UNIVERSIDAD DE EL SALVADOR

FACULTAD DE INGENIERÍA Y ARQUITECTURA

ESCUELA DE INGENIERÍA DE SISTEMAS INFORMÁTICOS

PROTOTIPO DE INFRAESTRUCTURA DE NUBE PÚBLICA PARA ORGANIZACIONES ORIENTADAS AL DESARROLLO DE SOFTWARE

PRESENTADO POR:

DÍAZ CHÁVEZ, EDWIN JOSÉ - DC13018

HERNÁNDEZ MEJÍA, ERICK SERAFÍN - HM11019

MONGE MIRANDA, JENNIFER EUNICE - MM16110

PARA OPTAR AL TÍTULO DE:

INGENIERO DE SISTEMAS INFORMÁTICOS

CIUDAD UNIVERSITARIA, NOVIEMBRE 2024

UNIVERSIDAD DE EL SALVADOR

RECTOR:

M.Sc. JUAN ROSA QUINTANILLA

SECRETARIO GENERAL:

LIC. PEDRO ROSALÍO ESCOBAR CASTANEDA

FACULTAD DE INGENIERÍA Y ARQUITECTURA

DECANO:

ING. LUIS SALVADOR BARRERA MANCÍA

SECRETARIO:

ARQ. RAUL ALEXANDER FABIAM ORELLANA

ESCUELA DE INGENIERIA DE SISTEMAS INFORMÁTICOS

DIRECTOR:

ING. CÉSAR AUGUSTO GONZÁLEZ RODRÍGUEZ

UNIVERSIDAD DE EL SALVADOR

FACULTAD DE INGENIERIA Y ARQUITECTURA

ESCUELA DE INGENIERÍA DE SISTEMAS INFORMÁTICOS

Trabajo de Graduación previo a la opción de Grado de:

INGENIERO DE SISTEMAS INFORMÁTICOS

Título:

PROTOTIPO DE INFRAESTRUCTURA DE NUBE PÚBLICA PARA ORGANIZACIONES ORIENTADAS AL DESARROLLO DE SOFTWARE

Presentado por:

Díaz Chávez, Edwin José - DC13018

Hernández Mejía, Erick Serafín - HM11019

Monge Miranda, Jennifer Eunice - MM16110

Trabajo de Graduación Aprobado por:

Docente Asesor:

MSc. JULIO DAMIÁN MORALES AYALA

SAN SALVADOR, NOVIEMBRE DE 2024

Trabajo de Graduación Aprobado por:

Docente Asesor:

MSc. JULIO DAMIÁN MORALES AYALA

INDICE

INDICE DE FIGURAS

INDICE TABLAS

1 Introducción

En un mundo cada vez más digitalizado, la computación en la nube se ha consolidado como un elemento clave para la innovación, permitiendo a las organizaciones optimizar sus operaciones y responder con agilidad a los cambios demandantes del mercado. Sin embargo, algunas empresas, especialmente aquellas que se dedican al desarrollo de software, enfrentan desafíos al intentar acceder a servicios en la nube que se adapten perfectamente a sus necesidades en términos de escalabilidad, seguridad y costos.

En respuesta a estos desafíos potenciales, esta investigación se centra en la creación de un prototipo de plataforma de nube pública, diseñado para ofrecer una amplia gama de servicios virtualizados, orientados específicamente a empresas tecnológicas dedicadas al desarrollo de aplicaciones informáticas. Inspirada en las soluciones líderes del sector como AWS, Azure y Google Cloud, esta plataforma buscar ser una alternativa competitiva que no solo permita a las organizaciones gestionar sus recursos tecnológicos de manera más eficiente, sino que también garantice la transparencia y optimización de costos mediante una gestión efectiva de cuotas y facturación.

A lo largo del documento, se comienza con una descripción del proyecto, donde se expone un panorama general de los objetivos y el alcance de este. Seguidamente, el planteamiento del problema aborda los antecedentes y la justificación del prototipo, contextualizando su necesidad en el mercado actual. El marco teórico proporciona un contexto detallado sobre la computación en la nube, abarcando conceptos clave y modelos existentes. La metodología de la investigación describe el enfoque adoptado para identificar la necesidad de una nueva opción de nube pública, detallando las técnicas y herramientas utilizadas.

Posteriormente, se presentan los resultados obtenidos y el desarrollo de la solución, donde se detalla la arquitectura y configuración del prototipo para replicar una nube pública. El caso de estudio demuestra la viabilidad de la plataforma y, finalmente, se evalúan tanto la factibilidad técnica como la factibilidad económica del proyecto, analizando su potencial para ser adoptado por empresas del sector.

Seguidamente, se presentan conclusiones basadas en los resultados obtenidos del prototipo, evaluando su impacto sobre las necesidades tecnológicas de las empresas que se dedican al desarrollo de software en El Salvador. Finalmente, a partir de estos resultados, se formulan recomendaciones que incluyen optimizaciones técnicas y operativas, así como estrategias para incrementar la adopción de este tipo de soluciones en el mercado.

2 Descripción del proyecto

Esta investigación se centra en la creación de un prototipo de plataforma innovadora de nube pública que ofrece una amplia gama de servicios virtualizados para organizaciones enfocadas en el desarrollo de software. Inspirada en líderes del mercado como AWS, Azure y Google Cloud, esta plataforma tiene como objetivo proporcionar una alternativa competitiva y robusta en el ámbito de la computación en la nube, con un enfoque particular en la transparencia y eficiencia financiera. Uno de los principales diferenciadores de la plataforma es su capacidad para optimizar los gastos en la nube a través de una gestión de cuotas y facturación.

La plataforma ofrece diversos servicios, como almacenamiento en bloques permitiendo a los usuarios guardar archivos, imágenes, videos y otros tipos de datos. También incluye servicios de bases de datos para satisfacer diversas necesidades de almacenamiento y consultas de registros, con la capacidad de gestionar la información de manera eficiente. Además, se han configurado servicios de máquinas virtuales que permitan a los usuarios gestionarlas y desplegarlas con el sistema operativo de su elección.

Asimismo, se ofrece servicios de contenedores, lo que facilita la encapsulación de aplicaciones junto con sus dependencias, y un sistema de orquestación que permite a los usuarios la capacidad de gestionar de manera eficiente la creación, modificación y eliminación de los recursos en la nube a través de plantillas. Cabe destacar que la nube tiene el servicio de balanceo de carga que posibilita la distribución del tráfico de red de manera eficiente entre múltiples servidores, garantizando la alta disponibilidad y optimización de recursos. Se incluyen servicios para el aprovisionamiento de DNS para traducir nombres de dominios en direcciones IP.

Además, se incluyen opciones de software como servicio (SaaS), ofreciendo soluciones de software listas para usarse. De igual manera, se integra el servicio de API de autenticación para garantizar conexiones seguras a la red y la verificación de los usuarios al acceder a las aplicaciones.

Para garantizar la redundancia y continuidad, la infraestructura está distribuida en al menos dos zonas de disponibilidad para asegurar el servicio en caso de fallos. Además, se incorporan herramientas de telemetría para monitorear el rendimiento de los servicios.

3 Planteamiento del problema

3.1 Antecedentes

La computación en la nube ha transformado la forma en que las empresas gestionan sus recursos tecnológicos, y su impacto es particularmente relevante en las empresas dedicadas al desarrollo de software. Este tipo de organizaciones dependen cada vez más de la nube para acceder a entornos de desarrollo flexibles, escalables y eficientes en costos, permitiéndoles optimizar sus procesos sin tener que invertir en infraestructura física.

El concepto de computación en la nube tiene sus orígenes en los años 60, cuando Joseph Carl Robnett Licklider introdujo la idea de una “red intergaláctica de ordenadores”, donde los usuarios pudieran acceder a datos y aplicaciones desde cualquier lugar (Wang, Ranjan, Chen & Benatallah, 2011).

Además, la introducción de herramientas de virtualización, como las máquinas virtuales en la década de 1970, permitió la creación de entornos aislados y personalizados para distintos proyectos dentro de una misma infraestructura física, lo que marcó un importante paso en la compartición de recursos (educaOpen, s.f.).

Esto fue fundamental para que las empresas de software puedan simular diferentes configuraciones de producción antes de lanzar aplicaciones, minimizando errores y garantizando una mayor fiabilidad en los productos finales. Esta evolución no solo ha facilitado la implementación de soluciones, sino que también ha impulsado la innovación y la colaboración entre equipos distribuidos geográficamente.

Sin embargo, no fue hasta el inicio del siglo XXI, con el surgimiento de empresas como Amazon Web Services (AWS), Luego, en 2006 Amazon lanzó Elastic Compute Cloud (EC2) como un servicio web comercial que permitía a pequeñas empresas e individuos alquilar

computadoras para ejecutar sus propias aplicaciones informáticas (educaOpen, s.f). Con este tipo de innovaciones, las empresas dejaron de depender de costosas infraestructuras locales, adoptando servicios en la nube que les proporcionaban entornos de trabajo bajo demanda.

A lo largo de la década de 2000, se produjeron más avances significativos. En 2009 con el auge de la Web 2.0, Google y otros comenzaron a ofrecer aplicaciones empresariales basadas en navegadores, como Google Apps (educaOpen, s.f.). En 2010, Rackspace Hosting y NASA lanzaron OpenStack, una iniciativa de software de nube de código abierto que buscaba brindar servicios de computación en la nube utilizando hardware estándar. En 2012, Oracle Cloud ofreció un conjunto de soluciones informáticas en la nube, mientras que Google Drive se lanzó como un servicio de almacenamiento y sincronización de archivos. (educaOpen, s.f.).

3.2 Justificación

La creación de un prototipo de plataforma de nube pública responde a la creciente demanda de soluciones tecnológicas que aborden los desafíos del mercado actual, especialmente en el ámbito del desarrollo de software. Muchas organizaciones enfrentan limitaciones en el acceso a servicios en la nube que sean escalables y seguros, lo que restringe su capacidad para innovar y adaptarse a las tecnologías en constante evolución.

Desplegar este prototipo permite explorar y demostrar cómo una plataforma de nube pública puede ofrecer una amplia gama de servicios relevantes para empresas de desarrollo de software, tales como almacenamiento, bases de datos, máquinas virtuales, contenedores y orquestación, DNS, balanceadores de carga, software como servicios, telemetría y API de autenticación. Al ofrecer esta solución, se atenderá la necesidad de las empresas de software para mejorar su eficiencia y reducir costos, al mismo tiempo que se atraerá a nuevas empresas que buscan soluciones tecnológicas modernas sin la necesidad de grandes inversiones en infraestructura propia.

El prototipo también contempla el uso de regiones dentro de la arquitectura para garantizar alta disponibilidad de los servicios, distribuyendo los recursos en distintas ubicaciones geográficas. Esta iniciativa proporciona a las empresas de desarrollo de software las herramientas necesarias para optimizar sus operaciones, salvaguardar y preservar la integridad de la información de los clientes, mejorar su agilidad empresarial y mantener su competitividad en un entorno empresarial cada vez más digitalizado y exigente.

3.3 Pregunta de investigación

¿Es factible crear una plataforma de nube pública que ofrezca una amplia gama de servicios virtualizados para satisfacer las necesidades tecnológicas de empresas dedicadas al desarrollo de aplicaciones informáticas?

3.4 OBJETIVOS

3.4.1 Objetivo General

Desplegar una plataforma de nube pública que ofrezca una amplia gama de servicios virtualizados para satisfacer las necesidades tecnológicas de empresas dedicadas a la creación de aplicaciones informáticas.

3.4.2 Objetivos Específicos

Desplegar un servicio de almacenamiento en bloques que permita gestionar datos de manera segura.

Configurar un servicio de bases de datos para almacenar los datos y recuperar información de manera eficiente.

Desplegar un servicio de máquinas virtuales que permita a las organizaciones gestionar instancias virtuales según sus necesidades.

Desplegar un servicio de contenedores y orquestación que facilite el despliegue y la gestión de aplicaciones en la plataforma.

Configurar un servicio de sistema de nombres de dominio (DNS) para traducir nombres de dominios en direcciones IP.

Desplegar un servicio de balanceadores de carga que distribuya el tráfico de red de manera uniforme para mejorar el rendimiento.

Ofrecer aplicaciones de software listas para usar, permitiendo a los usuarios acceder fácilmente a las herramientas sin necesidad de ser instaladas.

Desplegar una API de autenticación para gestionar de forma centralizada el acceso a los recursos de la plataforma.

Desplegar un servicio de facturación y control de costos en la plataforma que permita a los usuarios monitorear el uso de recursos y comprender claramente los costos asociados a los servicios de la plataforma.

3.5 Alcances

1. Diseño y configuración de una infraestructura de nube pública, que permita a las organizaciones desplegar y gestionar servicios virtualizados, incluyendo la configuración de almacenamiento, bases de datos, máquinas virtuales, contenedores y orquestación.
2. Diseño de una infraestructura basada en Openstack para el despliegue de una nube pública limitada por dos zonas de disponibilidad para garantizar alta disponibilidad ante fallos.
3. Configuración de Ceph como la solución de almacenamiento distribuido para la plataforma Openstack, proporcionando un sistema de archivos escalable, altamente disponible y resiliente para soportar las cargas de trabajo críticas.
4. Sistema de facturación para monitorear y gestionar el uso de recursos en la nube, permitiendo un control efectivo de los costos.
5. Configuración de un servidor LDAP para gestionar de manera centralizada la autenticación y facilitando la administración de usuarios.
6. Configuración de un nodo de firewall utilizando OPNsense para proteger y redirigir el tráfico de red a través de una VPN, garantizando un control eficiente y seguro de las conexiones.

4 Marco Teórico

4.1 Virtualización

La virtualización es una tecnología que permite generar múltiples entornos simulados o recursos dedicados a partir de un solo sistema de hardware (Red Hat, 2024). Un software conocido como hipervisor se conecta directamente al hardware y permite dividir el sistema en entornos separados, distintos y seguros, denominados máquinas virtuales. Estas máquinas dependen de la capacidad del hipervisor para separar los recursos del hardware y distribuirlos adecuadamente.

El hardware físico, equipado con un hipervisor, se denomina anfitrión, mientras que las numerosas máquinas virtuales que utilizan sus recursos son invitados. Estos invitados tratan los recursos informáticos (como la CPU, la memoria y el almacenamiento) como un conjunto de recursos que se pueden reubicar fácilmente. Los operadores pueden controlar instancias virtuales de CPU, memoria, almacenamiento y otros recursos, de modo que los invitados reciban los recursos que necesitan cuando los necesitan. (Red Hat, 2024).

4.2 Beneficios de la virtualización

La virtualización de recursos permite a los administradores agrupar sus recursos físicos, de modo que su hardware pueda convertirse en un producto básico. De esta forma, la infraestructura heredada, cuyo mantenimiento es costoso, pero que admite aplicaciones importantes, se puede virtualizar para un uso óptimo.

Los administradores ya no tienen que esperar a que todas las aplicaciones estén certificadas en el nuevo hardware; solo hay que configurar el entorno, migrar la máquina virtual y todo funciona como antes. Durante las pruebas de regresión, se puede crear o copiar un banco de pruebas fácilmente, lo que elimina la necesidad de contar con hardware de prueba dedicado o

servidores de desarrollo redundantes. Con la capacitación y los conocimientos adecuados, estos entornos se pueden optimizar aún más para obtener mayores capacidades y densidad. (Red Hat, 2024).

4.3 Seguridad en la virtualización

La virtualización es una solución elegante para muchos problemas de seguridad comunes. En entornos donde las políticas de seguridad requieren que los sistemas estén separados por un firewall, esos dos sistemas podrían residir de manera segura en la misma caja física. En un entorno de desarrollo, cada desarrollador puede tener su propio sandbox, inmune al código malicioso o descontrolado de otro desarrollador. (Red Hat, 2024).

4.4 Cloud Computing vs. Virtualización

Según Red Hat (2024), Es fácil Confundir Cloud computing y virtualización, en particular porque ambos giran en torno a la separación de recursos del hardware para crear un entorno útil. La virtualización ayuda a crear nubes, pero eso no la convierte en computación en la nube. Se puede ver de la siguiente manera:

- Las funciones del hardware son separadas mediante la virtualización.
- La computación en la nube es más bien una solución que se basa en esa división

4.5 Hipervisores

(Red Hat, 2024) Un hipervisor es un software que crea y ejecuta máquinas virtuales (VM), este es llamado a veces monitor de máquina virtual (VMM), aísla el sistema operativo y

los recursos del hipervisor de las máquinas virtuales y permite la creación y administración de esas VM.

