBox. *Data Science at Prince Sultan University*, 1(4), 58-60. doi:10.1109/WiDS-PSU54548.2022.00023
- Khasawneh, B., & Alarmouty, B. (2019). Towards IPV6 Adoption in Developing Arab Countries in Western Asia Region. *International Journal of Technology and Engineering Studies*, 5(2), 55-61. Obtenido de https://www.researchgate.net/publication/339173295_Towards_IPV6_Adoption_in_Developing_Arab_Countries_in_Western_Asia_Region
- Kurniawan, A., & Warlina, L. (2020). Web Service for Academic Information Systems. *Engineering, Science, and Technology*, 87(9), 1-8. doi:10.1088/1757-899X/879/1/012009
- Madsen, D. (2018). Asterisk: The Definitive Guide 5th Guide. O'Reilly Media.

- Minda, C., & Pacheco, R. (2019). Aplicación de Balanceo De Carga Dinámico Para Servidores, Basada En Redes Definidas Por Software. *Revista Ibérica de Sistemas e Tecnologías de Informação*, 67-81.
- Momtaz, F., & Noor, R. (2015). Asterisk: An Open Source Framework for Building Communications Applications. *ICIS 2015*.
- Ndiaye, L., Gueye, K., Degboe, B., & Ouya, S. (2020). Proposal of an IVR Solution and Granting of Credit With Asterisk's AGI and a Flask RESTful Framework in an IMS Network: Case of an Advertisement. *Advanced Communication Technology*, 1(1), 1-12. doi:10.23919/ICACT48636.2020.9061249
- NIC México. (2024). Network Information Center México S.C. Obtenido de Fundamentos de IPv6: <https://www.ipv6.mx/index.php/informacion/fundamentos/ipv6>
- Peña, J. (2011). ESTUDIO DEL COMPORTAMIENTO DEL PROTOCOLO DTLS PARA VOIP, BASADA EN SEÑALIZACIÓN SIP SOBRE UDP, CONSIDERANDO EL TIEMPO DE RESPUESTA COMO VARIABLE DE INTERÉS. [Tesis de Grado], UCV.
- Pérez-Hernández, J., Vázquez-Blázquez, M., & García-Macías, J. (2023). Asterisk: A Powerful Open-Source VoIP PBX.
- PostgreSQL. (2024). About PostgreSQL. Obtenido de <https://www.postgresql.org/about/>
- Singh, J. (2023). Asterisk es una sucursal privada (Pbx) para VoIP. Obtenido de [https://www.idtexpress.com: https://www.idtexpress.com/es/blog/asterisk-private-branch-exchange-pbx-voip/](https://www.idtexpress.com:https://www.idtexpress.com/es/blog/asterisk-private-branch-exchange-pbx-voip/)
- SIPp. (2014). Documentación de referencia SIPp. Obtenido de <https://sipp.sourceforge.net/doc/reference.html#:~:text=SIPp%20is%20a%20performance%20testing,describing%20any%20performance%20testing%20configuration.>
- Sivankalai, S., & Virumandi, A. (2021). Web Services in Cloud Computing r vices in Cloud Computing research: Insights fr ch: Insights from Scientometric. *Digital Commons*. Obtenido de <https://digitalcommons.unl.edu/cgi/viewcontent.cgi?article=10865&context=libphilprac>
- Tanutama, L., Halim, B., & Kanggara, A. (2021). Voice Quality Assessment of SIP-PBX Softphone Extension in 3G Cellular Service Environment. *Earth and Environmental Science*, 79(4), 1-9. doi:10.1088/1755-1315/794/1/012130
- Tolliver, K., Fields, J., Stepler, R., & Williams, A. (2023). Case Prioritization in the SIPP: A five year review. *Survey Practice*, 16(1), 1-10. doi:<https://doi.org/10.29115/SP-2023-0010>

- Universidad de Lleida. (2024). ¿Qué es un prototipo? Obtenido de <https://mpiua.invid.udl.cat/fases-mpiua/prototipado/que-es-un-prototipo/>
- Voznak, M., & Rozhon, J. (2010). Methodology for SIP Infrastructure Performance Testing. *WSEAS TRANSACTIONS on COMPUTERS*, 9(9).
- Zhang, L., Li, X., & Zhang, Y. (2020). IPv6 Deployment in Higher Education: A Survey.

ANEXO

Generación de Red IPv6

Para configurar la red IPv6 del prototipo, se utiliza la topología descrita en la Figura 34. Esta estructura de red consta de cuatro Routers, cuatro switches y dos servidores destinados al alojamiento de las máquinas y servicios tanto de PostgreSQL como de Asterisk. Además, se cuenta con una máquina cliente desde la cual se realizarán las llamadas hacia los servidores, en donde, cada uno de estos, así como el cliente se encuentran en una subred diferente, también cuenta con un cliente donde está instalado la máquina para realizar las pruebas de rendimiento del prototipo. En la primera subred, donde se ubica el cliente del Softphone, se asigna la red 2003:db8:a::/64 a través del Router 1 con la interfaz FastEthernet0/0. A partir del Router 2, la red 2003:db8:b::/64 se utiliza para el servidor de VoIp en Asterisk, mientras que desde el Router 3, la tercera subred del sistema se configura con la red 2003:db8:c::/64 para la máquina de PostgreSQL, también a través de la interfaz FastEthernet0/0 y por último en el Router 4, la cuarta subred del sistema se configura con la red 2003:db8:d::/64 con la interfaz FastEthernet0/0 .

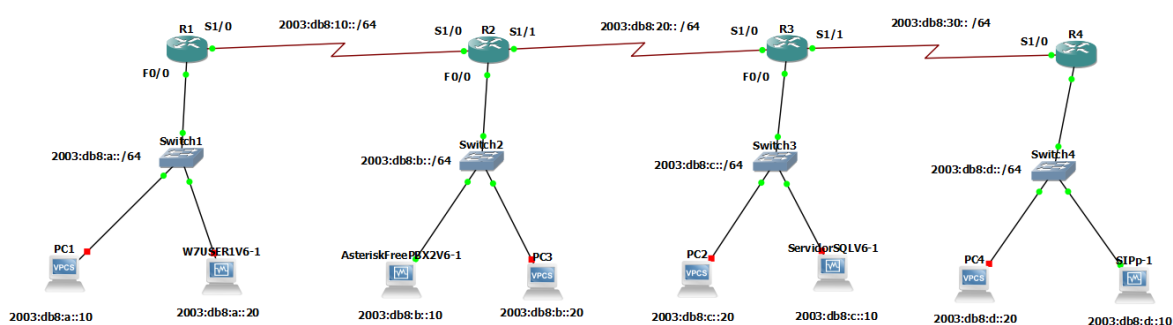


Figura 34. Topología de red IPv6 del prototipo

El router seleccionado tanto para R1, R2, R3 y R4 es en el Cisco C3725 que posee una memoria RAM de 128 megas Tiene 2 interfaces Fast Ethernet F0/0 y la F0/1 así como una tarjeta NM-4T con cuatro puertos Seriales S1/0, S1/1 S1/2 y S1/3. La comunicación entre el router R1 y R2 se la realiza mediante la interfaz S1/0 para R1 y su contraparte en R2 es decir la interfaz S1/0. Esta redes se comunican mediante una nueva subred correspondiente a las IPv6 2003:db8:10::/64. La comunicación entre R2 y R3 se la realiza por medio de las interfaces S1/1 de R2 y S1/0 de R3 utilizando la subred 2003:db8:20::/64. Mientras que las interfaces para R3 y R4 es S1/1 y S1/0 respectivamente, utilizando la subred 2003:db8:30::/64.

En la Figura 35 se muestra las características técnicas de los router utilizados para la implementación del prototipo. Una vez identificadas las redes y sus redes dedicadas a la comunicación entre routers se procede a la configuración de cada uno de estos, en dónde se asigna como puerta de enlace la IP uno de cada una de las subredes mencionadas anteriormente (2003:db8:a::1), así como la primera dirección IP válida para la comunicación entre cada uno de los routers. Una vez establecidas las direcciones de cada uno de los puertos de salida, se procede a trazar las rutas respectivas mediante el comando ROUTE IPv6 de la

consola de comandos del dispositivo y así se obtiene logra obtener la comunicación entre todos los dispositivos.

```
Router R1 is started
Running on server LENOVO-GUS with port 3080
Local ID is 7 and server ID is 65d08be3-d52e-4f40-9e9a-53f1c3fc6d2b
Dynamips ID is 1
Hardware is Dynamips emulated Cisco c3725 with 128MB RAM and 256KB NVRAM
Console is on port 5016 and type is telnet, AUX console is on port None
IOS image is "c3725-adventerprisek9-mz.124-25d.image"
with idlepc value of 0x602467a4, idlemax of 500 and idlesleep of 30 ms
PCMCIA disks: disk0 is 0MB and disk1 is 0MB
slot 0 hardware is GT96100-FE with 2 ports
FastEthernet0/0 connected to Switch1 on port Ethernet0
FastEthernet0/1 is empty
slot 1 hardware is NM-4T with 4 ports
Serial1/0 connected to R2 on port Serial1/0
Serial1/1 is empty
Serial1/2 is empty
Serial1/3 is empty
WIC-1T installed in WIC slot 0 with 1 port
Serial0/0 is empty
```

Figura 35. Características de la imagen del router cisco c3725

Una vez implementadas las rutas entre cada una de las redes que componen el sistema hoy se procede a realizar la configuración de las IPs en cada uno de los usuarios que se conectan a los servicios para hacer las pruebas respectivas del correcto funcionamiento de la red se implementan peces virtuales a los cuales se les va a otorgar una dirección IPv6 de cada una de las subredes en el PC1 que es parte de la red del router uno se le asigna la dirección IPv6 2003:db8:a::10/64, para PC2 se asigna 2003:db8:b::20/64 y finalmente para PC3 hoy se asigna la IP 2003:db8:c::20/64. En la Figura 36 se puede observar la configuración de la red correspondiente al PC1.



```
Welcome to Virtual PC Simulator, version 0.6.2
Dedicated to Daling.
Build time: Apr 10 2019 02:42:20
Copyright (c) 2007-2014, Paul Meng (mirnshi@gmail.com)
All rights reserved.

VPCS is free software, distributed under the terms of the "BSD" licence.
Source code and license can be found at vpcs.sf.net.
For more information, please visit wiki.freecode.com.cn.

Press '?' to get help.

Executing the startup file

Checking for duplicate address...
PC1 : 192.168.2.10 255.255.255.0 gateway 192.168.2.1

PC1 : 2003:db8:a::10/64
PC1> █
```