Cuando el hardware físico, se utiliza como hipervisor, se denomina anfitrión, mientras que las numerosas máquinas virtuales que utilizan sus recursos son invitados. El hipervisor trata los recursos (como CPU, memoria y almacenamiento) como un grupo que puede reasignarse fácilmente entre invitados existentes o nuevas máquinas virtuales.

Existen muchas opciones de hipervisores de proveedores tradicionales y de código abierto. VMware es una opción popular para la virtualización y ofrece el hipervisor ESXi y la plataforma de virtualización vSphere. La máquina virtual basada en kernel (KVM) es una opción popular y de código abierto que está integrada en el kernel de Linux. Otras opciones incluyen Xen, que es de código abierto, y Microsoft Hyper-V.

4.6 Tipos de hipervisores

Hay dos tipos diferentes de hipervisores que se pueden utilizar para la virtualización: hipervisores de tipo 1 bare metal y de tipo 2 software instalado.

Un hipervisor de tipo 1, también conocido como hipervisor nativo o de hardware, se ejecuta directamente en el hardware del anfitrión para administrar los sistemas operativos invitados. Reemplaza al sistema operativo del anfitrión y los recursos de la máquina virtual se programan directamente en el hardware a través del hipervisor. Este tipo de hipervisor es más común en un centro de datos empresarial u otros entornos basados en servidores. KVM, Microsoft Hyper-V y VMware vSphere son ejemplos de un hipervisor de tipo 1. KVM se fusionó con el kernel de Linux en 2007, por lo que, si está utilizando una versión moderna de Linux, ya tiene acceso a KVM.

Un hipervisor tipo 2 también se conoce como hipervisor alojado y se ejecuta en un sistema operativo convencional como una capa de software o aplicación. Funciona abstrayendo los sistemas operativos invitados del sistema operativo anfitrión. Los recursos de la máquina virtual se programan en un sistema operativo anfitrión, que luego se ejecuta en el hardware. Un hipervisor de tipo 2 es mejor para usuarios individuales que desean ejecutar múltiples sistemas operativos en una computadora personal.

VMware Workstation y Oracle VirtualBox son ejemplos de un hipervisor de tipo 2 (Red Hat, 2024).

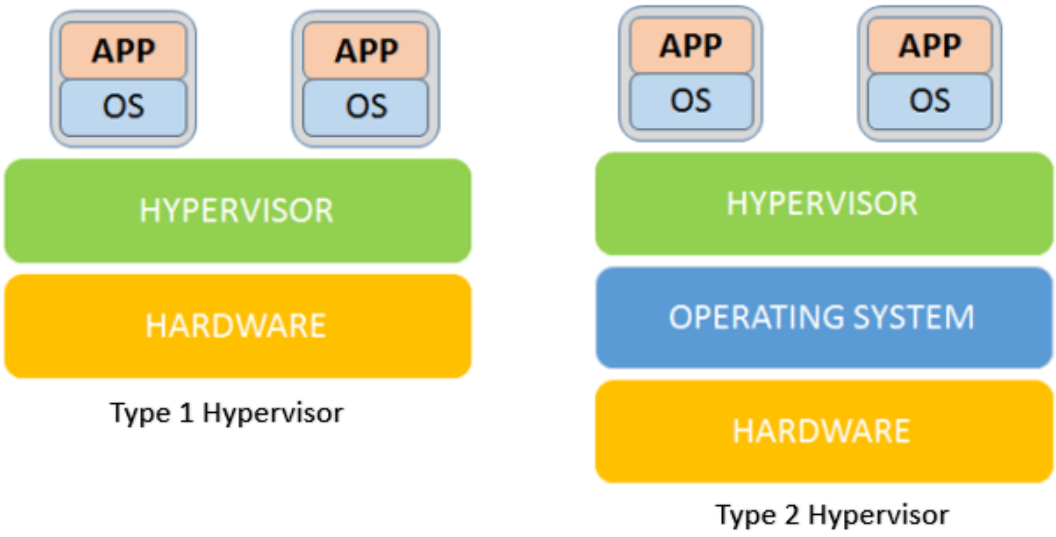


Figura 1: *Tipos de Hypervisores (devopspace, 2024).*

4.7 Máquinas virtuales vs. Contenedores

Según (Red Hat, 2024), los contenedores y las máquinas virtuales parecen similares. Ambos son entornos informáticos empaquetados que combinan varios componentes de TI y los aíslan del resto de un sistema. La distinción importante está en cómo escalan y su portabilidad.

Un contenedor es un conjunto de uno o más procesos que están aislados del resto del sistema. El contenedor permite que el proceso acceda únicamente a las solicitudes de recursos

que se han especificado. Estos límites de recursos garantizan que el contenedor pueda ejecutarse en un nodo que tenga suficiente capacidad.

Las máquinas virtuales contienen su propio sistema operativo (SO), lo que les permite realizar múltiples funciones que consumen muchos recursos a la vez. Los mayores recursos disponibles para las máquinas virtuales les permiten abstraer, dividir, duplicar y emular servidores, SO, escritorios, bases de datos y redes completos. Un hipervisor también permite ejecutar varios sistemas operativos en las máquinas virtuales, pero los contenedores solo pueden ejecutar un único tipo de sistema operativo. Un contenedor que se ejecuta en un servidor Linux, por ejemplo, solo puede ejecutar un sistema operativo Linux. A veces se piensa en los contenedores como un reemplazo de los hipervisores, aunque esto no es del todo exacto, ya que los contenedores y la virtualización satisfacen necesidades diferentes.

La virtualización proporciona los recursos que pueden utilizar los contenedores. Estas máquinas virtuales son entornos en los que se pueden ejecutar contenedores, pero estos no están vinculados a entornos virtuales. Algunos programas (como Red Hat OpenShift Virtualization) pueden organizar contenedores y administrar máquinas virtuales, pero eso no significa que las dos tecnologías sean iguales.

Las máquinas virtuales tienen capacidades limitadas porque los hipervisores que las crean están vinculados a los recursos limitados de una máquina física. Los contenedores, por otro lado, comparten el mismo núcleo del sistema operativo y las aplicaciones de paquetes con sus entornos de ejecución, de modo que todo se puede mover, abrir y usar en configuraciones de desarrollo, prueba y producción.

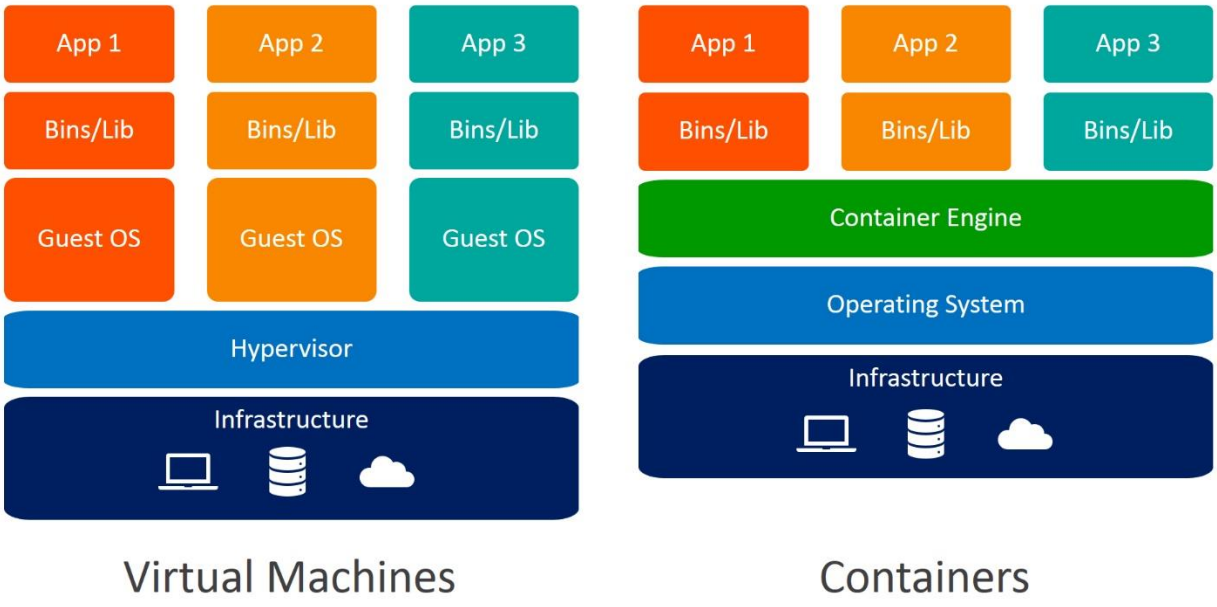


Figura 2: *Máquinas virtuales VS contenedores (devskillbuilder, 2024).*

4.8 Contenedores

Según (Red Hat, 2024) los contenedores son tecnologías que permiten empaquetar y aislar aplicaciones con todo su entorno de ejecución (todos los archivos necesarios para ejecutarse). Esto facilita el traslado de la aplicación contenida entre entornos (desarrollo, prueba, producción, etc.) sin perder la funcionalidad completa. Los contenedores también desempeñan un papel crucial en la seguridad de TI. Al incorporar seguridad en la cadena de suministro de contenedores y defender la infraestructura, los contenedores se mantienen confiables, escalables y seguros. También puede trasladar fácilmente la aplicación en contenedores entre entornos de nube pública, privada e híbrida y centros de datos (o locales) con un comportamiento y una funcionalidad consistentes.

4.9 Beneficios de los contenedores

Según (Red Hat, 2024) los contenedores ayudan a reducir los conflictos entre los equipos de desarrollo y operaciones al separar las áreas de responsabilidad. Los desarrolladores pueden

centrarse en sus aplicaciones y los equipos de operaciones pueden centrarse en la infraestructura. Y, como los contenedores se basan en tecnología de código abierto, obtienes los últimos y mejores avances tan pronto como están disponibles.

Las tecnologías de contenedores se pueden categorizar según su función:

Herramientas para crear y gestionar contenedores, lo que ofrece mayor flexibilidad al momento de trabajar con soluciones que requieren integrarse. En esta categoría podemos hallar a Podman, Buildah, y Skope.

Herramientas para la orquestación de contenedores las cuales pueden automatizar y administrar las redes internas de los contenedores con el fin de poder crear aplicaciones a mayor escala. Entre los orquestadores populares se puede encontrar a kubernetes que fue creado por google y docker swarm el cual es la alternativa de kubernetes que viene incluida dentro del entorno de docker

Herramientas para ejecución de contenedores entre los cuales se puede encontrar docker y containerd

Los contenedores comparten el mismo núcleo del sistema operativo y aíslan los procesos de aplicación del resto del sistema, de modo que todo se puede mover, abrir y utilizar en configuraciones de desarrollo, prueba y producción. Debido a que son livianos y portátiles, los contenedores brindan la oportunidad de acelerar el desarrollo y satisfacer las necesidades comerciales a medida que surgen.

4.10 Seguridad en contenedores

Según (Red Hat, 2024) la seguridad de los contenedores implica definir y obedecer prácticas de diseño, implementación y tiempo de ejecución que protejan un contenedor de Linux, desde las aplicaciones que admiten hasta la infraestructura en la que se basan.

Dado que las empresas requieren adoptar nuevos patrones de arquitectura para escalar sus soluciones, los equipos de estas empresas deben enfocarse en la seguridad utilizando servicios de contenedores y orquestación para que no exista mucho cambio en la infraestructura. La seguridad en los contenedores permiten gestionar de manera confiable y eficiente las políticas de seguridad de cada empresa, aplicándolas de forma íntegra y permanente para garantizar la protección de sus entornos y recursos (Red Hat, 2024).

El orquestador forma una parte esencial en el clúster de contenedores, este debe ser configurado con las políticas de seguridad dictaminada por la empresa, de forma que exista una mayor integridad y permanencia de la información. (Red Hat, 2024).

Según Red Hat (2024), la seguridad permanente de los contenedores de la empresa consiste en tres tareas:

- Asegurar la protección del medio por el cual los contenedores y la aplicación se comunicarán
- Asegurar la protección de las aplicaciones y los contenedores esperados.
- Asegurar la protección de las cargas de trabajo que son organizadas en contenedores a través de la orquestación para que estas operen correctamente en tiempo de ejecución.

4.11 Uso de los contenedores

Según (Red Hat, 2024) se puede implementar contenedores para una variedad de cargas de trabajo y casos de uso. Los contenedores le brindan a su equipo la tecnología subyacente necesaria para un estilo de desarrollo nativo de la nube, de modo que puede comenzar con DevOps, CI/CD (integración continua e implementación continua) e incluso optar por la tecnología sin servidor.

Las aplicaciones basadas en contenedores pueden funcionar en arquitecturas de nube altamente distribuidas. El middleware de ejecución de aplicaciones proporciona herramientas para respaldar un entorno unificado para el desarrollo, la distribución, la integración y la automatización (Red Hat, 2024).

También puede implementar tecnologías de integración en contenedores, de modo que pueda escalar fácilmente la forma en que conecta aplicaciones y datos, como la transmisión de datos en tiempo real a través de Apache Kafka. Si está creando una arquitectura de microservicios, los contenedores son la unidad de implementación ideal para cada microservicio y la red de malla de servicios que los conecta (Red Hat, 2024).

Cuando su empresa necesita la máxima portabilidad en múltiples entornos, utilizar contenedores puede ser la decisión más sencilla.

4.12 Cloud Computing

La computación en la nube se ha convertido en una herramienta para la gestión moderna de recursos tecnológicos, permitiendo a las organizaciones y a los individuos a una variedad de servicios a través de internet. Según el Instituto Nacional de Estándares y Tecnología (Téllez Valdés, 2013) “ La computación en la nube se define como un modelo para permitir un acceso conveniente y bajo demanda a una red compartida de recursos informáticos configurables (por ejemplo, red, servidores, almacenamiento, aplicaciones y servicios) o la interacción con el proveedor de servicios. "

Una de las principales ventajas de este modelo es el pago por uso, lo que permite a las organizaciones adaptar rápidamente su capacidad según sus necesidades sin incurrir en el costo y el mantenimiento de centros de datos propios.

4.13 Tipos de modelos de despliegue en la nube

En la Cloud Computing, existen varios modelos de despliegue que se diferencian en términos de organización, y calidad de los servicios ofrecidos. Estos modelos permiten a las organizaciones elegir la configuración que mejor se ajusta a sus necesidades específicas y objetivos empresariales. Los principales modelos de despliegue son:

4.13.1 Nube pública

En una nube pública, el proveedor de servicios y el consumidor de servicios pertenecen a organizaciones diferentes. Las nubes públicas siempre implementan modelos comerciales y solo se contabiliza el uso real de los recursos. Si bien las nubes públicas ofrecen una excelente relación calidad-precio debido a su economía de escala del lado de la oferta, existe la preocupación de que se produzca una dependencia del proveedor (Wang, L., Ranjan, R., Chen, J., & Benatallah, B. 2017).

4.13.2 Nube privada

Los servicios de una nube privada siempre son operados por la misma organización a la que pertenece el consumidor. La motivación para construir y operar una nube privada puede ser la preocupación por la seguridad y la privacidad. Sin embargo, es difícil alcanzar la economía de escala, la disponibilidad y el nivel de seguridad de un proveedor de servicios de nube pública profesional certificado (Wang, L., Ranjan, R., Chen, J., & Benatallah, B. 2017).

4.13.3 Nube híbrida

En una nube híbrida, se combinan los servicios de las nubes pública y privada. Los servicios de nube pública en una nube híbrida se pueden aprovechar para satisfacer cargas pico o para distribuir datos redundantes dentro de la nube para lograr una alta disponibilidad. Una implementación especial de la nube híbrida es la llamada nube privada virtual, en la que los

recursos del proveedor de servicios se asignan de forma transparente a una red de clientes mediante el uso de un túnel de red (Wang, L., Ranjan, R., Chen, J., & Benatallah, B. 2017).

4.14 Modelos de Servicio en la nube

En la computación en la nube, las definiciones varían de empresa a empresa, no obstante, tradicionalmente han existido 3 modelos de servicio que se consideran los principales en la computación en la nube. Cada uno de los modelos representa un aspecto diferente de la computación en la nube. De acuerdo a diversas fuentes, los tres modelos de servicio son los siguientes: Infraestructura como servicio (o por sus siglas en inglés, IaaS), Plataforma como servicio (PaaS), y Software como servicio (SaaS) (AWS, 2024).

4.14.1 Infraestructura como servicio (IaaS)

Según la definición de IBM (IBM, 2024), este modelo es una forma de computación en la nube que ofrece recursos fundamentales de computación, red y almacenamiento a los consumidores bajo demanda (on-demand), a través de Internet y de pago por uso. Este modelo permite que los usuarios finales escalen los recursos que necesiten de acuerdo a su consumo, minimizando los gastos por adelantado o la instalación de infraestructura física innecesaria. Es en este modelo de servicio en la nube en el que se proporciona el control de recursos más bajo en la nube.

AWS (AWS, 2024) describe las ventajas de utilizar este modelo de servicio a través de cinco beneficios: Velocidad, destacando el aprovisionamiento de recursos según la demanda en cuestión de minutos. Rendimiento, enfatizando que este tipo de soluciones permiten que las aplicaciones de los usuarios finales sean escalables verticalmente y en ubicaciones

geográficamente más cercanas a sus clientes. Fiabilidad, donde los recursos de respaldo entran en funcionamiento de manera rápida y predecible. Respaldo y recuperación, que permite la sincronización y la creación de copias de seguridad para mantener el desempeño de las aplicaciones. Y finalmente, Precios competitivos, recalcando que, bajo este modelo, se paga solo por el recurso consumido, haciendo que los recursos sean administrados de manera más eficiente.

4.14.2 Plataforma como servicio (PaaS)

Salesforce define las Plataformas como servicio como un conjunto de servicios en la nube que permite a los usuarios crear aplicaciones a una velocidad que las soluciones físicas no pueden alcanzar. (Microsoft Azure, 2024) Los usuarios son capaces de centrarse en crear la mejor experiencia de usuario gracias a la variedad de elementos que se adquieren que facilitan la administración y configuración de los mismos. Microsoft Azure recalca que las PaaS, al igual que las IaaS, proveen servicios de infraestructura, pero también incluyen middleware, herramientas de desarrollo, servicios de inteligencia empresarial, administración de bases de datos, entre otros. (Microsoft Azure, 2024) Esto implica que las Plataformas como servicio han sido diseñadas para sustentar el ciclo de vida de las aplicaciones web.

IBM (IBM, 2024) describe los beneficios del uso de las Plataformas como servicios sobre estas cinco ventajas: Tiempos de comercialización más rápidos, ya que no es necesario hacer instalaciones físicas del hardware o la instalación del software que se utilice para la creación y mantenimiento de las aplicaciones, sin tiempos de espera. Acceso asequible a una variedad más amplia de recursos, incluyendo sistemas operativos, herramientas de desarrollo, middleware, bases de datos, entre otros. Escalabilidad fácil y rentable, debido a que se pueden adquirir los

recursos necesarios de acuerdo a la demanda en las aplicaciones. Flexibilidad en el desarrollo, ya que se provee un entorno compartido que permite a los equipos acceder a todas las herramientas necesarias. Y, como punto final, la Reducción de costos, al hacer que los clientes eviten los gastos excesivos en infraestructura física y el escalado de aplicaciones, además de reducir o eliminar los costos vinculados al software y las herramientas de desarrollo.

4.14.3 Software como servicio (SaaS)

De acuerdo a la definición de Microsoft Azure, este modelo de servicio en la nube permite a los usuarios conectarse a aplicaciones basadas en la nube y utilizarlas, ejemplo de ello son el correo electrónico, los calendarios y las herramientas ofimáticas en línea. Los SaaS ofrecen una solución integral que se adquiere de un proveedor de servicios en la nube con un modelo de pago por uso (on-demand). Toda la infraestructura subyacente, el middleware, el software y los datos de las aplicaciones están alojados en centros de datos del proveedor, administrando el hardware y software. (Microsoft Azure, 2024)

De acuerdo a AWS (AWS, 2024), los beneficios de utilizar este modelo de servicio son los siguientes: Accesibilidad a la nube, ya que es posible acceder al SaaS desde cualquier dispositivo conectado a internet, permitiendo flexibilidad en los modelos de trabajo. Menos costos iniciales, al estar basados en un modelo de pago por suscripción, reduce los costos iniciales de hardware o software, y no implica adquirir recursos adicionales. Despliegue rápido, al eliminar la configuración e instalación asociada al software, las soluciones son desplegadas tan pronto como se comience el plan de suscripción. Actualizaciones automáticas, lo que permite que los clientes inviertan más tiempo en utilizar la solución mientras las actualizaciones se realizan en segundo plano. Finalmente, la Integración permite que aplicaciones y sistemas

terceros sean integrados a través de APIs, personalizando el software para las necesidades específicas sin costos a la infraestructura.

4.15 Openstack

Según Rackspace Technology (2024), OpenStack es una plataforma de software de código abierto para nubes privadas y públicas. Ofrece la capacidad de crear infraestructura como servicio (IaaS), además de dar a las empresas la ventaja de agregar servidores y componentes de redes y almacenamiento de manera fácil y eficiente a su nube.

OpenStack es gestionado por la Fundación OpenStack, una organización sin fines de lucro enfocada en su desarrollo y mejora constante. Esta plataforma se distingue por su capacidad para ofrecer una infraestructura robusta y escalable, lo que ha facilitado su adopción por parte de grandes empresas y organizaciones.

La amplia aceptación y el éxito de OpenStack se atribuyen en gran medida a su modelo de código abierto, que permite una personalización y flexibilidad considerables para adaptarse a las necesidades específicas de cada entidad. Además, su comunidad activa y en expansión continua impulsa la mejora constante de la plataforma, garantizando que siga siendo una solución líder en la gestión de infraestructuras en la nube.

4.15.1 Descripción general del servicio de Openstack

OpenStack sigue una arquitectura modular que ofrece un conjunto de servicios esenciales, permitiendo la escalabilidad y elasticidad como principios fundamentales en su

diseño.

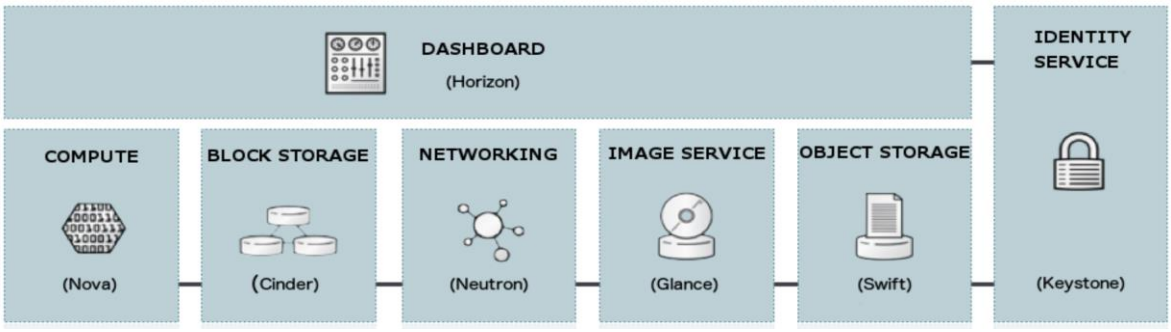


Figura 3: *Diagrama muestra los componentes de OpenStack y su conjunto de servicios básicos (OpenStack, 2024).*

4.16 Apache CloudStack

Apache CloudStack es un software de código abierto diseñado para implementar y administrar grandes redes de máquinas virtuales como una plataforma de computación en la nube de infraestructura como servicio (laaS) altamente disponible y escalable. CloudStack es utilizado por varios proveedores de servicios para ofrecer servicios de nube pública y por muchas empresas para ofrecer una oferta de nube local (privada) o como parte de una solución de nube híbrida (Apache CloudStack's, s. f.).

Apache cloudstack tiene soporte para varios hypervisores, dependiendo de los requerimientos de la nube que se desea desplegar Las opciones más comunes en esta herramienta son: KVM El cual está integrado con el kernel de Linux, VMware vSphere el cual utiliza el hypervisor ESXI's junto al uso de vCenter Server, XenServer el cual utiliza el hypervisor xen como su tecnología de virtualización principal, junto a la API de este mismo que permite administrar los stack de virtualización (Apache CloudStack's, s. f.).

4.17 OpenNebula

Según OpenNebula (2024) se describen como la plataforma de computación en la nube y Edge que unifica la simplicidad y la agilidad de la nube pública con el rendimiento, la seguridad y el control de la nube privada. OpenNebula aporta flexibilidad, escalabilidad, simplicidad e independencia de proveedores para satisfacer las crecientes necesidades de sus desarrolladores y prácticas de DevOps.

OpenNebula permite en cuanto a las aplicaciones, automatizar las operaciones y controlar las máquinas virtuales junto con los clústeres de Kubernetes en un entorno compartido integrado. En el ámbito de la infraestructura OpenNebula es una arquitectura de nube abierta destinada a orquestar la computación, el almacenamiento y las redes mediante software. Permite, además, la integración de las operaciones de nubes privadas, públicas y Edge en un único panel de control, ofreciendo una capa de interoperabilidad.

Open nebula es flexible con el hypervisor que se desea utilizar, dependiendo de las necesidades del proyecto se puede optar por KVM El cual está integrado con el kernel de Linux,

VMware vSphere el cual utiliza el hypervisor ESXI's junto al uso de vCenter Server,

Contenedores linux LXC y MicroVM firecracker (OpenNebula,2024).

5 Metodología de Investigación

Para llevar a cabo la investigación sobre la factibilidad de lanzar una nube pública, se ha elegido un caso de estudio centrado en empresas dedicadas al desarrollo de software. Las bases de la investigación se centran en la facilidad de acceso a los recursos necesarios para estas empresas, tales como almacenamiento, orquestación, manejo de zonas de disponibilidad, computación, servicios de imágenes, entre otras. Se ha utilizado el enfoque cuantitativo dentro de la investigación para recopilar datos confiables y evaluar la viabilidad del proyecto con el fin de crear un prototipo de nube pública que ofrezca una amplia gama de servicios virtualizados basados en las necesidades específicas de las empresas involucradas. Además, la investigación tuvo necesidades puntuales para utilizar una metodología cuantitativa enfocada a descubrir la factibilidad basada en el caso de estudio definido.

5.1 Enfoque cuantitativo

El enfoque cuantitativo de esta investigación se centra en la recopilación de datos que permitan evaluar las características y necesidades tecnológicas de las empresas dedicadas al desarrollo de software. Se busca medir variables como el tamaño de la organización, los modelos de nube más utilizados por las organizaciones, los proveedores de servicios en la nube que las empresas utilizan. Además, se han cuantificado aspectos críticos relacionados con la variabilidad en el manejo de recursos de infraestructuras de nube, la importancia de la alta disponibilidad, y las principales necesidades tecnológicas de las organizaciones, tales como el almacenamiento de datos, la escalabilidad de aplicaciones, seguridad y el cumplimiento normativo, los desafíos que enfrenta el equipo de desarrollo al utilizar servicios de nube pública, y los factores que influyen en la decisión de cambiar de proveedor de nube pública. Estos datos permiten obtener una visión

clara y detallada sobre el estado actual de la infraestructura tecnológica en el sector, facilitando el análisis de la factibilidad de una plataforma de nube pública.

5.2 Población y Muestreo

El universo considerado para la investigación con la metodología cuantitativa se enfocó en organizaciones tecnológicas que se dedican al desarrollo de software en el área metropolitana de San Salvador. Se identificaron 97 empresas que cumplen con esta categoría (CompuTrabajo, 2024). Dada la limitación de contacto y la dificultad para acceder a un número elevado de empresas, se optó por un muestreo no probabilístico mediante un enfoque de muestreo subjetivo por decisión razonada. Según Corbetta (2007), en este tipo de muestreo, las unidades de la muestra no se eligen usando procedimientos probabilísticos, sino en función de ciertas características relevantes.

Se eligieron 7 empresas específicas donde el grupo de investigadores dispuso de contactos personales que facilitaron la recolección de datos. Se optó por incluir criterios relacionados con la viabilidad de la investigación, priorizando aquellas empresas que mostraran el interés en la adopción de nuevas tecnologías, lo que permite que las empresas seleccionadas tuvieran la disposición de evaluar la viabilidad de un servicio de nube pública. También se buscó incluir empresas de diferentes tamaños con el fin de obtener una muestra que refleje la diversidad del sector. Otro de los criterios que se tomaron en cuenta fue que estas empresas tuvieran experiencia en el consumo de servicios en la nube para obtener información más precisa. Finalmente, estos contactos también facilitaron la difusión de la encuesta entre otros empleados del equipo de desarrollo, lo que permitió ampliar el alcance de la investigación y obtener una variedad de perspectivas sobre las necesidades tecnológicas del sector.

5.3 Métodos de recolección de datos

Para llevar a cabo la investigación sobre la factibilidad de lanzar una nube pública, se utilizó la técnica de encuestas como el método principal de recolección de datos. Esta metodología permite obtener información directa y cuantificable sobre las necesidades tecnológicas de las empresas dedicadas al desarrollo de software en el área metropolitana de San Salvador.

Las preguntas fueron diseñadas para recopilar información sobre el uso actual de servicios en la nube, así como las características y necesidades específicas de cada empresa. Siguiendo la clasificación de Corbetta (2007), las encuestas se consideran una herramienta fundamental en la investigación cuantitativa, ya que facilitan el análisis estadístico de los datos recolectados (ver Anexo 1).

El cuestionario se dirigió específicamente a analistas desarrolladores involucrados en el desarrollo de software dentro de las empresas seleccionadas, quienes son responsables de la programación, implementación y mantenimiento de aplicaciones. Este grupo proporcionó una visión clave sobre las necesidades tecnológicas en el proceso de desarrollo y las herramientas utilizadas.

Para la distribución de la encuesta, se utilizó Google Forms, lo que facilitó el acceso y la recolección de datos de manera eficiente. Esta herramienta permitió alcanzar directamente a los analistas desarrolladores, asegurando así que las opiniones y experiencias de grupo fueran reflejadas en la investigación.

6 Resultados

6.1 Análisis de resultados

Se llevó a cabo una encuesta dirigida a analistas desarrolladores en el área metropolitana de El Salvador, específicamente aquellos que trabajan en empresas dedicadas al desarrollo de software. Las preguntas de la encuesta se diseñaron para obtener información relevante sobre el uso de servicios en la nube y las capacidades actuales de las empresas.

Este análisis se centra en identificar las tendencias, necesidades y retos que enfrentan estas empresas en relación con la infraestructura en la nube, lo que permitió fundamentar la propuesta de creación de un prototipo de nube pública como solución.

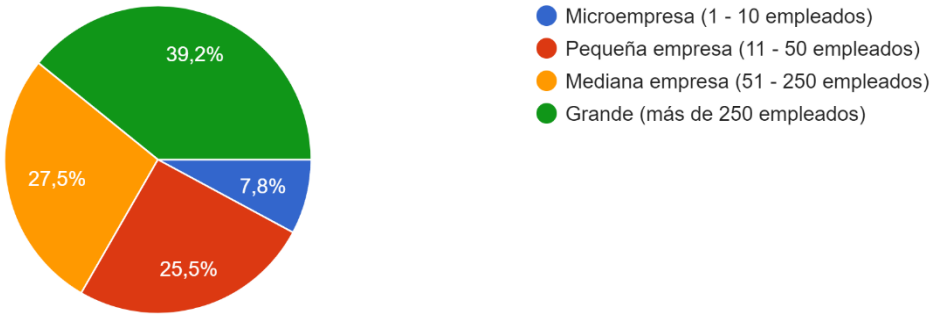


Figura 4: *¿Cuál es el tamaño de la organización donde trabaja?*

La pregunta busca identificar el tamaño de las organizaciones donde laboran los encuestados. De un total de 51 respuestas, se obtuvieron los siguientes resultados, el 39.2% de los encuestados trabajan en grandes empresas con más de 250 empleados, el 27.5% pertenecen a medianas empresas de 51 a 250 empleados, el 25.5% están empleados en pequeñas empresas de 11 a 50 empleados y por último el 7.8% en microempresas entre 1 a 10 empleados.

Este resultado muestra que la mayoría de los encuestados proviene de grandes y medianas empresas, lo que sugiere que existe una fuerte demanda de soluciones de nube

robustas, escalables y seguras. Las grandes empresas tienen requisitos más complejos, mientras que las medianas, pequeñas y microempresas buscan flexibilidad y costos controlados.

Por lo tanto, la creación de una plataforma de nube pública resulta factible, ya que puede atender a las necesidades de organizaciones de diversos tamaños, ofreciendo soluciones adaptadas en términos de escalabilidad y seguridad.