Figura 36. Configuración ipv6 red de pc1

Una vez configuradas las direcciones IP en cada uno de los equipos virtuales PC1, PC2 y PC3 se procede a comprobar el funcionamiento y comunicación de la red para lo cual se realiza un ping desde el PC1 hacia la dirección IP 2003:db8:b::20 correspondiente a PC2 tal

y como se observa en la Figura 37, en donde se muestra un tiempo de respuesta de 40ms aproximadamente.

```
PC1> ping 2003:db8:b::20
2003:db8:b::20 icmp6_seq=1 ttl=60 time=58.606 ms
2003:db8:b::20 icmp6_seq=2 ttl=60 time=42.790 ms
2003:db8:b::20 icmp6_seq=3 ttl=60 time=40.752 ms
2003:db8:b::20 icmp6_seq=4 ttl=60 time=40.291 ms
2003:db8:b::20 icmp6_seq=5 ttl=60 time=40.799 ms
```

Figura 37. Respuesta ping pc1 a pc2

Para comprobar el acceso hacia la red del PC3 se realiza el mismo procedimiento anterior, haciendo ping desde PC1 hacia la dirección IP 2003:db8:c::20 tal y como se observa en la Figura 38, obteniendo un tiempo de respuesta de 62ms, comprobando así el correcto funcionamiento y enrutamiento del sistema.

```
PC1> ping 2003:db8:c::20
2003:db8:c::20 icmp6_seq=1 ttl=58 time=71.406 ms
2003:db8:c::20 icmp6_seq=2 ttl=58 time=64.427 ms
2003:db8:c::20 icmp6_seq=3 ttl=58 time=62.598 ms
2003:db8:c::20 icmp6_seq=4 ttl=58 time=63.889 ms
2003:db8:c::20 icmp6_seq=5 ttl=58 time=61.516 ms
```

Figura 38. Respuesta ping pc1 a pc3

Una vez obtenida una correcta respuesta en las diferentes redes por medio de los PC's virtuales, se procede a la configuración de cada una de las máquinas virtuales correspondientes al Cliente y los Servidores de Asterisk y PostgreSQL. Para la configuración de red en máquina del usuario Softphone, Se ingresa al centro de redes y recursos compartidos de Windows 7, se selecciona la interfaz de red del ordenador virtual, se busca la opción IPV6 y se da Clic en propiedades, donde aparecerán los cuadros de ingreso para colocar de forma manual la dirección IP tal y como se observa en la Figura 39, la cual está asignada con la dirección 2003:db8:a::20 con un prefijo de subred 64 y puerta de enlace 2003:db8:a::1.

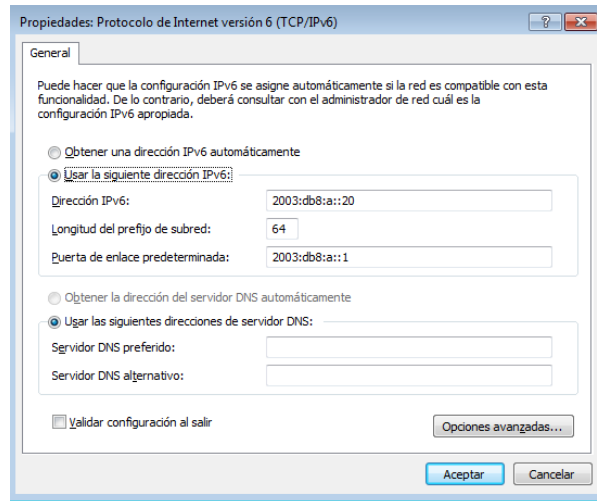


Figura 39. Configuración IPv6 windows 7 – softphone

Para la configuración de red mediante el protocolo IPv6 dentro del servidor de Asterisk basado en Linux, se requiere ingresar con permisos de administrador y utilizando el editor de texto “nano” al archivo ifcfg-eth0, ubicado en la ruta /etc/sysconfig/network-scripts/ifcfg-eth0. Al abrir este archivo de configuración, tal y como se puede observar en la Figura 40, se muestran las opciones de configuración de la interfaz de red ETH0, en dónde se debe colocar para IPV6ADDR la dirección IP asignada a este servidor, la cual es 2003:db8:b::10/64 y en IPV6_DEFAULTGW se debe colocar la puerta de enlace correspondiente a esta red la cual corresponde a 2003:db8:b::1.

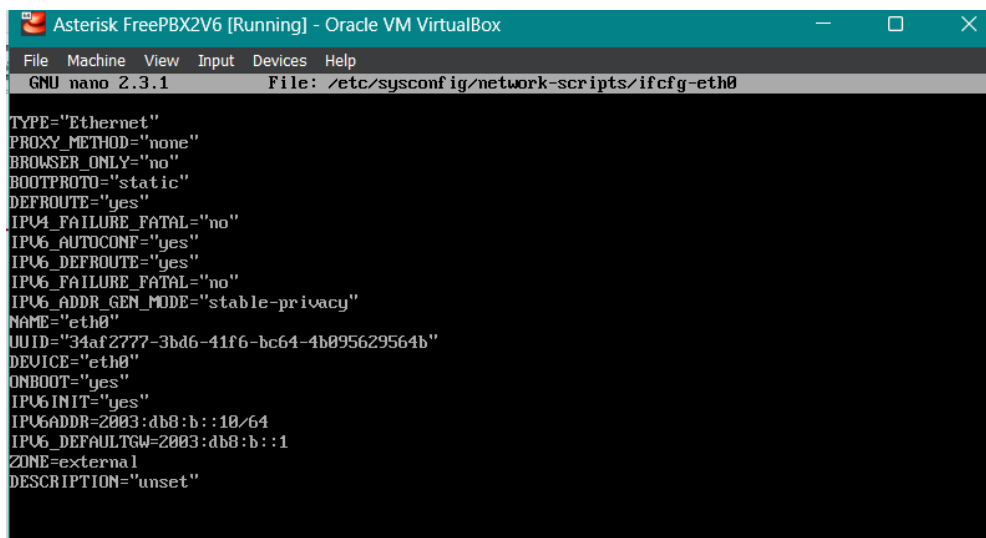


Figura 40. Configuración IPv6 Asterisk - servidor VoIP

Finalmente para la configuración del protocolo IPv6 en la máquina de Ubuntu 18.04 que contiene el servidor de PostgreSQL, se ingresa a la configuración del adaptador de red, se selecciona el ingreso manual de la dirección IP y se ingresan los valores correspondientes a las direcciones asignadas para este equipo, mismas que corresponden a 2003:db8:c::10, con prefijo de red 64 y puerta de enlace 2003:db8:c::1, tal y como se observa en la Figura 41.



Figura 41. Configuración IPv6 Ubuntu - servidor PSQL

Implementación de Servidor PostgreSQL

Para la implementación del Servidor PostgreSQL se seleccionó una máquina virtual basada en el sistema operativo Ubuntu 18.04 de 64 bits, este equipo virtualizado consta de las especificaciones que se observan en la Figura 42, misma que arranca con 2 procesadores, los cuales están ligados a una memoria base de 2048MB, una memoria de vídeo de 16MB y un adaptador de red el cual está configurado como “no conectado” para que pueda comunicarse mediante la interfaz de GNS3.

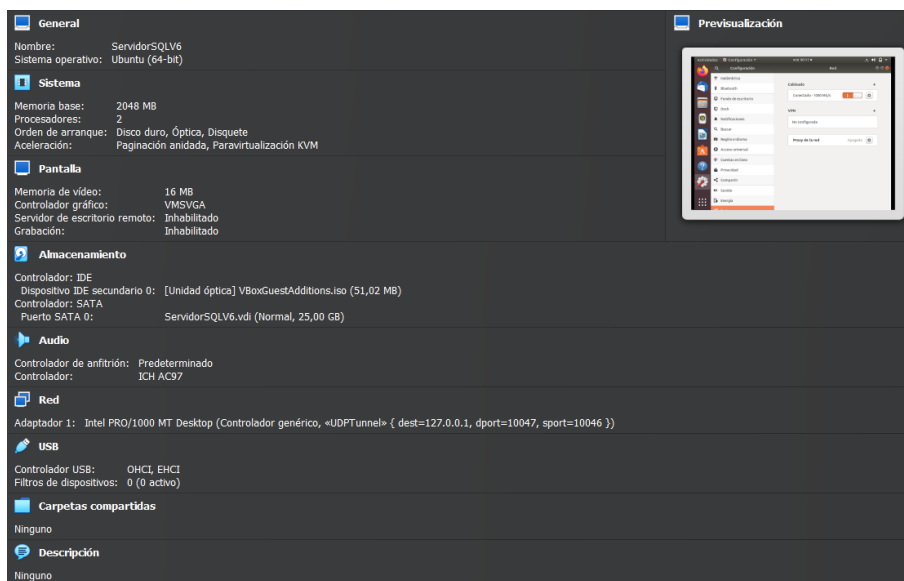


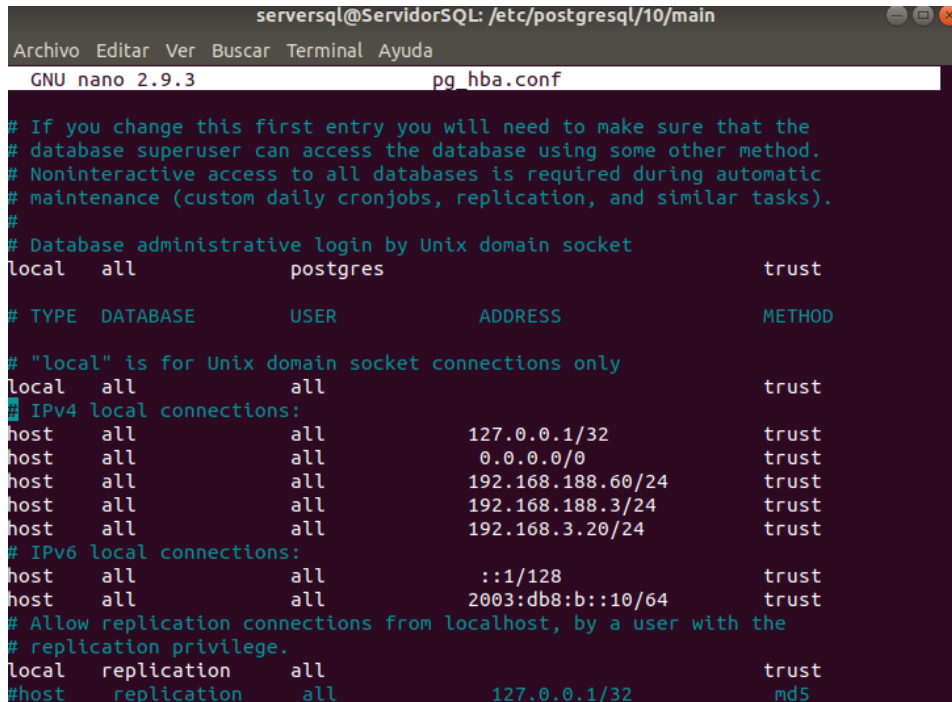
Figura 42. Configuración de máquina virtual Ubuntu – PostgreSQL

Establecida la máquina virtual y configurada su dirección IPv6, se procede a la instalación del servidor postre SQL para lo cual se debe ingresar a la ventana de comandos o terminal de Ubuntu e ingresar el comando *sudo apt install postgresql postgresql-contrib*, con el cual se iniciará la instalación del programa para el servidor de base de datos.