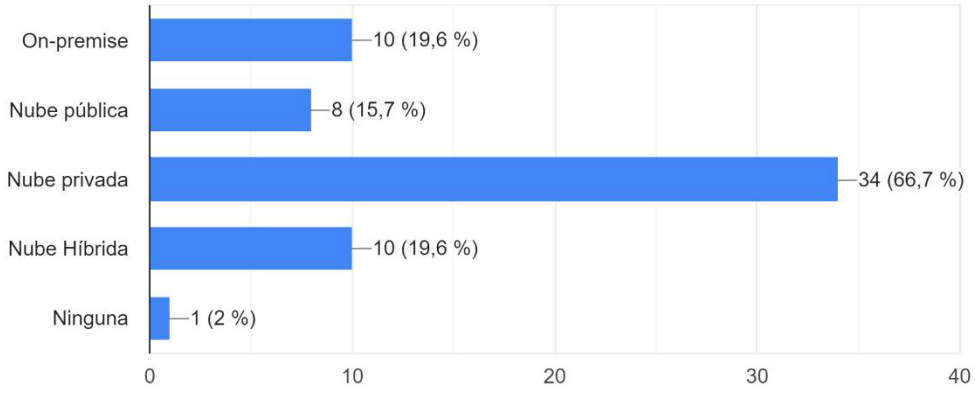


Figura 5: *¿Qué modelo de nube utiliza actualmente su organización?*

La pregunta busca identificar los modelos de nube que utilizan las organizaciones de los encuestados. De un total de 51 respuestas, el 66.7% indica que utilizan nube privada, seguido de un 19.6% que utilizan tanto soluciones on-premise como nubes híbridas. El 15.7% de las organizaciones utiliza una nube pública, mientras que el 2% reporta no utilizar ningún modelo de nube.

Es importante señalar que algunos encuestados podrían no distinguir completamente entre los modelos de nube privada y nube pública. Esto es particularmente relevante en el caso de servicios públicos como AWS o Azure los cuales podrían estar siendo clasificados incorrectamente como “nube privada” debido a la percepción de control o personalización de recursos. Este sesgo podría influir en los resultados, reflejando un mayor uso de nubes privadas del que realmente existe en el mercado.

A pesar de este posible sesgo, los resultados sugieren un nicho en el mercado para una plataforma de nube pública, ya que muchas empresas medianas y aquellas que buscan flexibilidad y menores costos podrían beneficiarse de una plataforma pública siempre que ofrezca características competitivas. Por lo tanto, estos resultados apoyan la factibilidad de crear una nube pública, pero subrayan la importancia de ofrecer servicios robustos que puedan competir con las soluciones privadas como las híbridas.

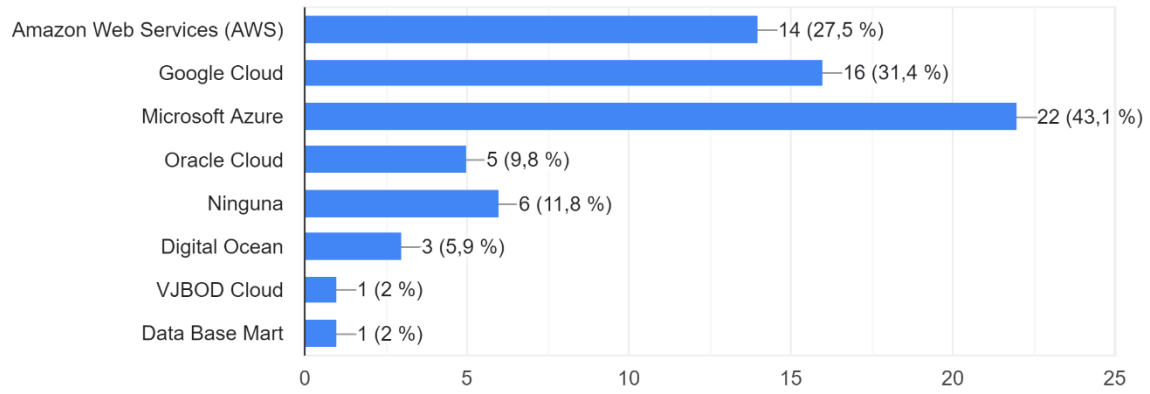


Figura 6: *¿Con cuál proveedor de servicios de nube trabaja actualmente su empresa para el desarrollo o implementación de sus soluciones tecnológicas?*

La encuesta indica la siguiente distribución de uso de proveedores de servicios de nube, de las 51 respuestas, el 27.5% prefiere los servicios de Amazon Web Services, mientras que el 31.4% opta por Google Cloud. Además, el 43.1% utiliza Microsoft Azure, y el 9.8% hace uso de Oracle Cloud, Lo que demuestra una alta aceptación de proveedores establecidos en el mercado. Sin embargo, el 11.8% de los participantes no utiliza actualmente ningún servicio en la nube.

El interés en alternativas más pequeñas, como Digital Ocean, VJBOD Cloud y Data Base Mart, sugiere que las empresas están abiertas a explorar soluciones especializadas. A pesar de esta tendencia hacia la nube pública, la pregunta anterior reveló que el 66.7% de las organizaciones utilizan nubes privadas, lo que puede atribuirse a que las grandes empresas, son

más propensas a elegir este modelo. También es posible que algunos encuestados confundan los servicios públicos, como AWS o Azure, con nubes privadas, clasificándolos erróneamente. Estos hallazgos subrayan la necesidad de una plataforma de nube pública que ofrezca servicios competitivos, especialmente para empresas medianas que buscan flexibilidad y menores costos.

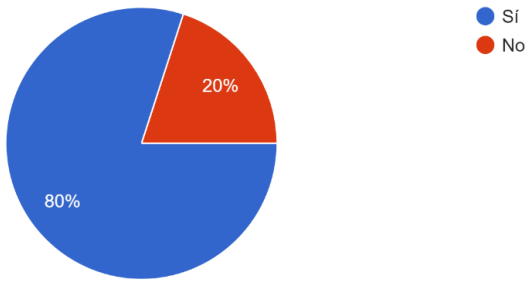


Figura 7: *¿El manejo de recursos de infraestructura es variable dentro de la organización?*

Los resultados de la encuesta indican que, de las 51 respuestas, el 80% de los encuestados perciben un manejo variable de los recursos de infraestructura y el 20% indica que no, esto respalda la factibilidad de crear una plataforma de nube pública por que las empresas tienen la facilidad de escalar los recursos según sea sus necesidades, a largo plazo esto les da un beneficio ya que no tienen que invertir en hardware físico, de tal manera que les ayuda en el ahorro de costos.

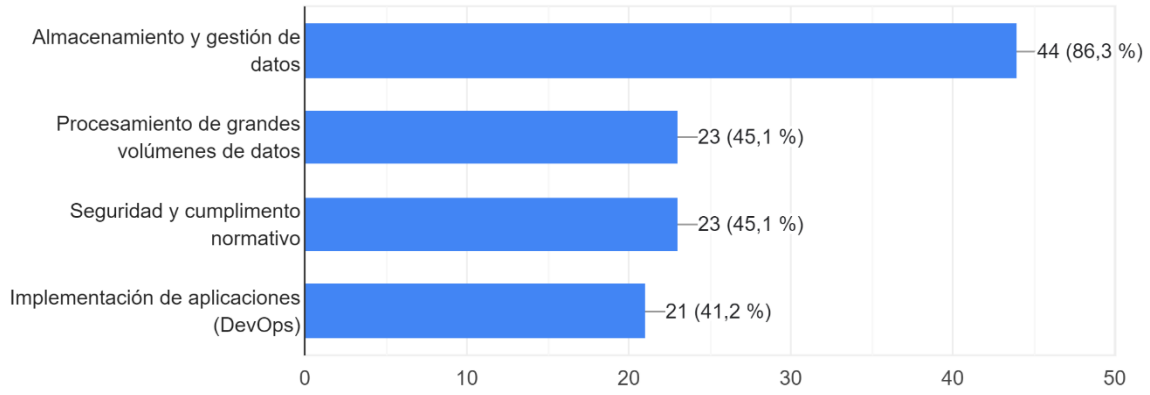


Figura 8: *¿Cuáles son las principales necesidades tecnológicas en la organización?*

Los resultados muestran lo siguiente, de las 51 respuestas obtenidas, el 86.3% de los encuestados seleccionó el almacenamiento y gestión de datos, como una de las principales necesidades tecnológicas de sus organizaciones. Además, el 45.1% de las empresas destacó el procesamiento de grandes volúmenes de datos como una prioridad, mientras que el 45.1% también mencionó la seguridad y el cumplimiento normativo como aspectos críticos. Por otro lado, el 41.2% de los encuestados identificó la implementación de aplicaciones (DevOps) como una necesidad relevante para su infraestructura tecnológica.

Estos resultados indican que las empresas del sector de desarrollo de software tienen múltiples necesidades tecnológicas que pueden ser satisfechas mediante una plataforma de nube pública. Las demandas clave, como el almacenamiento eficiente, la capacidad para procesar grandes volúmenes de datos, la seguridad y el soporte para DevOps, no solo justifica la creación de dicha plataforma, sino que también sugiere que hay un mercado receptivo y una demanda significativa para estos servicios. Esto demuestra la factibilidad de crear una plataforma de nube pública que ofrezca estos servicios, proporcionando una alternativa competitiva y eficiente que

puede mejorar la eficiencia y reducir los costos para las empresas.

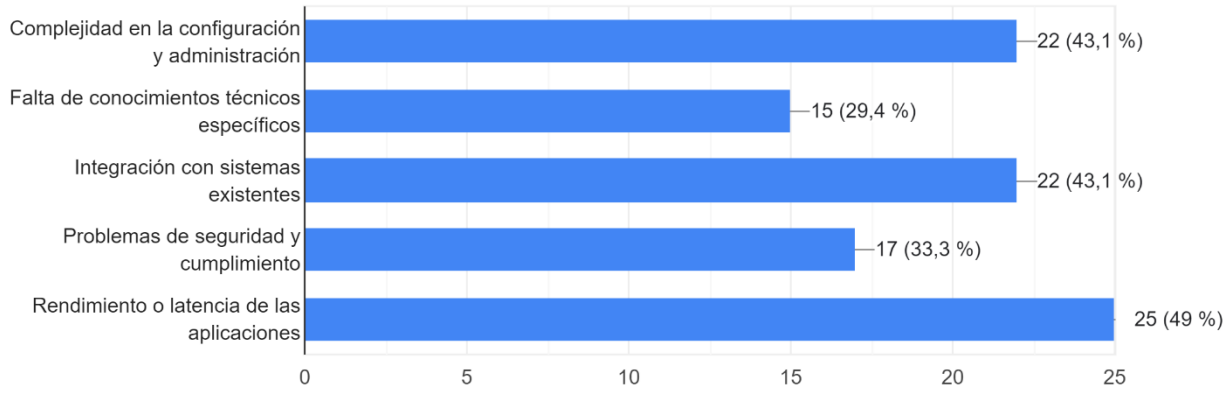


Figura 9: *¿Cuáles son los mayores desafíos que enfrenta el equipo de desarrollo al utilizar servicios de nube pública?*

De las 51 respuestas obtenidas, el 43.1% de los encuestados identificó la complejidad en la configuración y administración de los servicios de nube pública como uno de los principales desafíos. Además, el 29.4% mencionó la falta de conocimientos técnicos específicos, el 43.1% destacó la integración con sistemas existentes, el 33.3% indicó problemas de seguridad y cumplimiento, y el 49% reportó el rendimiento o latencia de las aplicaciones como un desafío significativo.

Estos resultados resaltan que la percepción de complejidad y la necesidad de habilidades técnicas específicas son barreras a la adopción de servicios en la nube. En particular, el desafío del rendimiento es crítico, ya que afecta directamente la experiencia del usuario y la eficiencia operativa de las soluciones. Por lo tanto, abordar las preocupaciones sobre el rendimiento y la latencia será fundamental para evaluar la factibilidad de crear una nueva plataforma de nube pública. Estos resultados respaldan la viabilidad de crear una plataforma de nube pública que ofrezca soluciones a estos desafíos, proporcionando una alternativa competitiva en el mercado.

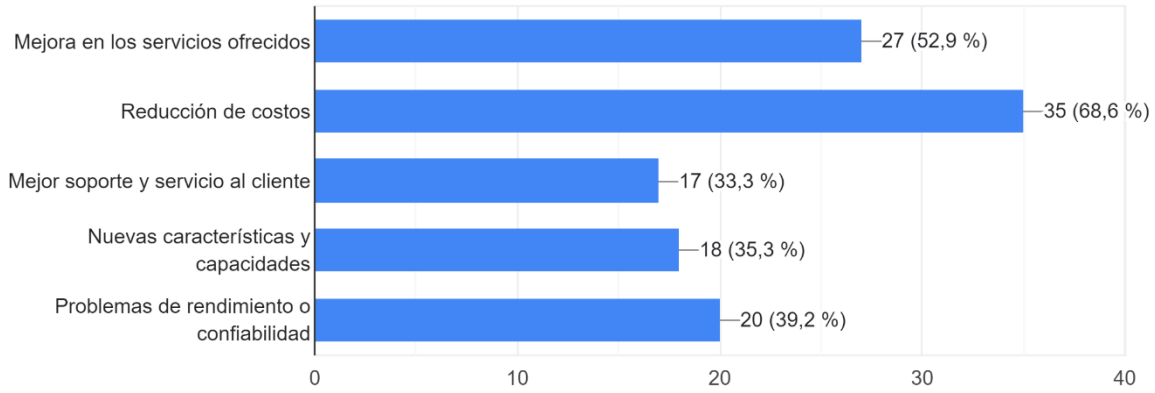


Figura 10: *¿Cuál es el principal factor que influye en la decisión de su organización para cambiar de proveedor de nube pública?*

De las 51 respuestas obtenidas, el 68.6% de los encuestados seleccionó la reducción de costos como uno de los principales factores que influyen en la decisión de cambiar de proveedor de nube pública. Asimismo, un 52.9% de los participantes destacó la mejora en los servicios ofrecidos, y un 39.2% consideró que el mejor soporte y servicio al cliente es crucial. Además, el 35.3% mencionó las nuevas características y capacidades como un factor relevante, mientras que el 33.3% indicó que problemas de rendimiento o confiabilidad también juegan un papel importante en esta decisión.

Los resultados evidencian que las organizaciones valoran significativamente la reducción de costos y la mejora en los servicios al considerar un cambio de proveedor de nube pública. Esto sugiere que las empresas están en búsqueda de soluciones más económicas y eficientes que se alineen mejor con sus necesidades. Por lo tanto, estos resultados respaldan la factibilidad de crear una plataforma de nube pública que no solo ofrezca servicios de calidad competitivos, sino que también provea un servicio en costo-beneficio que favorezca a los potenciales clientes.