Finalizada la instalación se requiere realizar unas modificaciones a los archivos de configuración de PostgreSQL para lo cual, y tal como se observa en la Figura 43, se ingresa

a la carpeta `/etc/postgresql/10/main` y mediante el editor de texto `nano` se ingresa al archivo `pg.hba.conf`, mismo que contiene las direcciones que pueden ingresar hacia este servidor.

Se agrega una nueva fila y se colocan los parámetros que se observan en la Figura 43 colocando la dirección IPv6 del Servidor de Asterisk la cual corresponde a `2003:db8:b::10/64` y se coloca un método de identificación `trust`, considerando que deben ser guardados y verificados los cambios efectuados dentro del archivo.



```
serversql@ServidorSQL: /etc/postgresql/10/main
GNU nano 2.9.3 pg_hba.conf
# If you change this first entry you will need to make sure that the
# database superuser can access the database using some other method.
# Noninteractive access to all databases is required during automatic
# maintenance (custom daily cronjobs, replication, and similar tasks).
#
# Database administrative login by Unix domain socket
local all postgres trust
# TYPE DATABASE USER ADDRESS METHOD
# "local" is for Unix domain socket connections only
local all all trust
# IPv4 local connections:
host all all 127.0.0.1/32 trust
host all all 0.0.0.0/0 trust
host all all 192.168.188.60/24 trust
host all all 192.168.188.3/24 trust
host all all 192.168.3.20/24 trust
# IPv6 local connections:
host all all ::1/128 trust
host all all 2003:db8:b::10/64 trust
# Allow replication connections from localhost, by a user with the
# replication privilege.
local replication all trust
#host replication all 127.0.0.1/32 md5
```

Figura 43. Configuración de redes de escucha en PostgreSQL

Otro de los archivos que debe ser considerado modificar para poder establecer una conexión con los servicios que otorga este servidor se encuentra dentro de la carpeta `/etc/postgresql/10/main` el cual debe ser abierto con permisos de administrador y mediante el uso del editor de texto `nano` el cual es `postgresql.conf`.

Dentro de este archivo se encuentran las configuraciones básicas del servidor PSQL, para este caso se debe colocar en la opción `listen_addresses` la dirección IP correspondiente al servidor la cual corresponde a `2003:db8:c::10`, así también se debe verificar que el puerto al cual se conecta es el 5432 y que las conexiones Unix deben tener permiso 777 tal y como se observa en la Figura 44.

```

serversql@ServidorSQL: /etc/postgresql/10/main
Archivo Editar Ver Buscar Terminal Ayuda
GNU nano 2.9.3 postgresql.conf
#-----
# CONNECTIONS AND AUTHENTICATION
#-----
# - Connection Settings -

listen_addresses = '2003:db8:c::10'          # what IP address(es) $
                                             # comma-separated list of addresses;
                                             # defaults to 'localhost'; use '*' for$
                                             # (change requires restart)
port = 5432                                  # (change requires restart)
max_connections = 100                        # (change requires restart)
#superuser_reserved_connections = 3         # (change requires restart)
#unix_socket_directories = '/var/run/postgresql' # comma-separated list$
                                             # (change requires restart)
#unix_socket_group = ''                     # (change requires restart)
unix_socket_permissions = 0777              # begin with 0 to use octal notation
                                             # (change requires restart)
#bonjour = off                              # advertise server via Bonjour
                                             # (change requires restart)
#bonjour_name = ''                          # defaults to the computer name
                                             # (change requires restart)

# - Security and Authentication -

```

Figura 44. Configuración de dirección de servidor PostgreSQL

Una vez realizadas todas las configuraciones del servidor se procede a crear la base de datos que van a contener la información de los alumnos y que será extraída por el servidor de Asterisk. Como primer paso, se crea una base de datos llamada “ALUMNOS” la cual va a estar conectada a un usuario denominado Sistema_Academico mediante el puerto 5432. Al ingresar mediante la ventana de comandos del terminal de Ubuntu se puede colocar el comando `\conninfo`, el cual permitirá observar el socket conectado a la base de datos tal y como se observa en la Figura 45.

```

ALUMNOS=# \conninfo
You are connected to database "ALUMNOS" as user "ALUMNOS" via socket in "/var/r
un/postgresql" at port "5432".

```

Figura 45. Verificación de base de datos " Sistema_Academico " en PSQL

Cuando se ha verificado la conexión con la base de datos se procede a crear las tablas correspondientes que contendrán la información de cada uno de los alumnos. Como primer punto se ha creado la tabla denominada “Notas_Estudiantes” la cual consta de 6 columnas las cuales son:

- **id:** valor que será auto incremental para identificar el orden de con he tenido de la información.
- **nombre:** en donde estará el nombre del estudiante.
- **apellido:** en donde estará el apellido del estudiante.
- **ci:** que contendrá la información de la cédula de cada uno de los estudiantes y el cuál será el principal gestor para la búsqueda de información.
- **nota1, nota2, nota3, nota4, nota5** contiene la información de las 5 calificaciones creadas para este prototipo.

En la Figura 46 se puede observar la tabla notas uno de la base de datos de alumnos la cual contiene actualmente n filas las cuales corresponden a n diferentes alumnos con sus identificaciones correspondientes y las notas de cada asignatura.

id	nombre	apellido	ci	nota1	nota2	nota3	nota4	nota5
4	Carlos	Morales	1275896822	18	20	10	15	16
5	Maria	Almada	1720587415	14	15	18	18	20
6	Jose	Ramirez	1234567891	14	17	18	19	13
7	Pedro	Salcan	0602731911	19	20	14	17	16
8	Cristian	Salcan	0604938514	12	19	19	20	20
9	Ricardo	Pinto	1234567892	19	19	20	20	17

Figura 46. Tabla Notas_Estudiantes de la base de datos Sistema_Academico PSQL

Para la identificación de las asignaturas en el sistema se utilizó una segunda base de datos la misma que contiene cuatro columnas, la primera es el *id* que corresponde al orden en el cual se va ingresando las materias dentro de la tabla. La segunda columna *código* contiene el número o identificador de asignatura el cual son cuatro números que permitirán diferenciar al sistema la materia a la cual se encuentra intentando hacer la búsqueda. La tercera columna denominada *materia* contiene el identificador que nos permitirá enlazar esta información con la base de datos **Notas_Estudiantes**, es decir dónde se encuentra la información de cada uno de los alumnos. Finalmente, la última columna denominada *asignatura* contiene el nombre específico de cada una de las materias para poder ser mencionada dentro del sistema. En la Figura 47 se puede observar la tabla materias en donde constan las 2 asignaturas creadas para las pruebas del presente prototipo.

id	codigo	materia	asignatura
1	1028	NOTA1	REDES Y ESCALABILIDAD
2	1030	NOTA2	APLICACIONES WEB
3	1032	NOTA3	BASES DE DATOS AVANZADA
4	1034	NOTA4	INTELIGENCIA ARTIFICIAL
5	1036	NOTA5	VIRTUALIZACION

Figura 47. Tabla Materias de base de datos Sistema_Academico

Implementación de Servidor Asterisk

Una vez implementada la base de datos del servidor PSQL se procede a la implementación del servidor de Asterisk que contendrá el servicio de telefonía VoIP, para ellos, se utiliza una máquina virtual denominada Asterisk, la misma que contiene todos los archivos necesarios para montar un servidor de voz. Para este servicio se configuró los parámetros que se pueden observar en la Figura 48, misma que es una máquina virtual de 64 bits con una memoria base de 1024 MB, 16 MB de memoria de vídeo y un adaptador de red que debe estar en configuración de *no conectado*.

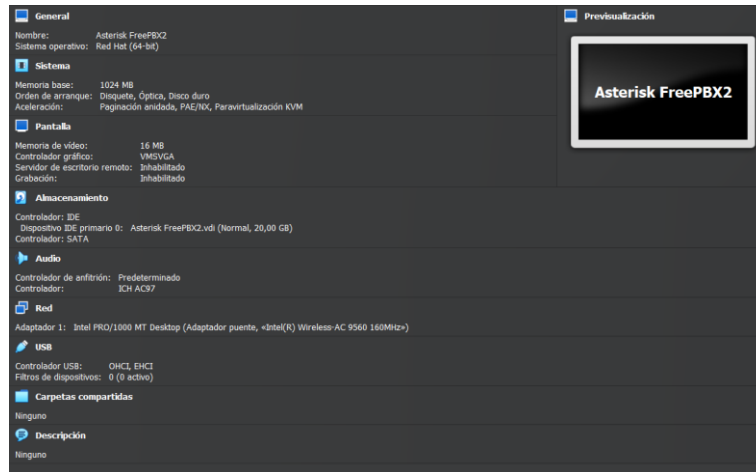


Figura 48. Configuración máquina virtual servidor Asterisk

Una vez implementada la máquina virtual e inicializada se procederá a mostrar la pantalla de carga que se observa en la Figura 49, en dónde se puede verificar las notificaciones del sistema tomando en cuenta que debe mostrar la dirección IP asignada a este servicio que para este caso es 2003:db8:b::10 en la interfaz **ETH0**.

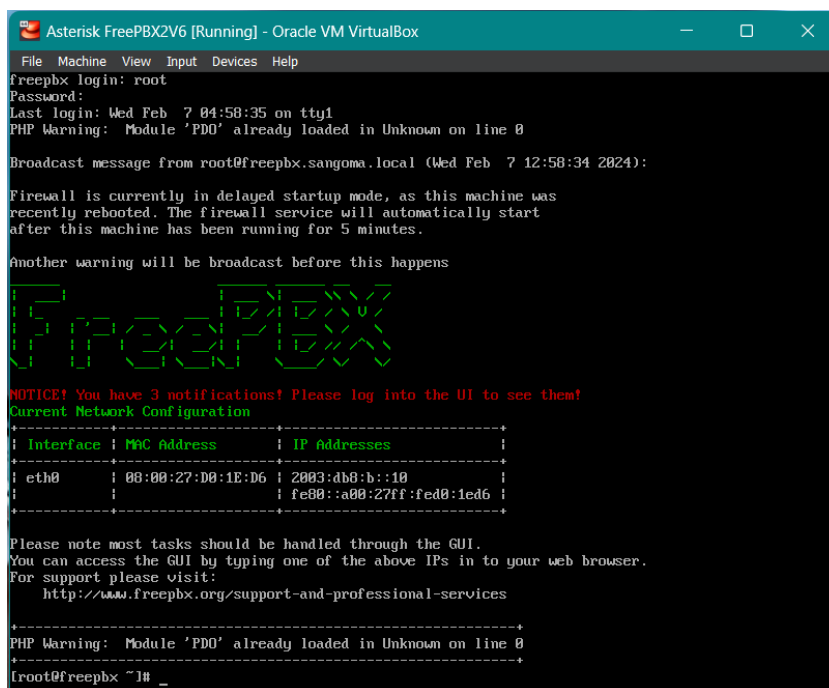


Figura 49. Verificación de conexión de red y configuración de protocolo IPv6

Una vez configurada la dirección IP del servicio y comprobado la conexión a la red se procede a realizar las configuraciones de los archivos necesarios para poder implementar el servicio mediante protocolo IPv6. Para esto se debe buscar los archivos raíz de Asterisk que se encuentran ubicados en la dirección */etc/asterisk/*. Dentro de esta carpeta se encuentra el archivo denominado *manager.conf* el cual contiene información sobre el servidor cómo el puerto la dirección y ciertas librerías necesarias para su funcionamiento tal y como se observa en la Figura 50. Se requiere configurar la línea *bindaddr* colocando “:::” para que el servicio pueda establecer conexión con cualquier dirección IPv6 de la red.

```
File Machine View Input Devices Help
GNU nano 2.3.1 File: manager.conf

AMI - Asterisk Manager interface - Generated at 2023-12-28T00:14:11+00:00

FreePBX needs this to be enabled. Note that if you enable it on a different IP, you need
to assure that this can't be reached from un-authorized hosts with the ACL settings (permit/deny).
Also, remember to configure non-default port or IP-addresses in amportal.conf.

The AMI connection is used both by the portal and the operator's panel in FreePBX.

FreePBX assumes an AMI connection to localhost:5038 by default.

[general]
enabled = yes
port = 5038
bindaddr = ::
displayconnects=no ;only effects 1.6+

[admin]
secret = a0k/vhzeZf1t
deny=0.0.0.0/0.0.0.0
permit=127.0.0.1/255.255.255.0
read = system,call,log,verbose,command,agent,user,config,command,dtmf,reporting cdr,dialplan,origi$
write = system,call,log,verbose,command,agent,user,config,command,dtmf,reporting cdr,dialplan,origi$
write timeout = 5000

#include manager_additional.conf
#include manager_custom.conf
```

Figura 50. Configuración de archivo manager.conf

Dentro de la misma ubicación que el archivo anterior es decir */etc/Asterisk/* se encuentra el archivo *sip_general_additional.conf*. El cual contiene información propia del servicio sin embargo se deben modificar dos parámetros necesarios para poder acceder al servidor, en la línea *tlsbindaddr* se debe colocar “[::]:5161” para que se puedan conectar las direcciones IPv6 por ese puerto. Otra línea que debe ser editada dentro de este archivo es *localnet* en dónde se debe colocar *[2003:db8:b::0]/64* tal y como se observa en la Figura 51, con el objetivo de que todas las direcciones de esta red puedan conectarse a este servidor principal.

```
GNU nano 2.3.1 File: sip_general_additional.conf

bindport=5160
jbenable=no
checkmwi=10
maxexpiry=3600
minexpiry=60
srlookup=no
allowguest=yes
notifyhold=yes
rtptimeout=30
canreinvite=no
rtptimeout=0
videosupport=no
defaultexpiry=120
notifying=yes
maxcallbitrate=384
rtpholdtimeout=300
g726nonstandard=no
registertimeout=20
registerattempts=0
nat=force_rport,comedia
ALLOW_SIP_ANON=no
tlsbindaddr=[::]:5161
tlscafile=/etc/pki/tls/certs/ca-bundle.crt
localnet=192.168.188.0/24
localnet=192.168.3.0/24
localnet=[2003:db8:b::0]/64
accept_outofcall_message=yes
auth_message_requests=no
outofcall_message_context=dpma_message_context
```

Figura 51. Configuración de archivo sip_general_additional.conf

Establecida la conexión por protocolo IPv6 del servidor, se requiere la instalación de ciertos componentes funcionalidades para poder implementar el servicio. Para poder ejecutar

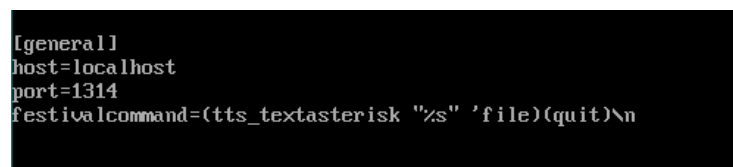
instrucciones de lectura de base de datos y que estas puedan ser entendidas por el usuario que está consumiendo el servicio o realizando la llamada se requiere de una aplicación que permita que el texto que se está generando en tiempo real se convierta en sonido es decir un sistema *TEXT TO SPEECH (TTS)*. Existen diversos tipos de aplicaciones que permiten efectuar este requerimiento, siendo la más utilizada dentro de Asterisk el servidor denominado Festival. En la Figura 52 se encuentra el comando respectivo para poder realizar la instalación del aplicativo o servidor dentro del servicio de Asterisk.



```
sudo nano /etc/sysconfig/asterisk
sudo yum install festival festival-devel
```

Figura 52. Comando para instalación de TTS Festival

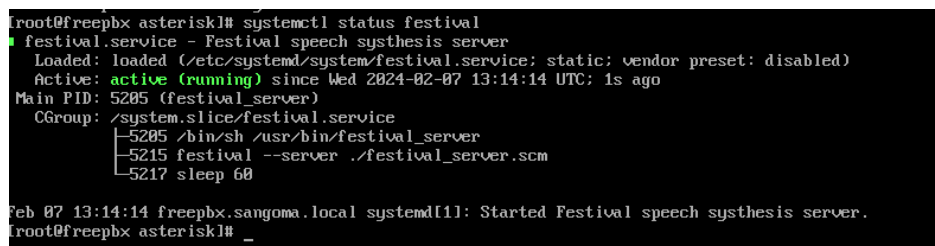
Una vez finalizada la instalación del servicio TTS festival se crearán una serie de archivos dentro de la carpeta raíz de Asterisk ubicada en la dirección */etc/Asterisk/*. Entre estos archivos se encuentra *festival.conf*, que contiene la información observada en la Figura 54. En este archivo se debe verificar la configuración del comando y el host al cual está direccionado el servicio.



```
[general]
host=localhost
port=1314
festivalcommand=(tts_textasterisk "%s" 'file')(quit)\n
```

Figura 53. Configuración básica de servidor Festival

Cuando se haya ejecutado la verificación del archivo mencionado anteriormente se utiliza el comando *systemctl start festival*, para iniciar el servicio del TTS. Sí se requiere comprobar que el sistema se encuentra en funcionamiento o el servidor esté activo y en ejecución se utiliza el comando *systemctl status festival*, tal y como se observa en la Figura 54.



```
root@freepbx asterisk1# systemctl status festival
festival.service - Festival speech synthesis server
Loaded: loaded (/etc/systemd/system/festival.service; static; vendor preset: disabled)
Active: active (running) since Wed 2024-02-07 13:14:14 UTC; 1s ago
Main PID: 5205 (festival_server)
CGroup: /system.slice/festival.service
├─5205 /bin/sh /usr/bin/festival_server
├─5215 festival --server ./festival_server.scm
└─5217 sleep 60

Feb 07 13:14:14 freepbx.sangoma.local systemd[1]: Started Festival speech synthesis server.
root@freepbx asterisk1#
```

Figura 54. Status de servidor Festival activo

Culminada la instalación configuración y comprobación del funcionamiento del TTS Festival se procede a la creación de las líneas o extensiones PJSIP que se conectarán mediante los Softphones para poder ejecutar la llamada y establecer la conexión entre este dispositivo y el servidor de Asterisk junto con todos sus complementos. Para esto se requiere abrir dentro del archivo raíz de Asterisk con el uso del editor de texto *nano* el archivo *pjsip.endpoint.conf* el cual contiene la información respectiva de cada una de las extensiones que pueden comunicarse con el servidor. Para el presente prototipo se utilizó la extensión 1888 colocando la configuración que viene por defecto tal y como se observa en la Figura 55.

```
GNU nano 2.3.1 File: pjsip.endpoint.conf
;-----;
; Do NOT edit this file as it is auto-generated by FreePBX.
;-----;
; For information on adding additional paramaters to this file, please visit
; FreePBX.org wiki page, or ask on IRC. This file was created by the new F
; BMO - Big Module Object. Any similarity in naming with BMO from Adventure
; is totally deliberate.
;-----;
#include pjsip.endpoint_custom.conf

[1888]
type=endpoint
aors=1888
auth=1888-auth
tos_audio=ef
tos_video=af41
cos_audio=5
cos_video=4
allow=ulaw,alaw,gsm,g726,g722
context=ivr_academico
callerid=Usuario1 <1888>
dtmf_mode=auto
aggregate_mwi=yes
use_avpf=no
rtcp_mux=no
ice_support=no
media_use_received_transport=no
trust_id_inbound=yes
user_eq_phone=no
send_connected_line=yes
media_encryption=no
timers=yes
```

Figura 55. Configuración de extensión PJSIP 1888

Otro de los archivos que se deben configurar para establecer una correcta conexión con las extensiones que pueden ingresar dentro del sistema es del archivo *pjsip.auth.conf*, el cual contiene el nombre de usuario y la contraseña de cada una de las extensiones configuradas para comunicarse y que puedan realizar llamadas. En la Figura 56 se puede observar la configuración para las extensiones 1888 donde se establece la contraseña y usuario que posteriormente serán ingresados y validados por medio del Softphone.

```
GNU nano 2.3.1 File: pjsip.auth.conf
;-----;
; Do NOT edit this file as it is auto-generated by FreePBX.
;-----;
; For information on adding additional paramaters to this file, please visit the
; FreePBX.org wiki page, or ask on IRC. This file was created by the new FreePBX
; BMO - Big Module Object. Any similarity in naming with BMO from Adventure Time
; is totally deliberate.
;-----;
#include pjsip.auth_custom.conf

[1888-auth]
type=auth
auth_type=userpass
password=12345
username=1888
```

Figura 56. Parámetros de ingreso para autorización de enlace PJSIP

Implementación de Cliente Softphone

Establecidos los servicios tanto de Asterisk para la ejecución y procesamiento de la llamada, como de PostgreSQL para el almacenamiento de los datos se requiere de una máquina virtual adicional conectada a la red del router R1 que permita la ejecución del Softphone que servirá como usuario de los servicios de consulta de notas.

Para esto se seleccionó el sistema operativo Windows 7 de 64 bits mismo que tiene una memoria base de 512 MB Con un procesador, una memoria de video de 27 MB como se observa en la Figura 57 y una configuración del adaptador de red en no conectado para poder realizar la conexión virtual mediante GNS3.