7 Diseño del Prototipo

7.1 Arquitectura del Prototipo

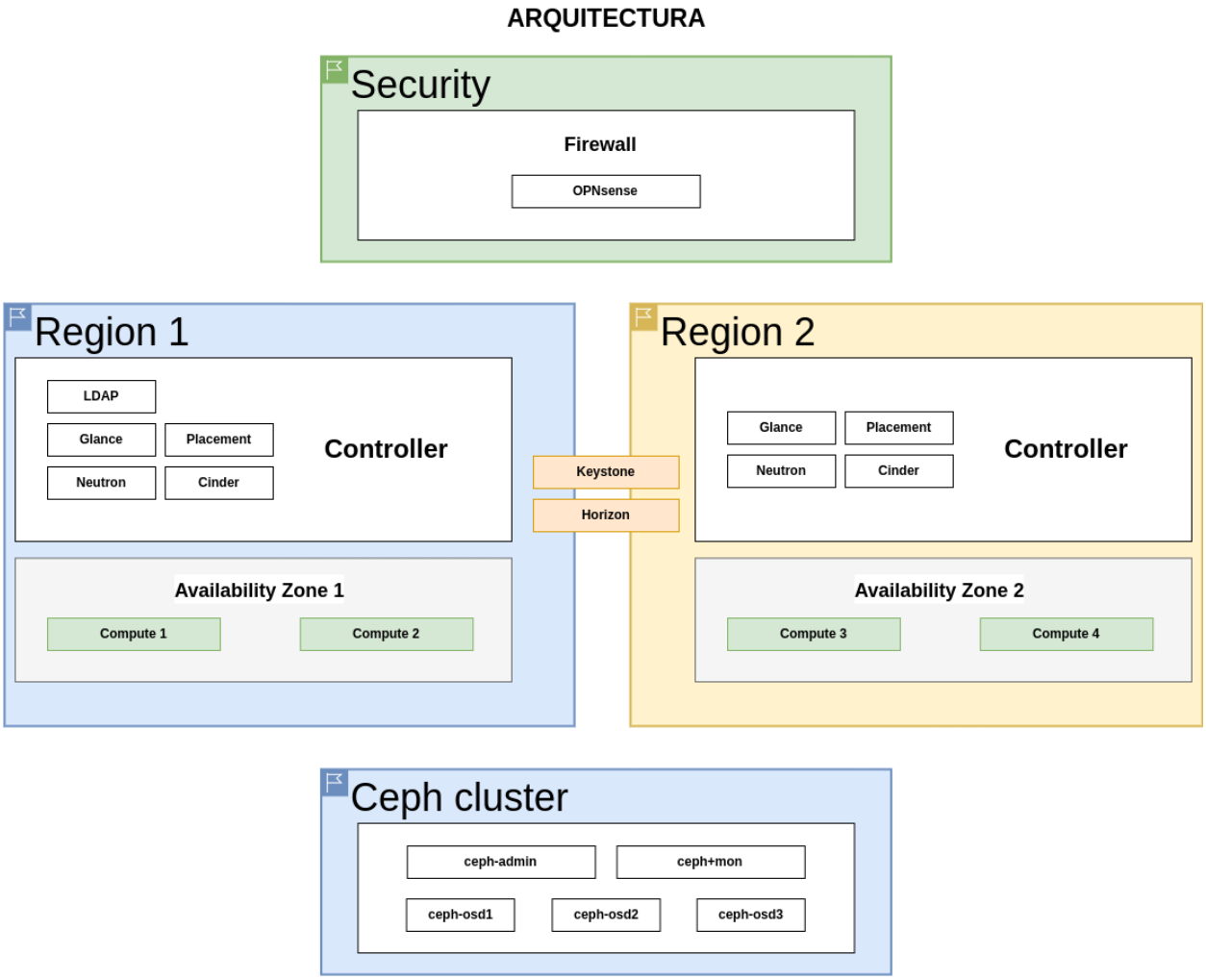


Figura 11: *Arquitectura de la Infraestructura.*

La arquitectura propuesta está diseñada para configurar una nube pública que ofrezca una amplia gama de servicios a empresas que se dedican al desarrollo de software, se utiliza OpenStack como una plataforma para proporcionar infraestructura en la nube. A continuación, se describen los componentes que conforman la arquitectura del prototipo.

7.1.1 Región 1 y Región 2

El diseño incluye dos regiones, lo que permite redundancia geográfica, alta disponibilidad y recuperación de datos ante fallos, asegurando que los servicios estén siempre disponibles.

7.1.1.1 Controller (Controlador)

El controlador es el núcleo que coordina y gestiona los diferentes servicios de la nube en cada región. A través de este nodo, los recursos de cómputo, almacenamiento y red se administran de manera eficiente. Los principales componentes de servicios en cada controller son:

Componente	Nombre	Descripcion
	**LDAP**	es un protocolo que permite el acceso a un directorio centralizado de usuarios y grupos. En esta arquitectura, se utiliza para gestionar la autenticación y los permisos de los usuarios en OpenStack.
	**Keystone**	es el servicio de autenticación e identidad en OpenStack, este servicio permite gestionar usuarios, proyectos y permisos. En esta arquitectura, Keystone es un servicio que está configurado en la región 1 y consumido por ambas regiones, ofreciendo un sistema de autenticación unificado para los usuarios.
	**Horizon**	es el panel de control web de OpenStack. A través de una interfaz gráfica, los usuarios pueden gestionar recursos de la nube

		como instancias, redes y almacenamiento de manera fácil e intuitiva. En esta arquitectura el servicio se ha configurado en la región 1 pero también es consumida por ambas regiones.
	**Glance**	Servicio de imágenes que permite el almacenamiento y la recuperación de estas, que son utilizadas para el despliegue de instancias dentro de OpenStack.
	**Placement**	Gestiona la asignación de recursos como CPU, memoria y almacenamiento entre las instancias. Es crucial para asegurar una distribución eficiente de los recursos disponibles en la infraestructura.
	**Neutron**	Es el servicio de redes en OpenStack, encargado de proporcionar conectividad entre las instancias. Permite crear redes virtuales, subredes, enrutamiento, y otros componentes de red necesarios para que las instancias puedan comunicarse entre sí o con el exterior.
	**Cinder**	Servicio de almacenamiento en bloque. Cinder permite a los usuarios crear y gestionar volúmenes de almacenamiento persistente que puedan ser adjuntados a las instancias, asegurando que los datos persistan más allá del ciclo de vida de una máquina virtual.

	**Trove**	Trove es el servicio que proporciona base de datos como servicio en OpenStack. Facilita el despliegue, operación y escalabilidad de bases de datos en la nube.
	**Nova**	Nova es el componente encargado de gestionar las máquinas virtuales en la nube. Coordina el ciclo de vida de las instancias, desde su creación hasta su destrucción, y se encarga de la asignación de recursos de cómputo.
	**Magnum**	Orquesta contenedores como servicio, permitiendo la gestión y despliegue de contenedores de aplicaciones de manera nativa en OpenStack.
	**Heat**	Es el motor de orquestación de OpenStack, que permite a los usuarios definir y desplegar infraestructuras complejas a través de plantillas.
	**Designate**	Este servicio proporciona gestión de DNS como servicio. Permite a los usuarios definir, gestionar zonas y registros DNS, asegurando que la instancias y servicios desplegados puedan ser accedidos mediante nombres de dominio personalizados.
	**Octavia**	Es el servicio de balanceo de carga de OpenStack. Proporciona balanceadores de carga escalables y redundantes para distribuir el tráfico, mejorando la disponibilidad y el rendimiento de aplicaciones desplegadas.

	**Yuyu**	Este servicio proporciona a los usuarios una visión clara de los costos asociados al uso de recursos en la nube. Está configurado para monitorear los servicios de nova, cinder, neutron y keystone para recopilar datos de uso de la infraestructura, ayudando a gestionar presupuestos y optimizar gastos.

7.1.1.2 Availability Zones

Zona de Disponibilidad 1 (Región 1) y Zona de Disponibilidad 2 (Región 2) Cada región cuenta con zonas de disponibilidad, que contienen los nodos de cómputo responsables de proveer recursos para las instancias de las máquinas virtuales. Los nodos de cómputo son utilizados para manejar las cargas de trabajo y el procesamiento de las aplicaciones desplegadas por los usuarios. Los nodos para esta arquitectura son:

- Compute 1 y Compute 2
- Compute 3 y Compute 4

Cada zona de disponibilidad asegura que los recursos estén aislados y distribuidos.

7.1.1.3 Ceph Cluster

El clúster de Ceph es una solución de almacenamiento distribuido que proporciona almacenamiento escalable y redundante. Está compuesto por los siguientes nodos:

- **ceph-admin:** Este componente centraliza la administración de todos los servicios del clúster Ceph, funcionando como el servidor principal para desplegar la

solución. Adicionalmente, ejecuta ceph-mgr, un servicio clave para monitorear el rendimiento y manejar la administración del clúster.

- **ceph-mon:** tiene la responsabilidad de gestionar un mapa detallado del clúster, que incluye la supervisión de los OSD y todos los elementos que forman parte del clúster.
- **Ceph-osd1, ceph-osd2, ceph-osd3:** Son los nodos de almacenamiento físico en Ceph, que se encargan de almacenar los datos reales. Estos nodos trabajan de manera redundante para garantizar que los datos siempre estén disponibles, incluso si algunos nodos fallan.

7.1.1.4 OpnSense

Como parte del prototipo de la plataforma de nube pública, se ha integrado un nodo adicional dedicado exclusivamente a la seguridad y administración de la red OPNsense. Este nodo funciona como firewall y gestor de red central, encargado de proteger la infraestructura y garantizar el flujo seguro de datos entre los diferentes servicios y usuarios.

7.2 Configuración

En esta sección, se detallan las especificaciones de hardware de los nodos que componen la infraestructura del prototipo para la solución de nube. Cada nodo juega un papel crucial en el funcionamiento de la arquitectura, ya sea como controlador, nodo de cómputo o parte del clúster de almacenamiento. Las configuraciones descritas a continuación incluyen información sobre el sistema operativo, los núcleos de CPU, la memoria RAM, el almacenamiento y las interfaces de red (NIC) de cada nodo, proporcionando una visión clara de los recursos asignados para asegurar un rendimiento óptimo y una gestión eficiente de las cargas de trabajo.

Se ha seleccionado la versión de Ubuntu 22.04 jammy debido que es compatible con OpenStack Bobcat la cual es una versión estable para el despliegue de la solución de este prototipo. Otra de las razones por la que se eligió esta versión de Ubuntu es debido al soporte extendido que esta posee facilitando el mantenimiento de la infraestructura.

7.2.1 Configuración nodo controller1

El hardware mínimo que se ha utilizado para el nodo controller 1 de este prototipo es el siguiente.

Tabla 1: Especificaciones de hardware en nodo controlador 1

Sistema Operativo	**CPU Núcleos**	**RAM**	**Almacenamiento**	**NIC**
Ubuntu 22.04 LTS	4	8 GB	100 GB	3

7.2.2 Configuración de nodo controller2

El nodo controller 2 de este prototipo ha sido implementado con el hardware mínimo siguiente.

Tabla 2: Especificaciones de hardware en nodo controlador2

Sistema Operativo	CPU Núcleos	RAM	Almacenamiento	NIC
Ubuntu 22.04 LTS	4	4 GB	100 GB	3

7.2.3 Configuración nodo compute1

El hardware mínimo que se ha utilizado para el nodo compute 1 para este prototipo es el que se describe a continuación.

Tabla 3: Especificaciones de hardware en nodo compute 1

Sistema Operativo	CPU Núcleos	RAM	Almacenamiento	NIC
Ubuntu 22.04 LTS	10	32 GB	100 GB	3

7.2.4 Configuración nodo compute2

A continuación, se detalla el hardware mínimo empleado para el nodo compute 2 de este prototipo.

Tabla 4: Especificaciones de hardware en nodo compute 2

Sistema Operativo	CPU Núcleos	RAM	Almacenamiento	NIC
Ubuntu 22.04 LTS	10	34 GB	100 GB	3

7.2.5 Configuración nodo compute3

El nodo compute 3 de este prototipo ha sido configurado con el siguiente hardware mínimo.

Tabla 5: Especificaciones de hardware en nodo compute3

Sistema Operativo	CPU Núcleos	RAM	Almacenamiento	NIC
Ubuntu 22.04 LTS	5	10 GB	100 GB	3

7.2.6 Configuración nodo compute4

El hardware mínimo requerido para el nodo compute 4 en este prototipo se describe a continuación.

Tabla 6: Especificaciones de hardware en nodo compute4

Sistema Operativo	CPU Núcleos	RAM	Almacenamiento	NIC
Ubuntu 22.04 LTS	5	12 GB	100 GB	3

7.2.7 Configuración Clúster de almacenamiento (ceph)

7.2.7.1 Ceph admin

El nodo ceph-admin de este prototipo utiliza el siguiente hardware mínimo.

Tabla 7: Especificaciones de hardware en nodo Ceph admin

Sistema Operativo	CPU Núcleos	RAM	Almacenamiento	NIC
Ubuntu 22.04 LTS	1	2 GB	100 GB	2

7.2.7.2 Ceph monitor

A continuación, se especifica el hardware mínimo asignado al nodo ceph-mon de este prototipo.

Tabla 8: Especificaciones de hardware en nodo Ceph monitor

Sistema Operativo	CPU Núcleos	RAM	Almacenamiento	NIC
Ubuntu 22.04 LTS	1	2 GB	100 GB	2

7.2.7.3 Ceph osd1

El siguiente es el hardware mínimo establecido para el nodo ceph-osd1 en este prototipo.

Tabla 9: Especificaciones de hardware en nodo Ceph osd1

Sistema Operativo	CPU Núcleos	RAM	Almacenamiento	OSD	NIC
Ubuntu 22.04 LTS	1	2 GB	100 GB	300 GB	2

7.2.7.4 Ceph osd2

Se detalla a continuación el hardware mínimo utilizado para el nodo ceph-osd2 de este prototipo.

Tabla 10: Especificaciones de hardware en nodo Ceph osd2

Sistema Operativo	CPU Núcleos	RAM	Almacenamiento	OSD	NIC
Ubuntu 22.04 LTS	1	2 GB	100 GB	300 GB	2

7.2.7.5 Ceph osd3

A continuación, se presenta el hardware mínimo empleado para el nodo ceph-osd3 en este prototipo.

Tabla 11: Especificaciones de hardware en nodo Ceph osd3

Sistema Operativo	CPU Núcleos	RAM	Almacenamiento	OSD	NIC
Ubuntu 22.04 LTS	1	2 GB	100 GB	300 GB	2

7.2.7.6 OPNsense

Tabla 12: Especificaciones de hardware en nodo OPNsense

Sistema Operativo	CPU Núcleos	RAM	Almacenamiento	NIC
Ubuntu 22.04 LTS	1	2 GB	40 GB	2

7.3 Topología de RED del prototipo

Este prototipo comprende dos regiones, cada una equipada con un nodo controlador que gestiona los servicios necesarios para construir y desplegar los stacks requeridos por las empresas, y con dos nodos de cómputo que aportan los recursos necesarios para las instancias. Además, se ha implementado un clúster de almacenamiento con cinco nodos, compuesto por un nodo administrador y un nodo de monitoreo, responsables de la administración del espacio en disco, la supervisión del estado de los discos y la gestión del espacio disponible. Los tres nodos restantes están destinados a ofrecer almacenamiento con réplica tres, garantizando la redundancia e integridad de la información.

El entorno cuenta con un nodo de firewall diseñado para proteger todas las redes y prevenir posibles brechas de seguridad, consolidando así una infraestructura robusta y segura. Finalmente, la red interna de cada región cuenta con un rango de direcciones IP que pertenecen a 10.0.0.0/24, la red virtual privada a la cual se conecta cada nodo de cada región cuenta con un rango de direcciones de 192.168.193.0/24 y todas la maquinas al estar dentro del entorno de kvm utilizan el rango de direcciones 192.168.122.0/24 lo cual les permite tener conexión a internet.

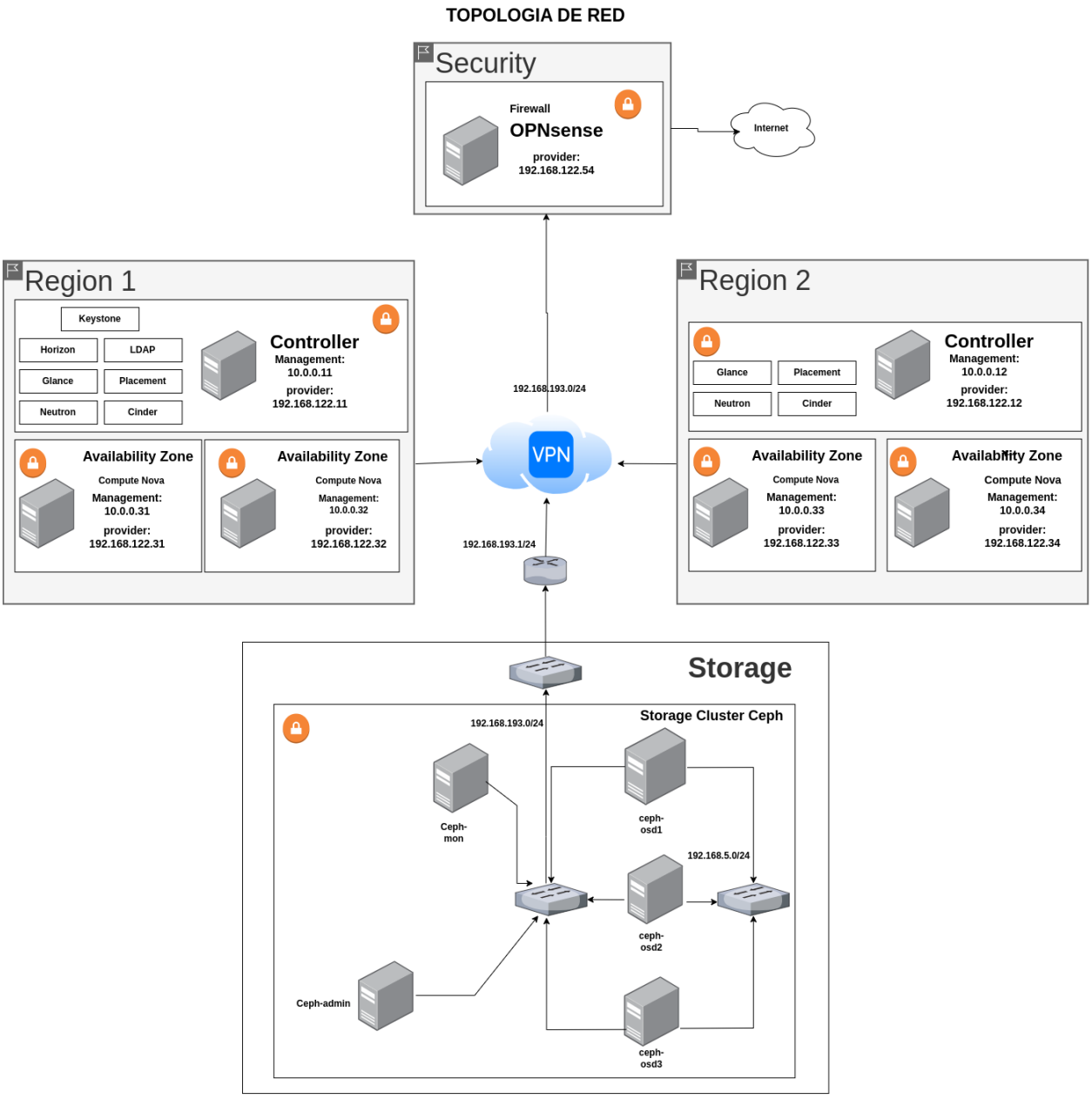


Figura 12: *Topología de RED del prototipo*

8 Caso de estudio

Las empresas dedicadas al desarrollo de software en el área metropolitana de El Salvador que están en creciente demanda de servicios tecnológicos por parte de sus clientes se ven en la necesidad de ofrecer servicios de alta disponibilidad, mantener la seguridad y la eficiencia de sistemas. El despliegue de una infraestructura de nube pública basada en OpenStack ofrece una solución flexible, escalable y segura para enfrentar los retos antes mencionados.

RAVN es una empresa de desarrollo de software en expansión que atiende a clientes de varias regiones de Centroamérica. Se especializa en desarrollar soluciones personalizadas para automatización de ventas, plataformas de comercio electrónico, y herramientas de gestión de inventarios, enfrenta crecientes demandas de sus clientes, lo que la lleva a requerir servicios de alta disponibilidad, seguridad y eficiencia operativa. Sin embargo, tiene dificultades para utilizar soluciones preconfiguradas que simplifiquen el desarrollo de aplicaciones personalizadas. Para satisfacer estas necesidades, este prototipo ofrece una infraestructura basada en OpenStack que incluye servicios esenciales como creación de instancias, carga de imágenes y almacenamiento, entre otros, Además, el prototipo proporciona stacks que facilitan el desarrollo al ofrecer plantillas preconfiguradas, como por ejemplo Spring Boot con MySQL, y WordPress con MySQL, junto con herramientas como Git, para la gestión de repositorios remotos. De esta manera, RAVN puede mejorar su capacidad de respuesta a las necesidades del mercado, optimizar su proceso de desarrollo y asegurar una operación eficiente y segura.

9 Factibilidad

La factibilidad de un proyecto es un aspecto esencial para garantizar su viabilidad antes de iniciar su desarrollo. Según la investigación realizada las empresas se inclinan al uso de la nube publica desde un punto de vista de rendimiento y financiero. Se busca rendimiento para agilizar sus procesos y tener soluciones a la mano que les facilite el desarrollo de sus actividades.

De igual forma buscan economizar en el uso de infraestructura, por eso se plantea hacer un análisis para poder ofrecer precios atractivos para que puedan aprovechar el rendimiento del prototipo al mejor precio. En el siguiente apartado, se evalúan las dimensiones técnica y económica del proyecto de creación de una plataforma de nube pública que responde a las necesidades específicas de empresas dedicadas al desarrollo de software.

9.1 Factibilidad Técnica

La factibilidad técnica es esencial para asegurar que la solución propuesta puede ser desplegada efectivamente. A continuación, se presenta una tabla que compara los componentes de OpenStack con los servicios equivalentes de proveedores de nube como AWS, Google Cloud y Azure, para proporcionar una comprensión de las opciones disponibles en el mercado:

Tabla 13: Comparación de componentes de OpenStack en AWS, Google Cloud y Azure

Componente	OpenStack	AWS	Google Cloud	Azure
Almacenamiento en bloque.	Cinder	EBS	Persistent Disk	Managed Disks
Redes	Neutron	VPC	Virtual Private Cloud	Virtual Network
Identidad	Keystone	IAM	Cloud IAM	Active Directory

Recursos	Placement	Resource Groups	Google Cloud Resource Manager	Resource Manager
Bases de datos	Trove	RDS	Cloud SQL	SQL Database
Máquinas virtuales	Nova	EC2	Compute Engine	Virtual Machines
Contenedores	Zun	Fargate	Run	Container Instances
Orquestación de servicios	Heat	CloudFormation	Deployment Manager	Resource Manager Templates
DNS	Designate	Route 53	Google cloud DNS	Azure DNS
Balanceo de carga	Octavia	Elastic Load Balancing	Load Balancing	Load Balancer
Monitoreo	Ceilometer	CloudWatch	Monitoring	Monitor
Autenticación	LDAP	Directory service	Identity	Directory
Almacenamiento de Objetos	Swift	S3	Storage	Blod Storage
Facturación y costos	Yuyu	AWS Billing and Cost Management	Google Cloud Billing	Management

Para identificar las fortalezas y debilidades se ha realizado una comparativa de las funcionalidades de infraestructura en la nube de código abierto, facilitando la toma de decisiones informadas para el despliegue de una infraestructura en la nube.

Tabla 14: Comparativa de plataformas de infraestructura en la nube de código abierto.

Funcionalidad	openstack.	cloudstack open source cloud computing	Open Nebula
Infraestructura de nube	Nubes privadas, públicas e híbridas	Nubes privadas, públicas e híbridas	Nubes privadas, públicas e híbridas
Infraestructura de recursos	Computación, almacenamiento y red	Computación, almacenamiento y red	Computación, almacenamiento y red
Arquitectura	Fragmentado en muchos módulos	Controlador central monolítico	Modular (componente de terceros)
Dificultad de instalación	Difícil (muchas opciones, no completamente automatizado)	Mediano (Pocas piezas para instalar)	Fácil (instaladores de paquetes basados en procesos)
Hipervisores soportados	Xen, KVM, VMware, HyperV, vCenter, LXC, vSphere	Xen, KVM, VMware, HyperV, vCenter, LXC, vSphere	Xen, KVM, VMware, vCenter
Administración	Web UI, CLI	Web UI, CLI	Web UI, CLI
Gestión de usuarios	Sí	Sí	Sí
Migración en vivo	Sí	Sí	Sí
Balanceo de carga	Sí	Sí	Sí
Tolerancia a fallos	Programación de máquinas virtuales, Replicación	Programación de máquinas virtuales, Replicación	Programación de máquinas virtuales, replicación
Alta disponibilidad	Sí	Sí	Sí
Seguridad	VPNs, firewall, Autenticación de usuario, otros	VPNs, firewall, Gestión de usuario, otros	Autenticación de usuario

Compatibilidad	Amazon EC2, Amazon S3	Amazon EC2, Amazon S3	Todas las interfaces de Amazon
Extensibilidad	Sí	Sí	Sí

La tabla 1 muestra plataformas de gestión de la nube con sus respectivas funcionalidades. El autor menciona que, para construir, implementar y gestionar infraestructuras en la nube, existen varias plataformas de gestión de la nube de código abierto disponibles para gestionar infraestructuras virtualizadas en las nubes (Aldossary, 2021b).

9.2 Factibilidad Económica

La implementación de una infraestructura de nube pública con OpenStack representa una decisión estratégica. Es esencial considerar los precios actuales de las empresas que otorgan servicios de nube pública, como AWS, Azure y Google Cloud estructuran sus ofertas, particularmente en lo que respecta a la venta de recursos de computación. Para garantizar una oferta competitiva, se ha realizado una investigación exhaustiva de los precios ofrecidos por estos proveedores, permitiendo así calcular nuestros propios precios como servicio de nube pública.

Los servicios de nube se presentan en forma de núcleos virtuales, donde las empresas comercializan la capacidad de procesamiento de manera que a menudo se asemeja a núcleos de CPU físicos. Sin embargo, es esencial aclarar que, cuando un proveedor menciona vCPUs, no necesariamente está ofreciendo un núcleo de procesador físico, sino que está virtualizando los recursos disponibles. En este sentido, la densidad de estos servicios puede variar significativamente entre proveedores.

Es importante destacar que un vCPU se refiere a un recurso virtual que puede compartir capacidad con otros vCPUs en el mismo servidor físico. Esto significa que los usuarios no siempre están recibiendo un núcleo dedicado, sino que pueden experimentar variaciones en el rendimiento dependiendo de la carga de trabajo de otros usuarios en el mismo entorno.

Para la comparación de precios con los proveedores mencionados, se han analizado varios factores clave que afectan el costo de los servicios de computación en la nube. Estos factores incluyen:

- Uso de recursos de cómputo en instancias, tales como:
    - Tipo de instancia basado en el sabor creado.
    - Número de vCPUs.
    - Memoria RAM (GiB).
- Volúmenes creados para las instancias.
- IP flotante utilizada para las instancias y routers.
- Snapshots generados a partir de instancias.
- Imágenes cloud subidas a la plataforma.

Esta comparativa nos permitirá establecer una estructura de precios competitiva y atractiva para los clientes interesados en utilizar nuestra nube pública basada en OpenStack. Se presenta a continuación las tablas comparativas de los diferentes servicios de las empresas que otorgan servicios de nube publica antes mencionadas.

Tabla 15: Precios estándar por cada servicio/ mensual

Servicio	**AWS**	**Azure**	**Google Cloud Platform**

Volumen/GB	$0.02	$0.15	$0.02
Floating IP	$03.60	$02.88	$02.88
Router	$36.00	$72.00	$36.00
SnapShot/GB	$0.05	$0.05	$0.05
Image (por núcleos)	$0.02	$0.02	$0.02

Nota: En el caso del servicio del flavor varía entre proveedores dependiendo de los recursos asignados, por consiguiente, no se maneja un precio estándar.

9.2.1 Precios Mensuales para Máquinas Virtuales (Instancias)

Tabla 16: Comparación de precios mensuales entre AWS, Google Cloud y Azure para diferentes configuraciones de instancias.

Sabor	AWS	Azure	Google Cloud Platform
2 CPU, 8 GB	$70.08	$70.08	$70.90
4 CPU, 16 GB	$140.16	$140.16	$141.79
8 CPU, 32 GB	$280.32	$280.32	$283.58
16 CPU, 64 GB	$560.64	$560.64	$567.17

9.2.2 Precios mensuales para almacenamiento en volúmenes

Tabla 17: Comparación de precios mensuales entre AWS, Google Cloud y Azure para almacenamiento en volúmenes.

Almacenamiento	AWS	Azure	Google Cloud Platform

Volumen/GB	$0.02	$0.15	$0.02

9.2.3 Precios mensuales y por hora IPs públicas

Los costos asociados a las direcciones IP son un aspecto crucial a considerar al evaluar la viabilidad económica en un prototipo de nube pública.

Tabla 18: Tabla comparativa de precios IPs públicas de diferentes proveedores

Proveedor	**Tipo de IP**	**Costo / hora**	**Costo / mes**
Google Cloud	IP Pública Estática	$ 0.004.00	$ 2.88
Azure	IP Pública Estática	$ 0.004.00	$ 2.88
Amazon Web Services	IP Elástica	$ 0.005.00	$ 3.60

9.2.4 Comparación de precios mensuales de Image

Tabla 19: Ejemplo de precios mensuales de almacenamiento de imágenes

Configuración	**AWS**	**Azure**	**Google Cloud Platform**
2 CPU (RHEL)	$21.35	$ 21.50	$ 21.02
4CPU (RHEL)	41.50	$ 40.01	$ 42.05
16 CPU (RHEL)	123.46	$ 121.07	$ 126.14

9.2.5 Precios mensuales y por horas routers

Los routers virtuales son componentes esenciales para la gestión y el enrutamiento del tráfico de datos. Los proveedores de servicios en la nube ofrecen distintos tipos de routers, que son

fundamentales para garantizar la conectividad y el rendimiento. A continuación, se detallan los costos asociados al uso de routers virtuales en los principales proveedores.

Tabla 20: Tabla comparativa de precios routers virtuales de diferentes proveedores

Proveedor	Tipo de router virtual	Costo por / hora	Costo por / mes	Costo por / GB
Google Cloud	Cloud Router	$ 0.05	$ 36.00	$ 0.02
Azure	Azure virtual Network Gateway	$ 0.10	$ 72.00	$ 0. 01
Amazon Web Services	Amazon VPC Router	$ 0.05	$ 36.00	$ 0.09

9.2.6 Comparación de precios mensuales de snapshots

Tabla 21: Ejemplo de precios mensuales de almacenamiento para snapshots

Almacenamiento	AWS	Azure	Google Cloud Platform
4 GB	$0.65	$0.60	$0.75
8 GB	$1.30	$1.20	$1.50
16 GB	$2.40	$2.40	$2.60
32 GB	$4.80	$5.0	$4.90

A continuación, se presentan los precios asignados a los servicios del prototipo de nube pública han sido cuidadosamente establecidos para ser competitivos en el mercado. Esta

competitividad se fundamenta en una exhaustiva investigación de los precios de servicios similares ofrecidos por los proveedores, lo que ha permitido posicionar tarifas de manera atractiva para los clientes. Además, la utilización de OpenStack permite operar con costos reducidos, lo que traduce en tarifas accesibles sin comprometerla calidad del servicio. Esta estructura de precios también contempla la oferta de paquetes flexibles que se adaptan a diversas necesidades, facilitando que los clientes elijan el plan que mejor se ajuste a sus requerimientos.

9.2.7 Hardware Utilizado

En esta sección, se detalla el tipo de hardware recomendado para el desarrollo del prototipo de la nube pública y los costos que implica tener este hardware funcionando en producción.

Para calcular la capacidad de almacenamiento y los costos de implementación de un clúster Ceph en una nube pública destinada a un entorno de producción se utilizó una herramienta en línea disponible en un sitio web. Al especificar el número de hosts, discos por host, tamaño de disco, costo por disco y el factor de replicación, es posible estimar los recursos necesarios para ofrecer una solución de almacenamiento confiable y escalable en un ambiente de nube pública. Esta estimación ayuda a garantizar que el clúster Ceph cumpla con los requisitos de capacidad y redundancia adecuados para un entorno de producción.

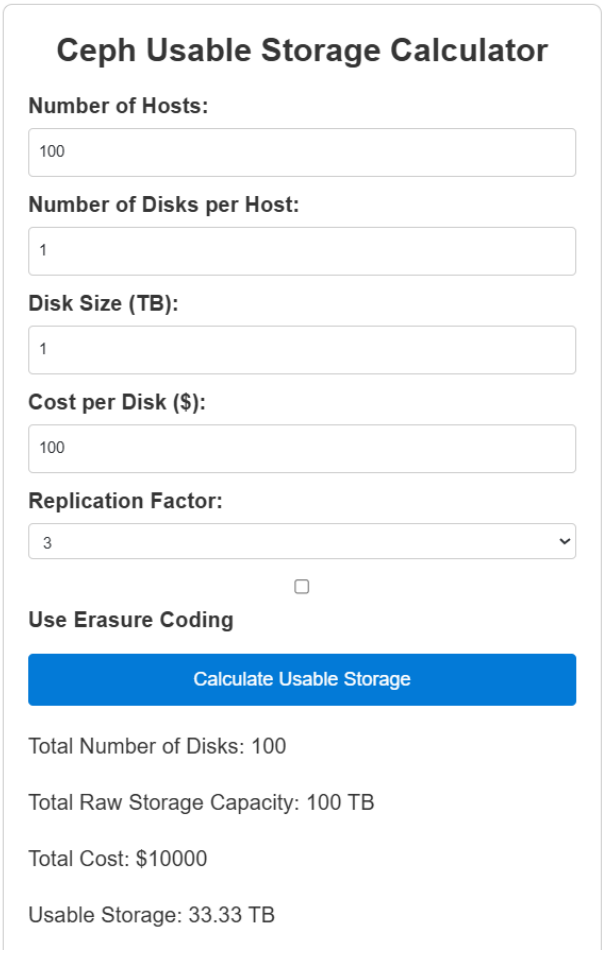


Figura 13: *Calculadora Ceph para el cálculo de uso en base a número de hosts. (Lee & Lee, 2024)*

A continuación, se presentan las especificaciones de los servidores propuestos para entornos de producción, entre ellos nodos de cómputo, controller, nodo firewall y nodos ceph. Estos servidores están diseñados para ofrecer un rendimiento óptimo en entornos de producción, garantizando una adecuada capacidad de procesamiento y almacenamiento.