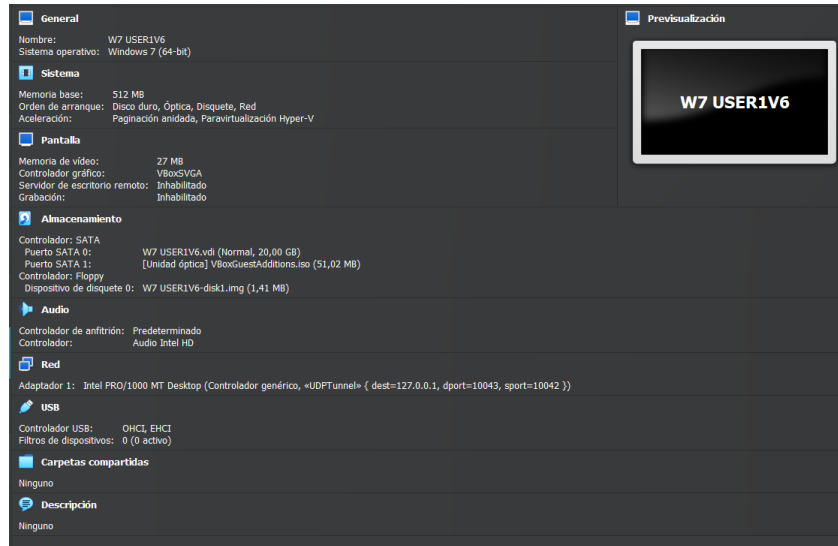


Figura 57. Configuración de máquina virtual softphone - windows 7

Una vez generada la máquina virtual con el sistema operativo Windows 7 se procede a la instalación del software denominado Zoiper5. Este es un software libre que permite comunicarse a un servidor VoIP realizar llamadas de manera gratuita o sea un servicio premium, sin embargo, se utilizará la versión gratuita para el presente prototipo. Para la configuración de este servicio se requiere colocar el dominio del servidor el cual está presentado por la dirección IP [2003:db8: b::10]. Como segundo parámetro para establecer el servicio y la comunicación se requiere colocar el usuario creado dentro de Asterisk es decir el número de extensión, para el caso de la presente máquina el usuario es 1888. También se debe colocar el *password* configurado dentro de Asterisk para la autorización del servicio que para este caso es 12345. Una vez configurados de estos parámetros dentro de SIP Credentials tal y como se observa en la Figura 59, se da clic al botón *Register* y se espera que se establezca la conexión con el servidor.

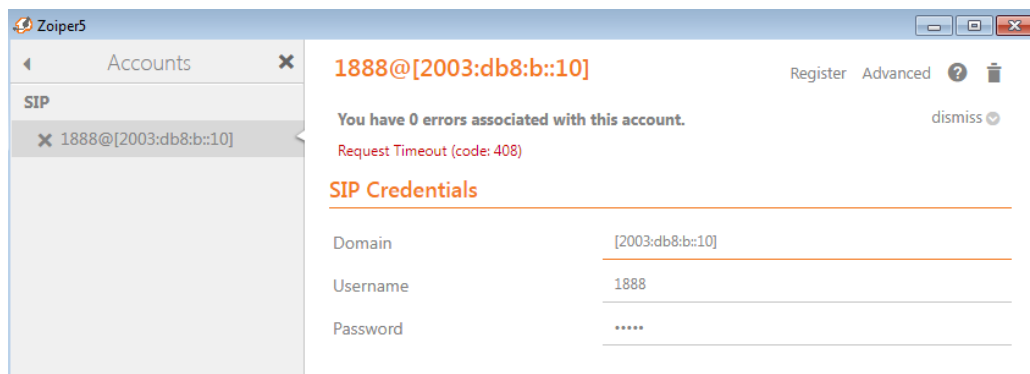


Figura 58. Configuración de extensión y dirección del dominio del servicio

Lograda establecer la conexión con el servidor VOIP, se procede a regresar a la interfaz donde se pueden digitar los números de extensión para ejecutar una llamada tal y como se puede observar en la interfaz presentada en la Figura 59, dónde se muestra el estado de conexión y los números para digitar la extensión de la llamada y la dirección IPv6 del dominio del servidor.

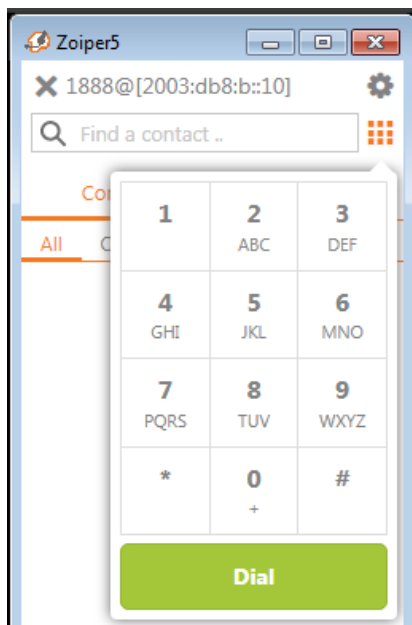


Figura 59. Interfaz de Llamada Zoiper

Una vez configurada la comunicación dentro del software Zoiper se procede a comprobar la conexión dentro del entorno de Asterisk, para lo cual se ejecuta el comando *Asterisk -rvvvvvv* para entrar en la línea de comandos del servidor. En la Figura 60 se puede observar la respuesta obtenida dentro de la interfaz de Asterisk cuando se ejecuta la comunicación y remoción del usuario o contacto desde Zoiper.

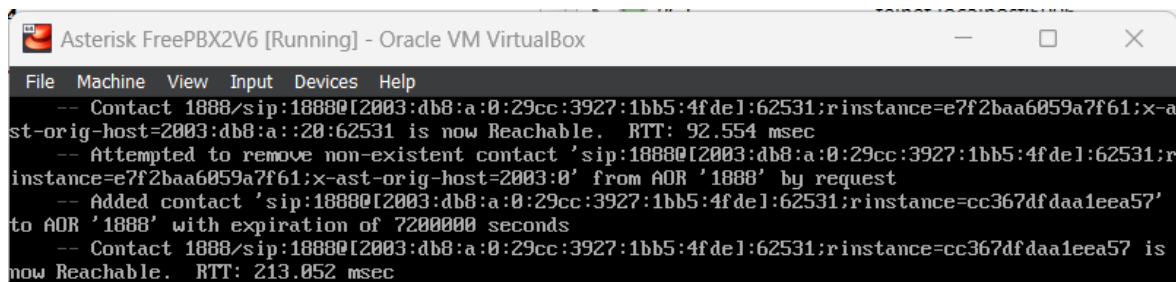


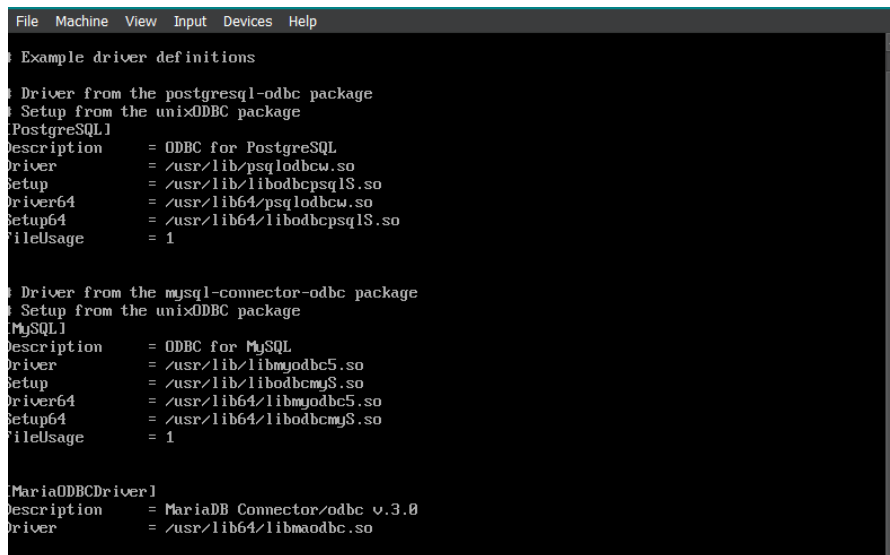
Figura 60. Conexión del softphone zoiper dentro de la interfaz de Asterisk

Integración del Sistema con ODBC (Escenario I)

La comunicación y lectura entre la base de datos y el sistema de Asterisk. Esta conexión puede ser realizada mediante el uso de un socket denominando ODBC, instalar el driver utilizando la siguiente instrucción *sudo yum install odbc-postgresql*, que permitirá agregar todos los archivos y complementos necesarios para poder ejecutar este socket.

Finalizada la instalación se requiere comprobar la existencia del archivo correspondiente al driver que permita la comunicación entre los servidores, para esto se requiere abrir el archivo

/etc/odbcinst.ini, el cual contiene, como se observa en la Figura 61, todas las instancias y drivers instalados en el ordenador para las conexiones ODBC con las diferentes bases de datos.



```
File Machine View Input Devices Help
Example driver definitions

Driver from the postgresql-odbc package
Setup from the unixODBC package
[PostgreSQL]
Description      = ODBC for PostgreSQL
Driver           = /usr/lib/psqlodbcw.so
Setup            = /usr/lib/libodbcpsqlS.so
Driver64         = /usr/lib64/psqlodbcw.so
Setup64          = /usr/lib64/libodbcpsqlS.so
FileUsage        = 1

Driver from the mysql-connector-odbc package
Setup from the unixODBC package
[MySQL]
Description      = ODBC for MySQL
Driver           = /usr/lib/libmyodbc5.so
Setup            = /usr/lib/libodbcmyS.so
Driver64         = /usr/lib64/libmyodbc5.so
Setup64          = /usr/lib64/libodbcmyS.so
FileUsage        = 1

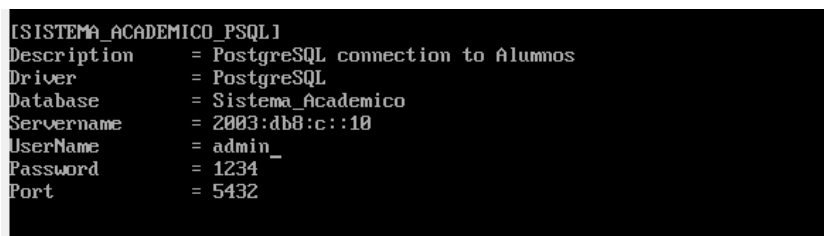
[MariaODBCDriver1]
Description      = MariaDB Connector/odbc v.3.0
Driver           = /usr/lib64/libmaodbc.so
```

Figura 61. Creación de objeto de drivers odbc psql

Para poder entablar una interacción entre Asterisk y PSQL se requiere definir las instancias respectivas, junto con los drivers descritos anteriormente. Se debe abrir el archivo */etc/Odbc.ini*, el cual contiene dentro de sus instrucciones el nombre de la instancia definida para la comunicación entre Asterisk con cualquier base de datos.

En primer lugar, se observa en la Figura 62 la instancia que se tiene por defecto, la cual permite establecer la conexión entre el servidor con MySQL propio de este servicio, sin embargo, se deben colocar una nueva instancia con las instrucciones para establecer la conexión con la base de datos implementada dentro de PSQL.

Para esto se crean la interfaz de conexión denominada *Sistema_Academico_PSQL*, qué contiene el driver PostgreSQL identificado en el archivo *odbcinst.ini*. Así también se deben colocar parámetros básicos propios de la conexión con la base de datos como, *usuario* que en este caso es *admin*, *contraseña*: *1234*, puerto de comunicación: *5432*, Dirección IP del servidor: *2003:db8: c::10* y nombre de la base de datos: *Sistema_Academico*.



```
[SISTEMA_ACADEMICO_PSQL]
Description      = PostgreSQL connection to Alumnos
Driver           = PostgreSQL
Database         = Sistema_Academico
Servername       = 2003:db8:c::10
Username         = admin_
Password         = 1234
Port             = 5432
```

Figura 62. Conexión de ODBC con la base de datos Sistemas_Academico

Para comprobar que existe una conexión entre Asterisk y PSQL mediante el socket implementado se puede ejecutar el comando *isql -v Sistema_Academico_PSQL*. Este comando entrega una respuesta de si está o no conectado o existe o no interacción mediante

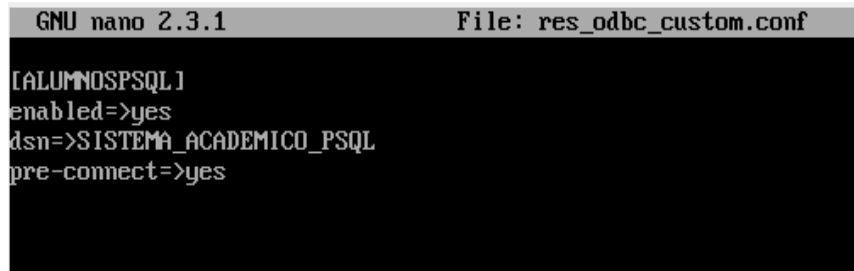
ODBC entre ambos servidores. Como se observa en la Figura 64 el socket denominado *Sistema_Academico_PSQL* tiene una conexión con la base de datos Sistema Academico.



```
Connected!
sql-statement
help [tablename]
quit
SQL> _
```

Figura 63. Comprobación de comunicación entre PSQL y Asterisk por ODBC

Para poder utilizar el socket implementado y extraer la información de la base de datos se requiere hacer configuraciones dentro de los archivos de Asterisk que ejecutan las llamadas. El primero de estos archivos es *res_odbc_custom.conf*, donde se inicializa el socket ALUMNOSPSQL, lo habilita, direcciona hacia el driver respectivo y establece una conexión previa para no tener que iniciar y cerrar una conexión con la BD. Esta configuración se puede observar en la Figura 64.



```
GNU nano 2.3.1 File: res_odbc_custom.conf
[ALUMNOSPSQL]
enabled=>yes
dsn=>SISTEMA_ACADEMICO_PSQL
pre-connect=>yes
```

Figura 64. Habilitación de inicio de objeto Sistema_Academico_PSQL

Seguido de la inicialización del socket de comunicación ODBC, se procede a crear las funciones respectivas dentro del archivo *func_odbc_conf*, mismas que contienen cada uno de los QUERY requeridos para la búsqueda de información dentro de la base de dato. Para esto se crean siete funciones diferentes cada una para extraer diferente información. En la Figura 65 se puede observar cada una de estas funciones y el QUERY asignado para la búsqueda de información. Esta se nombra con cada una de las variables que se van a extraer para poder utilizarlas dentro del DIALPLAN.


```

GNU nano 2.3.1 File: /etc/asterisk/extensions_custom.conf
[ivr_academico]
include => consulta_bd
include => consulta_webservice

[consulta_bd]
exten=> 2001,1,Answer()
exten=> 2001,2,Playback(Bienvenida)
exten=> 2001,3,Playback(Cedula)
exten=> 2001,4,Read(CI)
exten=> 2001,5,Set(nombre=${ODBC_nombre(${CI}})
exten=> 2001,6,GotoIf($I"${nombre}"=""?fcal:ccal)
exten=> 2001,7(ccal),Set(apellido=${ODBC_apellido(${CI}})
exten=> 2001,8(ccal),Festival(ESTUDIANTE)
exten=> 2001,9(ccal),Festival(${nombre})
exten=> 2001,10(ccal),Festival(${apellido})
exten=> 2001,11(ccal),Playback(Opcion1)
exten=> 2001,12(ccal),Playback(Opcion2)
exten=> 2001,13(ccal),WaitExten(3)
exten=> 2001,14(ccal),Goto(2001,11)
;CEDULA INCORRECTA
exten=> 2001,n(fcal),Festival(Numero de Cedula Incorrecto)
exten=> 2001,n(fcal),Festival(Saliendo del Sistema)
exten=> 2001,n(fcal),Hangup
;SI SELECCIONA 1 ENTRA A ESTE CODIGO
exten=> 1,1,Answer()
exten=> 1,n,Festival(INGRESE EL CODIGO DE ASIGNATURA)
exten=> 1,n,Read(codigo)
exten=> 1,n,Festival(UN MOMENTO PORFAVOR)
exten=> 1,n,Set(materia=${ODBC_codasignatura(${CI},${codigo}})
exten=> 1,n,GotoIf($I"${materia}"=""?nmat:smat)
exten=> 1,n(smat),Festival(BUSCANDO INFORMACION)
exten=> 1,n(smat),Set(asignatura=${ODBC_asignatura(${CI},${codigo}})

```

Figura 65. Generación de funciones de llamada ODBC en archivo *func_odbc.conf*

Generadas las funciones respectivas para cada uno de los datos necesarios se procede a la generación del DIALPLAN. Para esto se abre el archivo *extensions_custom.conf*, el cual contiene los procesos y orden de ejecución de las diferentes etapas de la llamada. Para el escenario se ha creado el plan de llamada denominado [consulta_bd].

El procedimiento que sigue este plan de marcación inicia recibiendo una llamada de cualquier Softphone hacia la extensión 2001. Como primer punto la extensión contesta, para después realizar una lectura del teclado para una cantidad de 10 dígitos mientras ejecuta un audio de bienvenida el cual indica al usuario que debe ingresar el número de cedula del estudiante que requiere conocer su información almacenando lo colocado en el teclado en la variable *CI*.

Se genera una variable denominada *nombre* la cual ejecuta el Query ODBC de la función con el mismo nombre, enviando como atributo la variable *CI* leída desde el teclado del teléfono. Se sigue a un bucle *GotoIf* el cual validará si se encontró o no un dato con ese número de cédula. Sí dentro de la búsqueda se logró encontrar la información del estudiante se procede a activar el TTS indicando el nombre y apellido del estudiante, seguido del menú de marcación, mismo que está compuesto por 2 opciones: la primera es marcar *1* para seleccionar la materia por código y la segunda opción marcando *2* es visualizar todas las materias. Se utiliza el comando *WaitExtend*, mismo que permite direccionar la llamada hacia 2 extensiones diferentes (1 - 2). Por otro lado, sí el número de cédula fue incorrecto o el

estudiante no está registrado dentro del sistema nuestro TTS nos indicará que el número de cédula no es válido y colgará la llama tal y como se observa en la Figura 66.

```
GNU nano 2.3.1 File: /etc/asterisk/extensions_custom.conf
live_academico
include => consulta_bd
include => consulta_webservice

[consulta_bd]
exten=> 2001,1,Answer()
exten=> 2001,2,Playback(Bienvenida)
exten=> 2001,3,Playback(Cedula)
exten=> 2001,4,Read(CI)
exten=> 2001,5,Set(nombre=${ODBC_nombre(${CI}})
exten=> 2001,6,GotoIf($[${nombre}=""?]fcal:ccal)
exten=> 2001,7(ccal),Set(apellido=${ODBC_apellido(${CI}})
exten=> 2001,8(ccal),Festival(ESTUDIANTE)
exten=> 2001,9(ccal),Festival(${nombre})
exten=> 2001,10(ccal),Festival(${apellido})
exten=> 2001,11(ccal),Playback(Opcion1)
exten=> 2001,12(ccal),Playback(Opcion2)
exten=> 2001,13(ccal),WaitExten(3)
exten=> 2001,14(ccal),Goto(2001,11)
;CEDULA INCORRECTA
exten=> 2001,n(fcal),Festival(Numero de Cedula Incorrecto)
exten=> 2001,n(fcal),Festival(Saliendo del Sistema)
exten=> 2001,n(fcal),Hangup
```

Figura 66. Dialplan escenario 1 - ingreso de Datos

El proceso continúa diferenciando las dos extensiones, en dónde la extensión 1 mediante el TTS solicitará el ingreso del código de asignatura. Se hará una lectura del teclado y se almacenará en la variable *código*. Se ejecutará el QUERY en donde se verificará la materia mediante el código ingresado y se direccionará a una instrucción GotoIf.