Tabla 22: Especificaciones para los nodos compute.

Nombre	CPU	Subprocesos	Almacenamiento	RAM
Smart Selection PowerEdge R7625 Servidor Rack (DELL)	2	128	480GB SSD	768GB

Tabla 23: Especificaciones para los nodos controller.

Nombre	CPU	Subprocesos	Almacenamiento	RAM
Smart Value PowerEdge R740 Server Optimal (DELL)	10	20	1.92TB SSD	64

Tabla 24: Especificaciones para los nodos admin y monitor ceph.

Nombre	CPU	Subprocesos	Almacenamiento	RAM
Smart Value PowerEdge R740 Server Optimal (DELL)	10	20	480GB SSD	64

Tabla 25: Especificaciones para los nodos ceph-osd.

Nombre	CPU	Subprocesos	Almacenamiento	RAM
Smart Selection PowerEdge R750 Rack Servidor (DELL)	12	24	100TB SSD	64

Tabla 26: Especificaciones para el nodo firewall.

Nombre	CPU	Subprocesos	Almacenamiento	RAM

Smart Value PowerEdge R740 Server Optimal (DELL)	10	20		480GB SSD	64

9.2.8 Evaluación de la Capacidad de Hardware para Desplegar Stacks

Evaluación de las especificaciones de hardware y su capacidad para soportar múltiples stacks de templates de Heat. Incluyendo el cálculo de la cantidad máxima de unidades de stacks de WordPress, Laravel, etc. que se pueden ejecutar en el hardware de producción.

A continuación, se calcula el número de stacks a lanzar en cuanto a capacidad de hardware y numero de servidores en nuestro data center sin aplicación de relación a los hilos de los CPU en cuanto a distribución de cargas de trabajo simultaneas entre múltiples usuarios. Es importante mencionar que los posibles stacks que se lanzaran son Wordpress stack, Laravel stack, Node JS stack, Spring boot stack y Django stack, etc.:

Tabla 27: Número de stacks a lanzar en base al hardware del data center.

Stack	Memoria RAM disponible	Número de subprocesos disponibles	Memoria RAM por MV	Subproceso por stack	Cant. De Máquinas virtuales
Unidad de stack	43,008 GB	7,168 CPU	4 GB	2 CPU	3,584 MV

Sin realizar una relación a los hilos de los CPU o sin distribuir cargas de trabajo simultaneas entre varios usuarios, podemos ver que el hardware de producción es capaz de lanzar 3,584 máquinas virtuales con los recursos de 2 CPU, 4 GB RAM y 25GB por stack de almacenamiento en Cluster Ceph.

Para un óptimo uso y aprovechamiento del hardware de producción se aplica una relación de 1:3 para multiplicar el número de hilos de cada subproceso disponible en nuestro hardware de nodos compute disponibles. Esto posibilita llegar a más usuarios con el mismo CPU convirtiendo estos en múltiples VCPU sin comprometer el performance de la máquina virtual ofrecida al cliente para el despliegue de sus aplicaciones.

Tabla 28: Número de stacks a lanzar en base a una relación de 1:3 en el hardware del data center.

Stack	Memoria RAM disponible	Número de subprocesos disponibles	Memoria RAM por MV	VCPU por stack	Cant. De Máquinas virtuales
Unidad de stack	43,008 GB	21,504 VCPU	4 GB	2 VCPU	10,752 MV

Los datos anteriores muestran que el hardware del data center está optimizado para la relación de **1:3**, maximizando la utilización de recursos disponibles. La capacidad de lanzar **10,752 máquinas virtuales** asegura flexibilidad para alojar múltiples stacks (como Laravel o WordPress) sin comprometer el rendimiento de la máquina virtual.

A continuación, se presenta la tabla de precios y especificaciones del hardware de producción propuesto.

Tabla 29: Especificaciones técnicas y precios de hardware de producción.

Nombre	CPU	Hilos	SSD	RAM	Precio unit.	Cant.	Monto

Smart Selection PowerEdge R7625 Servidor Rack	2	128	480GB	768GB	$10,025.98	56	$561,454.88
Smart Value PowerEdge R740 Server Optimal	10	20	1.92TB	64	$4,646.96	2	$9,293.92
Smart Value PowerEdge R740 Server Optimal	10	20	480GB	64	$3,619.12	3	$10,857.36
Smart Selection PowerEdge R750 Rack Servidor	12	24	100TB	64	$21,283.46	3	$63,850.38
Total							**$645,456.54**

El Costo total del hardware de servidores propuesto para producción es de **$645,456.54.**

9.2.9 Diseño y Costos de Infraestructura de Data Center

La infraestructura de un data center es fundamental para garantizar la operación eficiente y segura de servicios en la nube. En este apartado, se analizan los elementos necesarios para el montaje de un data center que soporte la infraestructura de nube pública, incluyendo aspectos críticos como espacio físico, sistemas de enfriamiento, distribución de energía, y medidas de seguridad. Esta sección busca proporcionar una visión integral de los requerimientos técnicos y financieros necesarios para establecer un entorno de data center adecuado y rentable.

- **Requerimientos de Espacio y Estructura del Data Center**:

- Área total a alquilar (1226 m²).
- Diseño estructural que incluye piso elevado y cielo falso para la optimización del flujo de aire y accesibilidad.

- **Sistemas de Enfriamiento y Distribución de Aire**:
  - **Sistema de Enfriamiento**: Detalle de la configuración del CRAH con chiller/tower de VFD, así como el tipo de distribución de aire (enfriamiento perimetral sin contención).
  - **Análisis de Costos**: Sistemas de enfriamiento con base en opciones como enfriamiento perimetral.
- **Sistemas de Energía y Redundancia**:
  - **Arquitectura de UPS**: Descripción de la arquitectura de UPS tradicional no escalable, especificando capacidad y características.
  - **Generador de Respaldo**: Especificación de capacidades de energía de generadores.
- **Sistemas de Protección y Seguridad**:
  - **Supresión y Detección de Incendios**: Tipo de sistemas de detección y extinción de incendios adaptados al entorno del data center.
  - **Distribución Eléctrica y Tableros**: Panelboards y distribución de energía para asegurar la continuidad y seguridad en el servicio.
- **Equipamiento Adicional e Iluminación**:
  - **Racks**: Cotización y especificaciones de racks necesarios.
  - **Iluminación**: Plan de iluminación LED especializado para data centers, que optimice el consumo energético y asegure visibilidad adecuada.

Tabla 30: Precios de elementos necesarios en el montaje de un data center.

Componente	Descripción	Cantidad	Costo Unit.	Monto
Sistema de enfriamiento	Suministro e instalación de aire de precisión para data Center 12 KW de capacidad de enfriamiento.	1	$20,355.00	$20,355.00
Distribución de aire		8	$5,000.00	$40,000.00
Arquitectura de UPS	Visench +Power Modular OEM UPS 20KVA 90KVA 120KVA 180KVA 00KVA 210KVA 300KVA PF 0.99 3 phase Ups Power Supply croOnline Ups	8	$2,721	$21,768.00
Racks	APC NetShelter SX, Server Rack Enclosure, 42U, Black, 1991H x 600W x 1070D mm. Capacidad de Albergue de 20 servers.	3	$1,875.00	$5,625.00
Generador de reserva	Generador Eléctrico Diesel 100KW/ 125 KvA. 400/230 V. Hyundai 82HY125CH	1	$20,049.990	$20,049.990
Piso elevado	Instalación de piso elevado para Data Center	1	$4,320.00	$4,320.00

Sistema de detección de incendios	*Detectores de humo, detectores de calor, detectores de llama*	1	$1,400.00	$1,400.00
Sistema de supresión de incendios	Sistemas de rociadores, sistemas de extinción por gas (FM-200, CO2)	1	$18,000.00	$18,000.00
Infraestructura de protección contra incendios	Construcción resistente al fuego Materiales, puertas cortafuegos, barreras	1	$14,000.00	$14,000.00
Tableros/Paneles	*Los tableros eléctricos se utilizan para transmitir energía a una o más fuentes.*	1	$11,090.00	$11,090.00

Tabla 31: Salario personal

Perfil	Responsabilidad	Salario Mensual
Administrador de Infraestructura	Gestión de infraestructura física y virtual, monitoreo de recursos, optimización del rendimiento	$ 1,800.00
Especialista en Seguridad	Implementación de seguridad, control de accesos, monitoreo de amenazas.	$2,200.00
Ingeniero de Redes	Configuración y mantenimiento de redes, gestión de conectividad, configuración de VPNs.	$2,000.00

Administrador de bases de Datos	Gestión de bases de datos, optimización de rendimiento, backup y recuperación, seguridad en DB.	$2,000.00
Gerente	Supervisión del ciclo de vida del producto. Análisis de mercado	$3,000.00
Arquitecto en la nube	Diseño de arquitectura en la nube, integración de soluciones, planificación de escalabilidad.	$2,000.00

Tabla 32: Gastos fijos mensuales

Concepto	Costo (USD)
Alquiler	$2,260.00
Agua	$300.00
Energía Eléctrica	$12,600.00
Mantenimiento hardware	$8,000.00
Mantenimiento de sistema de incendios	$ 2,000.00
Total, Costos fijos	$25,160.00

Se hace el cálculo de costos fijos para el año realizando la multiplicación del total de costos fijos x los 12 meses del año.

Total, costos fijos = $25,160 x 12

Total, costos fijos = $ 301,920.00

9.2.10 Tabla de Costos Detallada

Tabla 33: Costos totales

Concepto	Costo
Unidades de máquinas virtuales a crear	10,752 máquinas virtuales
Costo de Servidores	$ 645,456.54
Costos de elementos necesarios	$ 156,607.99
Costo de Recurso humano	$ 156,000
Gastos fijos	$ 301,920
Costo total	**$ 1,259,984.53**

9.2.11 Cálculo del Costo Unitario

Se presenta el cálculo del costo unitario dividiendo el costo total entre las unidades de máquinas virtuales. Se ha utilizado la siguiente formula.

Costo total Unitario = Costo total de producción / Unidades de máquinas virtuales a crear

Costo total Unitario = $ 1,259,984.53 / 10,752 = $ 117.19

Costo maquina al mes = Costo total unitario / 12

Costo maquina al mes = $ 117.19 /12

Costo maquina al mes = $ 9.77

9.2.12 Costos por máquina virtual

En entornos virtualizados se optó por una relación de 1:3 para el aprovechamiento del hardware de producción de manera optimizada, lo que implica que un VCPU se distribuye en

cargas de trabajo simultaneas entre 3 usuarios, siendo esta una relación moderada que permite dar un servicio optimizado sin afectar el rendimiento de la máquina virtual.

Se han calculado los costos asociados para máquinas destinadas a 3 usuarios.

Tabla 34: Costo anual y mensual del stack

Concepto	Costo anual	Costo mensual
Máquina virtual	$ 117.78	$ 9.77

9.2.13 Precio de venta

Se ha considera un margen de ganancia del 50% por cada máquina, el siguiente cálculo anual muestra como se ha considerado el precio.

Precio de venta = costo máquina virtual + (costo máquina virtual * margen de ganancia)

Precio de venta = $117.19 + $58.59

Precio de venta = $175.78 anual

Precio de venta = $175.78 / 12

Precio de venta = $ 14.65 mensual.

Tabla 35: Precio de máquina virtual por mes y año

Concepto	Precio de venta mensual	Precio de venta anual
Precio por máquina	$ 14.65	$ 175.78

9.2.14 Estudio de Rentabilidad y Costo-Beneficio

Ingreso total = Precio de venta anual x Numero de máquinas al año

Ingreso total = $175.78 x 2150

Ingreso total = $377,925.05

Ingreso total anual = $377,925.05 x 5

Ingreso total en 5 años = $1,889,625.24

Beneficio = Ingreso total anual – Costo de producción

Beneficio = $1,889,625.24 - $1,259,984.53

Beneficio = $629,640.71

El análisis anterior muestra que el proyecto es factible desde un punto de vista económico. Para asegurar la eficiencia se configurado una relación de 1 a 3 por VCPU, lo que asegura el aprovechamiento del hardware para la generación de ingresos. Además, el margen de ganancia del 50% esto asegura una buena rentabilidad y la proyección muestra una ganancia significativa de $629,640.71 en 5 años, equivalente a un promedio de $125,928.14 por año. Este nivel de beneficio es considerable y justifica la inversión inicial de $802,064.53.

10 Conclusiones

- ✓ La implementación de una infraestructura de nube pública basada en Openstack y orientada al desarrollo de software destacó la importancia de crear un entorno que vaya más allá de la simple provisión de recursos en la nube. Es fundamental garantizar la disponibilidad y seguridad de los datos, lo cual refleja la creciente necesidad en el mercado de soluciones en la nube que sean no solo funcionales, sino también altamente seguras y confiables para las organizaciones de software.
- ✓ La capacidad de gestionar la infraestructura de manera centralizada, utilizando una interfaz unificada, se identificó como un componente clave para el éxito del prototipo. Esta gestión centralizada resulta indispensable para supervisar y coordinar el entorno de desarrollo de software, permitiendo a los administradores tener una visión integral y facilitar la operación eficiente de múltiples recursos y servicios distribuidos.
- ✓ El proyecto abordó con éxito uno de los desafíos más importantes en la creación de infraestructuras en la nube: el desarrollo de un sistema capaz de soportar múltiples entornos de trabajo distribuidos geográficamente. La capacidad de manejar cargas de trabajo de desarrollo desde diferentes ubicaciones o zonas de disponibilidad es clave, y este prototipo demostró la viabilidad de una solución flexible y escalable para organizaciones.
- ✓ Se concluyó que una autenticación robusta y la gestión eficiente de sesiones son elementos críticos para garantizar un entorno seguro y efectivo. Estas características resultan esenciales para asegurar que solo usuarios autorizados puedan acceder a los recursos adecuados, permitiendo que las organizaciones de desarrollo operen de manera protegida y fluida.
- ✓ Los VCPU que ofrecemos a nuestros clientes son recursos de cómputo virtualizados, lo que permite una gestión eficiente y flexible de la infraestructura. Al ser virtuales, estos VCPU

permiten a los clientes escalar su capacidad de procesamiento según sus necesidades, optimizando costos y recursos sin requerir inversiones de hardware físico. Esta arquitectura proporciona a los clientes la flexibilidad necesaria para adaptarse a un entorno tecnológico dinámico.

11 Recomendaciones

- ✓ Recibir Retroalimentación de los Desarrolladores: Es importante recopilar de manera continua las opiniones y sugerencias de los usuarios y desarrolladores para ajustar la infraestructura a sus flujos de trabajo y mejorar su rendimiento y eficiencia en el entorno de desarrollo y pruebas.
- ✓ Monitoreo Constante del Desempeño y Capacidad: A lo largo del desarrollo de la infraestructura, es importante llevar a cabo revisiones periódicas de su desempeño, escalabilidad y capacidad, asegurando que se ajusten a las necesidades evolutivas de las organizaciones dedicadas al desarrollo de software.
- ✓ Expansión y Adaptación a Futuro: Si bien se realizarán pruebas de escalabilidad en la infraestructura, es esencial prever cómo podrá ajustarse y crecer en el futuro para cubrir las necesidades de las organizaciones de desarrollo de software a medida que aumenten sus demandas. Esto incluye planificar la expansión de recursos computacionales, almacenamiento, y garantizar que el rendimiento se mantenga óptimo a medida que se incremente la cantidad de usuarios y proyectos.
- ✓ Programas de Formación en Tecnología de la Nube: Para maximizar el uso de la infraestructura, ofrecer programas de formación para que los desarrolladores y administradores es una opción inteligente, ya que los desarrolladores de las empresas cliente pueden familiarizarse con las tecnologías de nube y las metodologías DevOps con mayor facilidad.

12 Limitaciones

La falta de acceso a todas las empresas identificadas para el estudio. Aunque se intentó obtener respuestas de las 97 empresas relacionadas al sector, no fue posible establecer contacto con todas ellas. En su lugar, los datos obtenidos provienen de colaboradores cercanos o amistades en de los investigadores en las empresas, lo cual pudo influir en la representatividad de los resultados. Esta limitación podría haber reducido la diversidad de perspectivas y alcance general de las conclusiones.

Una limitación adicional de esta investigación fue la falta de respuesta por parte de los líderes de los equipos de desarrollo. Aunque se diseñaron dos encuestas, una dirigida a los analistas programadores y otra a los líderes de equipo, solo se obtuvo respuesta de los primeros.