Se hará la búsqueda del nombre de la asignatura, así como el código para buscarlo dentro de la tabla Materias. Posteriormente el TTS indicará la asignatura a la cual se está solicitando la información e indicará mediante la instrucción SayNumber el valor de la nota solicitada para así finalizar la llamada. Por otro lado, si no existe el código registrado en la base dedatos, el sistema agradecerá la búsqueda y finalizará la llamada, tal y como se observa en la Figura 67.

```
exten=> 1,n(smat),Festival(BUSCANDO INFORMACION)
exten=> 1,n(smat),Set(asignatura=${ODBC_asignatura(${CI},${codigo}})
exten=> 1,n(smat),Set(busqueda=${ODBC_busqueda(${CI},${materia}})
exten=> 1,n(smat),Festival(Su nota en)
exten=> 1,n(smat),Festival(${asignatura})
exten=> 1,n(smat),Festival(es)
exten=> 1,n(smat),SayNumber(${busqueda})
exten=> 1,n(smat),Festival(Sobre Veinte)
exten=> 1,n(smat),Festival(Gracias por comunicarte al sistema de notas)
exten=> 1,n(smat),Hangup()
exten=> 1,n(nmat),Festival(CODIGO DE ASIGNATURA INCORRECTO)
exten=> 1,n(nmat),Festival(Gracias por comunicarte al sistema de notas)
exten=> 1,n(nmat),Hangup()
;2 PARA VISUALIZAR TODAS LAS NOTAS
exten=> 2,1,Answer()
exten=> 2,n,Set(nota=${ODBC_nota1(${CI}})
exten=> 2,n,Set(nota2=${ODBC_nota2(${CI}})
```

Figura 67. Dialplan menú de selección de visualización de notas

Finalmente, si en el menú de selección se escoge la opción 2, esto direccionará hacia la extensión del mismo número, la cual mediante el TTS procederá a nombrar cada una de las asignaturas registradas con su respectiva nota mediante el comando SayNumber y así

finalizando la llamada. Si no se escogió ninguna de las 2 opciones del menú, el sistema indicará que la opción ingresada no es válida y finalizará la llamada tal como se observa en la Figura 68.

```

exten=> 2,n,Festival(Su nota en Matematicas es)
exten=> 2,n,SayNumber(${nota})
exten=> 2,n,Festival(sobre veinte)
exten=> 2,n,Festival(Su nota en Ciencias es)
exten=> 2,n,SayNumber(${nota2})
exten=> 2,n,Festival(sobre veinte)
exten=> 2,n,Festival(Gracias por Comunicarte al Sistema de Notas)
exten=> 2,n,Hangup()
;TECLA INVALIDA
exten => i,n,Answer()
exten => i,1,Festival(OPCION INVALIDA)
exten => i,n,Hangup()

```

Figura 68. Dialplan respuesta de extensión menú 2

Para comprobar el correcto funcionamiento del sistema en el escenario 1, se realiza una llamada de prueba, en dónde se ejecutará la llamada a la extensión 2001 y se ingresará el número de cédula CI=0604938514. Cómo se observa en la Figura 69, el sistema detecta que el estudiante se llama Fernando Salcan y redirecciona la llamada hacia el menú de selección para escoger si se quiere visualizar todas las materias o únicamente una de ellas por código.

```

> 0x7f1cfc2d3b78 -- Strict RTP qualifying stream type: audio
> 0x7f1cfc2d3b78 -- Strict RTP switching source address to [2003:db8:a:a:0:34e6:995b:5333]:acd
:58966
> 0x7f1cfc2d3b78 -- Strict RTP learning complete - Locking on source address [2003:db8:a:a:0:
e6:995b:5333:acd]:58966
-- Executing [2001@ivr_academico:3] Playback("PJSIP/1888-00000002", "Cedula") in new stack
-- <PJSIP/1888-00000002> Playing 'Cedula.slin' (language 'es')
-- Executing [2001@ivr_academico:4] Read("PJSIP/1888-00000002", "CI") in new stack
-- User entered '0604938514'
-- Executing [2001@ivr_academico:5] Set("PJSIP/1888-00000002", "nombre=Cristian") in new stack
-- Executing [2001@ivr_academico:6] GotoIf("PJSIP/1888-00000002", "0?fcal:ccal") in new stack
-- Goto [ivr_academico,2001,7)
-- Executing [2001@ivr_academico:7] Set("PJSIP/1888-00000002", "apellido=Salcan") in new stack
-- Executing [2001@ivr_academico:8] Festival("PJSIP/1000-00000002", "ESTUDIANTE") in new stack
-- Executing [2001@ivr_academico:9] Festival("PJSIP/1888-00000002", "Cristian") in new stack
-- Executing [2001@ivr_academico:10] Festival("PJSIP/1888-00000002", "Salcan") in new stack
-- Executing [2001@ivr_academico:11] Playback("PJSIP/1888-00000002", "Opcion1") in new stack
-- <PJSIP/1888-00000002> Playing 'Opcion1.slin' (language 'es')
-- Executing [2001@ivr_academico:12] Playback("PJSIP/1000-00000002", "Opcion2") in new stack
-- <PJSIP/1888-00000002> Playing 'Opcion2.slin' (language 'es')
-- Executing [2001@ivr_academico:13] WaitExten("PJSIP/1888-00000002", "3") in new stack
-- Timeout on PJSIP/1888-00000002, continuing...
-- Executing [2001@ivr_academico:14] Goto("PJSIP/1888-00000002", "2001,11") in new stack
-- Goto [ivr_academico,2001,11)
-- Executing [2001@ivr_academico:11] Playback("PJSIP/1888-00000002", "Opcion1") in new stack
-- <PJSIP/1888-00000002> Playing 'Opcion1.slin' (language 'es')
-- Executing [2001@ivr_academico:12] Playback("PJSIP/1888-00000002", "Opcion2") in new stack
-- <PJSIP/1000-00000002> Playing 'Opcion2.slin' (language 'es')
-- Executing [2001@ivr_academico:13] WaitExten("PJSIP/1888-00000002", "3") in new stack
-- Timeout on PJSIP/1888-00000002, continuing...
-- Executing [2001@ivr_academico:14] Goto("PJSIP/1888-00000002", "2001,11") in new stack
-- Goto [ivr_academico,2001,11)
-- Executing [2001@ivr_academico:11] Playback("PJSIP/1888-00000002", "Opcion1") in new stack
-- <PJSIP/1888-00000002> Playing 'Opcion1.slin' (language 'es')
-- Executing [2001@ivr_academico:12] Playback("PJSIP/1888-00000002", "Opcion2") in new stack
-- <PJSIP/1000-00000002> Playing 'Opcion2.slin' (language 'es')
FreePBX> _

```

Figura 69. Respuesta en Asterisk de Llamada con CI=0604938514

Finalmente, en esta prueba se escoge dentro del menú de selección la opción uno, misma que solicita el código de asignatura, para lo cual se ingresa por teclado 1028 y se realiza la búsqueda respectiva dentro de las bases de datos correspondientes. Cómo se observa en la Figura 70 se obtuvo la información de nota uno correspondiente a matemáticas e indicando que el estudiante posee 10/20 en esta asignatura, así finalizando la llamada.

```

LA MATERIA POR CODIGO") in new stack
-- Executing [20010from-internal:10] Festival("PJSIP/1888-00000001", "MARQUE DOS PARA VISUALIZAR
TODAS LAS ASIGNATURAS") in new stack
-- Executing [20010from-internal:11] WaitExten("PJSIP/1888-00000001", "3") in new stack
-- Executing [10from-internal:1] Answer("PJSIP/1888-00000001", "") in new stack
-- Executing [10from-internal:2] Festival("PJSIP/1888-00000001", "INGRESE EL CODIGO DE ASIGNATUR
A") in new stack
-- Executing [10from-internal:3] Read("PJSIP/1888-00000001", "codigo") in new stack
-- User entered '1028'
-- Executing [10from-internal:4] Festival("PJSIP/1888-00000001", "UN MOMENTO PORFAVOR") in new s
tack
-- Executing [10from-internal:5] Set("PJSIP/1888-00000001", "materia=NOTA1") in new stack
-- Executing [10from-internal:6] GotoIf("PJSIP/1888-00000001", "0?nmat") in new stack
-- Executing [10from-internal:7] Festival("PJSIP/1888-00000001", "BUSCANDO INFORMACION") in new
stack
-- Executing [10from-internal:8] Set("PJSIP/1888-00000001", "asignatura=Matematicas") in new sta
ck
-- Executing [10from-internal:9] Set("PJSIP/1888-00000001", "busqueda=10") in new stack
-- Executing [10from-internal:10] Festival("PJSIP/1888-00000001", "Su nota en") in new stack
-- Executing [10from-internal:11] Festival("PJSIP/1888-00000001", "Matematicas") in new stack
-- Executing [10from-internal:12] Festival("PJSIP/1888-00000001", "es") in new stack
-- Executing [10from-internal:13] SayNumber("PJSIP/1888-00000001", "10") in new stack
-- <PJSIP/1888-00000001> Playing 'digits/10.ulaw' (language 'es')
-- Executing [10from-internal:14] Festival("PJSIP/1888-00000001", "Sobre Veinte") in new stack
-- Executing [10from-internal:15] Festival("PJSIP/1888-00000001", "Gracias por comunicarte al si
stema de notas") in new stack
freepbx*CLI> _

```

Figura 70. Respuesta en Asterisk para la consulta por código

Integración del Sistema con Web Service (Escenario II)

Para la implementación del servicio de consulta de datos mediante el escenario II, se procede a la generación del DIALPLAN denominado consulta_webservice tal y como se observa en la Figura 71. Se da la bienvenida al Sistema y se solicita al usuario el ingreso de su número de cedula de 10 dígitos, se guarda la información dentro de una variable denominada CI para dar paso a la ejecución de la función AGI, misma que permite acceder a la dirección donde se encuentra el archivo *prueba.php* que contiene el algoritmo que permite la obtención de información por medio de estas instrucciones, tal y como se vio en secciones anteriores. Al ejecutar este archivo se recibirá una validación, si no existe ese número de cédula en la base de datos, el sistema notificará al usuario que ese número de CI no existe y cerrará la llamada. En el caso de que exista el usuario que se está buscando, el algoritmo AGI devolverá las variables nombre y apellido y el TTS lo mencionará para después indicar el menú de opciones de búsqueda tal y como se observa en la Figura 71.

```

[consulta_bd]
exten=> 2001,1,Answer()
exten=> 2001,2,Playback(Bienvenida)
exten=> 2001,3,Playback(Cedula)
exten=> 2001,4,Read(CI)
exten=> 2001,5,Set(nombre=${ODBC_nombre(${CI}})
exten=> 2001,6,GotoIf($["${nombre}"=""?]fcal:ccal)
exten=> 2001,7(ccal),Set(apellido=${ODBC_apellido(${CI}})
exten=> 2001,8(ccal),Festival(ESTUDIANTE)
exten=> 2001,9(ccal),Festival(${nombre})
exten=> 2001,10(ccal),Festival(${apellido})
exten=> 2001,11(ccal),Playback(Opcion1)
exten=> 2001,12(ccal),Playback(Opcion2)
exten=> 2001,13(ccal),WaitExten(3)
exten=> 2001,14(ccal),Goto(2001,11)
;CEDULA INCORRECTA
exten=> 2001,n(fcal),Festival(Numero de Cedula Incorrecto)
exten=> 2001,n(fcal),Festival(Saliendo del Sistema)
exten=> 2001,n(fcal),Hangup
;SI SELECCIONA 1 ENTRA A ESTE CODIGO
exten=> 1,1,Answer()
exten=> 1,n,Festival(INGRESE EL CODIGO DE ASIGNATURA)
exten=> 1,n,Read(codigo)
exten=> 1,n,Festival(UN MOMENTO PORFAVOR)
exten=> 1,n,Set(materia=${ODBC_codasignatura(${CI},${codigo}})
exten=> 1,n,GotoIf($["${materia}"=""?]nmat:smat)
exten=> 1,n(smat),Festival(BUSCANDO INFORMACION)
exten=> 1,n(smat),Set(asignatura=${ODBC_asignatura(${CI},${codigo}})

```

Figura 71. Dialplan del escenario 2 mediante AGI

Cuando se haya seleccionado una de las 2 opciones, se enviará a la extensión uno o extensión 2 tal y como se lo planteó en el Escenario I. El sistema mediante la función SayNumber indicará la nota obtenida de la base de datos hoy dependiendo de la materia y agradecerá al usuario el uso del sistema para cerrar la llamada tal y como se observa en la Figura 72. Si existe un número de extensión no válida el sistema hoy cancelará la llamada y se deberá iniciar desde el principio.

```

[consulta_webservice]
exten => 2002,1,Answer()
exten => 2002,2,Playback(Bienvenida)
exten => 2002,3,Playback(Cedula)
exten => 2002,4,Read(CI)
exten => 2002,5,Festival(Buscando Informacion, Un momento Porfavor)
exten => 2002,6,Agi(/var/lib/asterisk/agi-bin/prueba.php,${CI})
exten => 2002,7,Gotoif(${!${validacion}="No existe"}?cal1:cal2)
exten => 2002,8(cal2),Festival('Estudiante ${nombre} ${apellido}')
exten => 2002,9(cal2),Festival('Presione tres si desea conocer la nota por codigo')
exten => 2002,10(cal2),Festival('Presione cuatros si de desea conocer todas las no
exten => 2002,11(cal2),WaitExten(3);
exten => 2002,12(cal2),Goto(2002,9);
;CEDULA INCORRECTA
exten => 2002,n(cal1), Festival(Numero de cedula incorrecta)
exten => 2002,n(cal1), Festival(Saliendo del Sistema)
exten => 2002,n(cal1), Hangup

;SI INGRESA POR LA OPCION 3
exten => 3,1,Answer()
exten => 3,n, Festival(Ingrese el codigo de la asignatura)
exten => 3,n, Read(CODIGO)
exten => 3,n, Festival(Buscando informacion)

```

Figura 72. Dialplan selección de menú escenario 2.

El algoritmo o servicio prueba.php que se menciona en el DIALPLAN, se puede observar en la Figura 74. Dentro de este programa hace referencia a la librería phpagi, que contiene las instrucciones que permiten la interacción entre este lenguaje con Asterisk, adicional se incluye un servicio que se encuentra dentro de la máquina virtual de Ubuntu en la dirección [2003:db8:c::10]:80/pruebacon.php, que ejecutará el archivo para la interacción con la base de datos. Posteriormente se crea un objeto AGI y se extrae la información obtenida desde el servidor, el argumento 1 que corresponde a la lectura de las cédulas de identidad o variable CI. Se realiza el QUERY para la búsqueda dentro de la base de datos y se extrae la información por medio de un arreglo, y así dividirá cada una de sus columnas en las variables correspondiente a la información necesaria del usuario, para así validar si estos datos existen y colocar estas variables dentro de la interfaz de Asterisk.

```

GNU nano 2.3.1 File: /var/lib/asterisk/agi-bin/prueba.php
#!/usr/bin/php -q
<?php
require_once(__DIR__ . '/phpagi/phpagi.php');
include('http://[2003:db8:c::10]:80/pruebacon.php');
// TimeZone
date_default_timezone_set('America/Bogota');
ob_implicit_flush(true);
// Execution Timeout
set_time_limit(30);
// New AGI Object
$agi = new AGI();
$CI=$SERVER['argv'][1];
$agi->verbose($CI);
$query="SELECT * FROM NOTAS1 WHERE ci='$CI'";
$result=pg_query($query);
$row=pg_fetch_array($result);
if ($row!=NULL)
{
$nombre=$row[1];
$apellido=$row[2];
$nota1=$row[4];
$nota2=$row[5];
}
else
{
$validacion="No existe";
}
$agi->set_variable("validacion",$validacion);
$agi->set_variable("nombre",$nombre);
$agi->set_variable("apellido",$apellido);
$agi->set_variable("nota1",$nota1);
$agi->set_variable("nota2",$nota2);

```

Figura 73. Archivo PHP AGI para la búsqueda dentro de la base de datos

Dentro del servidor de Ubuntu se encuentra el archivo *pruebacon.php*, que contiene la información para la conexión de la base de datos misma que será ejecutada por medio de la interfaz AGI dentro de Asterisk los parámetros de este archivo se pueden observar en la Figura 74.

```

$dbconn = pg_connect("host=2003:db8:c::10 port=5432 dbname=ALUMNOS user=ALUMNO$
$prueba = pg_query($dbconn, "SELECT * FROM NOTAS1");
?>

```

Figura 74. Código PHP para la conexión con la BD

Se procede a realizar la prueba de marcado mediante la extensión 2002 desde la extensión 1888. Para esto se ingresa el valor de cédula 1234567891 y se espera que se ejecuten las instrucciones del programa. Tal y como se observa en la Figura 75 se ejecuta la instrucción ágil donde se encuentra el archivo de *prueba.php* ejecutan las instrucciones dentro de este indicando la información obtenida de la base de datos en este caso el estudiante Xavier Moreno.

```

== Setting global variable 'SIPDOMAIN' to '2003:db8:b:10'
== Using SIP RTP Audio TOS bits 184
== Using SIP RTP Audio TOS bits 184 in TCLASS field.
== Using SIP RTP Audio CoS mark 5
-- Executing [2002@from-internal:1] Answer("PJSIP/1888-0000002", "") in new stack
> 0x7fd7c40d4650 -- Strict RTP learning after remote address set to: [2003:db8:a:0:201:53376]
> 0x7fd7c40d4650 -- Strict RTP qualifying stream type: audio
> 0x7fd7c40d4650 -- Strict RTP switching source address to [2003:db8:a:0:29cc:3927:1bb5:4fde1:53376]
-- Executing [2002@from-internal:2] Festival("PJSIP/1888-0000002", "Bienvenido al Sistema de Consulta de Notas") in new stack
-- Executing [2002@from-internal:3] Festival("PJSIP/1888-0000002", "Ingrese su numero de Cedula de 10 DIGITOS") in new stack
> 0x7fd7c40d4650 -- Strict RTP learning complete - Locking on source address [2003:db8:a:0:29cc:3927:1bb5:4fde1:53376]
-- Executing [2002@from-internal:4] Read("PJSIP/1888-0000002", "CI") in new stack
-- User entered '1234567891'
-- Executing [2002@from-internal:5] Festival("PJSIP/1888-0000002", "Buscando Informacion, Un momento Porfavor") in new stack
freepbx*CLI>
Broadcast message from root@freepbx.sangoma.local (Wed Feb  7 15:00:29 2024):
Firewall service now starting.
-- Executing [2002@from-internal:6] AGI("PJSIP/1888-0000002", "/var/lib/asterisk/agi-bin/prueba.php,1234567891") in new stack
-- Launched AGI Script /var/lib/asterisk/agi-bin/prueba.php
/var/lib/asterisk/agi-bin/prueba.php,1234567891: 1234567891
-- <PJSIP/1888-0000002>AGI Script /var/lib/asterisk/agi-bin/prueba.php completed, returning 0
-- Executing [2002@from-internal:7] GotoIf("PJSIP/1888-0000002", "0?call:cal2") in new stack
-- Goto (from-internal,2002,8)
-- Executing [2002@from-internal:8] Festival("PJSIP/1888-0000002", "'Estudiante Xavier Moreno'") in new stack
-- Executing [2002@from-internal:9] Festival("PJSIP/1888-0000002", "'MARQUE UNO PARA SELECCIONAR LA MATERIA POR CODIGO'") in new stack
freepbx*CLI>

```

Figura 75. Respuesta del sistema escenario 2

Se hace la búsqueda de las notas mediante la opción 2 la cual otorga y muestra los valores de todas las asignaturas como se observa en la Figura 76 el sistema envía nota 2 y nota uno con los valores de 11 y 10 respectivamente, los menciona mediante el TTS y agradece la utilización del servicio para finalizar con la llamada, demostrando el correcto funcionamiento del prototipo.

```

) in new stack
-- Executing [2002@from-internal:9] Festival("PJSIP/1888-0000000", "'MARQUE UNO PARA SELECCIONAR LA MATERIA POR CODIGO'") in new stack
-- Executing [2002@from-internal:10] Festival("PJSIP/1888-0000000", "'MARQUE DOS PARA SELECCIONAR TODAS LAS MATERIAS'") in new stack
-- Executing [2002@from-internal:11] WaitExten("PJSIP/1888-0000000", "3") in new stack
-- Executing [2@from-internal:1] Answer("PJSIP/1888-0000000", "") in new stack
-- Executing [2@from-internal:2] Set("PJSIP/1888-0000000", "nota=10") in new stack
-- Executing [2@from-internal:3] Set("PJSIP/1888-0000000", "nota2=11") in new stack
-- Executing [2@from-internal:4] Festival("PJSIP/1888-0000000", "Su nota en Matematicas es") in new stack
-- Executing [2@from-internal:5] SayNumber("PJSIP/1888-0000000", "10") in new stack
-- <PJSIP/1888-0000000> Playing 'digits/10.ulaw' (language 'es')
-- Executing [2@from-internal:6] Festival("PJSIP/1888-0000000", "sobre veinte") in new stack
-- Executing [2@from-internal:7] Festival("PJSIP/1888-0000000", "Su nota en Ciencias es") in new stack
-- Executing [2@from-internal:8] SayNumber("PJSIP/1888-0000000", "11") in new stack
-- <PJSIP/1888-0000000> Playing 'digits/11.ulaw' (language 'es')
-- Executing [2@from-internal:9] Festival("PJSIP/1888-0000000", "sobre veinte") in new stack
-- Executing [2@from-internal:10] Festival("PJSIP/1888-0000000", "Gracias por Comunicarte al Sistema de Notas") in new stack
-- Executing [2@from-internal:11] Hangup("PJSIP/1888-0000000", "") in new stack
== Spawn extension (from-internal, 2, 11) exited non-zero on 'PJSIP/1888-0000000'
-- Executing [h@from-internal:1] Macro("PJSIP/1888-0000000", "hangupcall") in new stack
-- Executing [s@macro-hangupcall:1] GotoIf("PJSIP/1888-0000000", "1?theend") in new stack
-- Goto (macro-hangupcall,s,3)
-- Executing [s@macro-hangupcall:3] ExecIf("PJSIP/1888-0000000", "0?Set(CDR(recordingfile)=)") in new stack
-- Executing [s@macro-hangupcall:4] NoOp("PJSIP/1888-0000000", " montior file=") in new stack
-- Executing [s@macro-hangupcall:5] GotoIf("PJSIP/1888-0000000", "1?skipagi") in new stack
-- Goto (macro-hangupcall,s,7)
-- Executing [s@macro-hangupcall:7] Hangup("PJSIP/1888-0000000", "") in new stack
== Spawn extension (macro-hangupcall, s, 7) exited non-zero on 'PJSIP/1888-0000000' in macro 'hangupcall'
== Spawn extension (from-internal, h, 1) exited non-zero on 'PJSIP/1888-0000000'

```

Figura 76. Respuesta selección de línea dos para búsqueda de todas las asignaturas