13 Referencias bibliográficas

Aldossary, M. (2021a). A Review of Dynamic Resource Management in Cloud Computing Environments. *Computer Systems Science and Engineering*, *36*(3), 461-476. https://doi.org/10.32604/csse.2021.014975

Corbetta, P. (2007). Metodología y técnicas de investigación social.

Carranza, C. P. M. (2023, 15 julio). Diseño y desarrollo de instrumentos de proyectos de investigación tecnológica [Diapositivas]. Recuperado de https://es.slideshare.net/ChristianPaoloMartel/diseo-y-desarrollo-de-instrumentos-de-proyectos-de-investigacin-tecnolgica-259230227

Companies & Reviews | Glassdoor. (s. f.). Recuperado 8 de septiembre de 2024, de https://www.glassdoor.com/Reviews/index.htm?overall_rating_low=1&page=1&locId=2254008&locType=C&locName=San%20Salvador%20(El%20Salvador)§or=10013&filterType=RATING_OVERALL

Containers explained: What they are and why you should care. (s. f.). Recuperado de https://www.redhat.com/en/topics/containers

Cuadro comparativo - Método científico y Método de la Ingeniería 2FB. (2017, 3 febrero). Recuperado de https://issuu.com/silvanav12/docs/cuadro_de_comparativo_silvana_termi_50d22f2d5c05f8

CloudStack's Documentation. (s. f.). Recuperado de https://docs.cloudstack.apache.org/en/latest/index.html

Empresas de informática y software en San Salvador. (s. f.). Recuperado 8 de septiembre de 2024, de https://sv.computrabajo.com/empresas/empresas-de-informatica-y-software-en-san-salvador-en-san-salvador

Introducción a OpenStack. (2024). Recuperado de https://docs.openstack.org/id/security-guide/introduction/introduction-to-openstack.html

Lee, B., & Lee, B. (2024, 24 septiembre). Ceph Storage Calculator to find Capacity and Cost. Recuperado de https://www.virtualizationhowto.com/2024/09/ceph-storage-calculator-to-find-capacity-and-cost/

Modelos de servicio en la nube | Tipos de cloud computing | AWS. (2024). Amazon Web Services, Inc. Recuperado de https://aws.amazon.com/es/types-of-cloud-computing/

Máquinas virtuales vs contenedores. (devskillbuilder, 2024). Recuperado de https://www.devskillbuilder.com

OpenNebula. (2024, 7 agosto). OpenNebula – Open-Source Cloud & Edge Computing Platform. Recuperado de https://opennebula.io/

OpenNebula Overview - Dataset. (2024, 7 agosto). Recuperado de https://support.opennebula.pro/hc/en-us/article_attachments/7124526151825

educaOpen. (s.f). Historia de la nube: desde los años 50 hasta nuestros días. Recuperado de https://www.educaopen.com/digital-lab/blog/software/historia-de-la-nube

¿Qué es el SaaS? - Explicación del software como servicio - AWS. (s. f.). Recuperado de https://aws.amazon.com/es/what-is/saas

¿Qué es la IaaS? (Infraestructura como servicio) | *IBM*. (2024). Recuperado de https://www.ibm.com/mx-es/topics/iaas

¿Qué es la IaaS? Explicación de la infraestructura como servicio: AWS. (2024). Recuperado de https://aws.amazon.com/es/what-is/iaas/

¿Qué es PaaS? Plataforma como servicio | Microsoft Azure. (2024). Recuperado de https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-paas

¿Qué es la plataforma como servicio (PaaS)? | IBM. (2024). Recuperado de https://www.ibm.com/mx-es/topics/paas

¿Qué es la plataforma como servicio (PaaS) y por qué utilizarla? (2024). Recuperado de https://www.salesforce.com/es/learning-centre/tech/paas/

¿Qué es una nube pública? | IBM. (s. f.). Recuperado de https://n9.cl/c62kv

¿Qué es OpenStack? (2024). Recuperado de https://www.rackspace.com/es/library/what-is-openstack

Téllez Valdés, J. (2013). *Computo en la Nube* [Electrónico]. Instituto de Investigaciones Jurídicas, Universidad Nacional Autónoma de México. Recuperado de https://archivos.juridicas.unam.mx/www/bjv/libros/7/3249/3.pdf

TeamNet. (2018). *¿Qué es la virtualización?* Recuperado de https://www.teanmet.com/virtualizacion

Tipos de Hypervisores (devopspace, 2024). Recuperado de https://www.devopsage.com/difference-between-type-1-and-type-2-hypervisor

Understanding virtualization. (s. f.-b). Recuperado de https://www.redhat.com/en/topics/virtualization

Welcome to Apache CloudStack's Documentation. (s. f.). Recuperado de https://docs.cloudstack.apache.org/en/latest/index.html

Wang, L., Ranjan, R., Chen, J., & Benatallah, B. (Eds.). (2011). *Cloud computing: Methodology, systems, and applications* (1st ed.). Taylor & Francis Group.

14 ANEXOS

- ANEXO 1: Encuesta
- ANEXO 2: Documento HLD
- ANEXO 3: Documento LLD

ANEXO 1: Encuesta

El objetivo de esta encuesta es recopilar información sobre el uso y adopción de servicios de nube en organizaciones tecnológicas. Estamos interesados en comprender qué modelos de nube se utilizan, cuáles son las necesidades tecnológicas de su organización y cómo lo servicios de nube pública pueden mejorar las operaciones.

Instrucciones: Por favor, responda a todas las preguntas según la situación actual de su organización.

* Indica que la pregunta es obligatoria

1. Correo *

---

2. ¿Cuál es el tamaño de la organización donde trabaja? *
    - Microempresa (1 - 10 empleados)
    - Pequeña empresa (11 - 50 empleados)
    - Mediana empresa (51 - 250 empleados)
    - Grande (más de 250 empleados)
3. ¿Qué modelo de nube utiliza actualmente su organización? puede seleccionar más de una respuesta.

    Selecciona todos los que correspondan.

    - On-premise
    - Nube pública
    - Nube privada
    - Nube Híbrida
    - Ninguna
    - Otro:

4. ¿Con cuál proveedor de servicios de nube trabaja actualmente su empresa para el desarrollo o implementación de sus soluciones tecnológicas? puede seleccionar más de una respuesta

   Selecciona todos los que correspondan.

   - Amazon Web Services (AWS)
   - Google Cloud
   - Microsoft Azure
   - Oracle Cloud
   - Ninguna
   - Otro:

5. ¿El manejo de recursos de infraestructura es variable dentro de la organización?

   Marca solo un óvalo

   - Sí
   - No

6. ¿Cuáles son las principales necesidades tecnológicas en la organización? puede seleccionar más de una respuesta

   Selecciona todos los que correspondan.

   - Almacenamiento y gestión de datos
   - Procesamiento de grandes volúmenes de datos
   - Seguridad y cumplimento normativo
   - Implementación de aplicaciones (DevOps)

7. ¿Cuáles son los mayores desafíos que enfrenta el equipo de desarrollo al utilizar* servicios de nube pública? puede seleccionar más de una respuesta

Selecciona todos los que correspondan.

- Complejidad en la configuración y administración
- Falta de conocimientos técnicos específicos
- Integración con sistemas existentes
- Problemas de seguridad y cumplimiento
- Rendimiento o latencia de las aplicaciones

8. ¿Cuál es el principal factor que influye en la decisión de su organización para * cambiar de proveedor de nube pública? puede seleccionar más de una respuesta

- Mejora en los servicios ofrecidos
- Reducción de costos
- Mejor soporte y servicio al cliente
- Nuevas características y capacidades
- Problemas de rendimiento o confiabilidad

ANEXO 2: Documento HLD

UNIVERSIDAD DE EL SALVADOR

FACULTAD DE INGENIERÍA Y ARQUITECTURA

DOCUMENTO DE DISEÑO DE ALTO NIVEL

HIGH LEVEL DESIGN DOCUMENT

NOVIEMBRE 2024

VERSIÓN 1.0 17/11/2024

INDICE

INTRODUCCIÓN

La computación en la nube ha revolucionado la forma en que las empresas gestionan sus recursos tecnológicos, ofreciendo soluciones que permiten la escalabilidad, la eficiencia de costos y agilidad operativa. En este documento se detalla el diseño de una plataforma de nube pública que proporciona un conjunto de servicios virtualizados, específicamente orientados a satisfacer las necesidades de empresas que se dedican al desarrollo de software en El Salvador.

El prototipo de la infraestructura se basa en OpenStack, una solución de código abierto que permite el despliegue de servicios de nube de manera flexible y escalable. La arquitectura de la plataforma está diseñada para garantizar alta disponibilidad y eficiencia en el uso de recursos. Se compone de al menos dos regiones, cada una con controladores que gestionan diversos servicios como la identidad, almacenamiento, redes entre otros, las zonas incluyen nodos de cómputo que gestionan el procesamiento de las aplicaciones y servicios.

Además, se incorporar un clúster de Ceph para el almacenamiento distribuido, lo que proporciona un sistema de almacenamiento robusto y confiable. Esto permite a las organizaciones acceder al recurso de manera eficiente y segura, facilitando el despliegue y la gestión de aplicaciones, así como la gestión de datos.

OBJETIVOS

GENERAL

Describir la arquitectura de un prototipo de nube pública basado en OpenStack, resaltando sus beneficios para empresas dedicadas al desarrollo de software

ESPECIFICOS

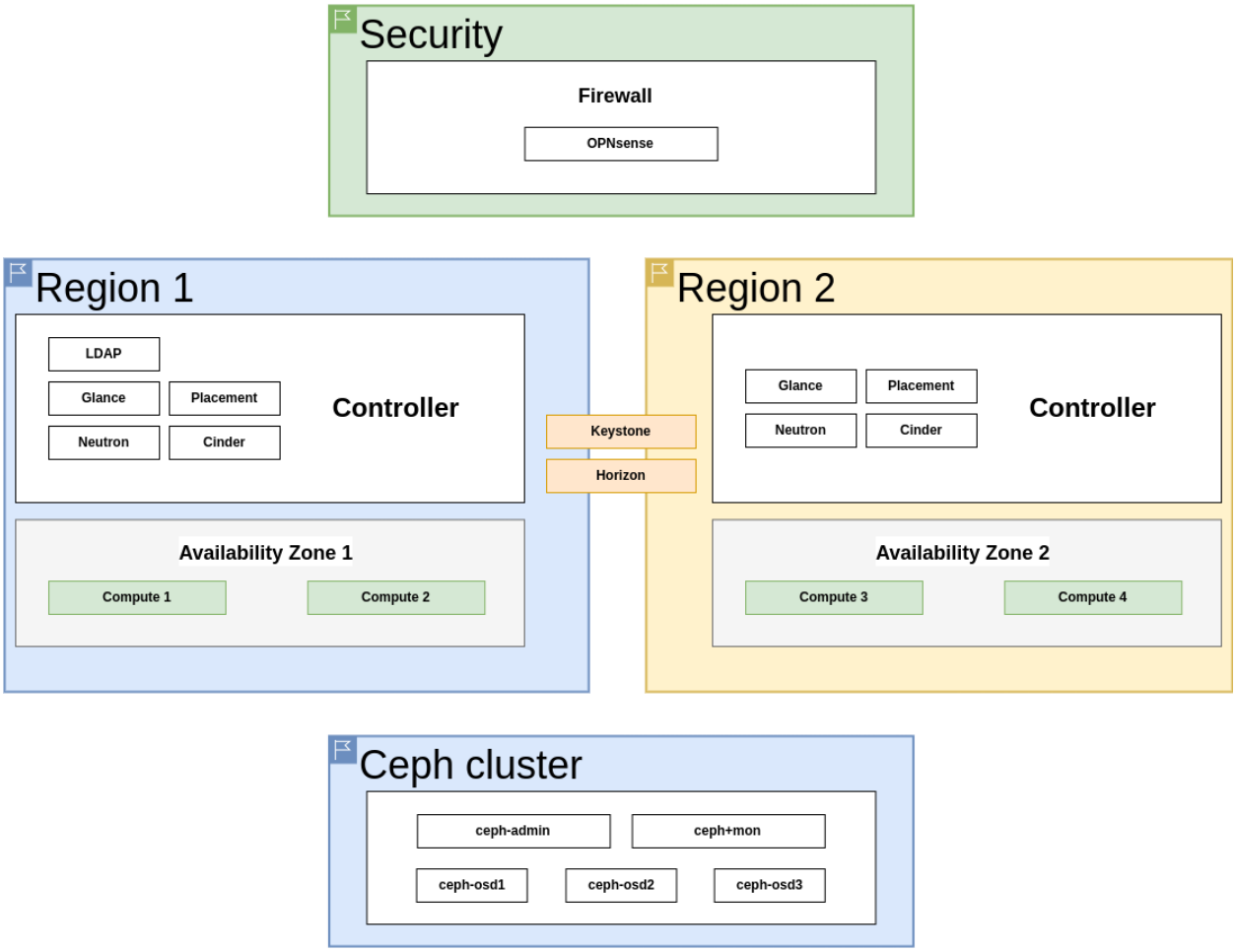
Detallar la disposición de los componentes clave como controladores, almacenamiento, compute en las diferentes regiones de la arquitectura.

Explicar las interacciones entre los servicios de nube, como Cinder, Neutron, Keystone, Nova y otros, su rol en la gestión de recursos y la seguridad del sistema.

Documentar el diseño de almacenamiento Ceph, destacando cómo se asegura la disponibilidad, redundancia y protección de los datos almacenados.

INFRAESTRUCTURA

ARQUITECTURA



La arquitectura está diseñada para implementar una plataforma de nube pública utilizando OpenStack, orientada a satisfacer las necesidades tecnológicas de empresas que se dedican al desarrollo de software. Este diseño asegura la disponibilidad y redundancia de los servicios críticos. Ambas regiones están interconectadas a través de los servicios de Keystone y Horizon, lo que facilita la gestión centralizada de la identidad y el acceso de los usuarios, así como la interfaz gráfica para administrar los recursos. Esta configuración asegura que, en caso de fallas en una región, los servicios puedan ser mantenidos en la otra sin interrupciones significativas.

El controlador es el nodo central en cada región, encargado de gestionar los servicios y coordinar la asignación eficiente de los recursos de cómputo, red y almacenamiento. A continuación, se describen los principales servicios en cada región:

LDAP protocolo utilizado para gestionar la autenticación y permisos de los usuarios de manera centralizada. Facilita el acceso seguro a los servicios de OpenStack mediante un directorio unificado de usuarios y grupos.

Keystone servicio de autenticación e identidad de OpenStack, responsable de la gestión de usuarios, proyectos y permisos. Keystone es un servicio que está configurado en la región 1 y consumido por ambas regiones, ofreciendo un sistema de autenticación unificado para los usuarios.

Horizon panel de control basado en web para OpenStack. A través de una interfaz gráfica, los usuarios pueden gestionar recursos de la nube como instancias, redes y almacenamiento de manera fácil e intuitiva. En esta arquitectura el servicio se ha configurado en la región 1 pero también es consumida por ambas regiones.

Glance la plataforma cuenta con el servicio de gestión de imágenes en OpenStack, Glance permite almacenar y recuperar imágenes que son utilizadas para el despliegue de nuevas instancias en la nube.

Placement este servicio es el encargado de la asignación de recursos (CPU, memoria y almacenamiento) entre las instancias. Placement asegura que los recursos sean utilizados de manera eficiente en toda la infraestructura.

Neutron servicio de redes en OpenStack que proporciona conectividad entre las instancias. Permite crear redes virtuales, subredes, enrutamiento, y otros componentes de red necesarios para garantizar la comunicación entre las instancias y al acceso a redes externas.

Cinder es el servicio de almacenamiento en bloque, este servicio permite a los usuarios crear y gestionar volúmenes de almacenamiento persistente que pueden ser adjuntados a las instancias, garantizando la durabilidad de los datos más allá del ciclo de vida de una máquina virtual.

Trove facilita el despliegue, operación y escalabilidad de bases de datos como servicio, ofreciendo una plataforma fácil de gestionar y optimizar para bases de datos en la nube.

Nova es el servicio encargado de gestionar las máquinas virtuales en la nube, coordinando todo el ciclo de vida de las instancias, desde su creación hasta su destrucción.

Magnum Orquesta contenedores como servicio, permitiendo la gestión y despliegue de contenedores de aplicaciones de manera nativa en OpenStack.

Heat es el motor de orquestación de OpenStack que permite a los usuarios definir infraestructuras complejas mediante plantillas y automatizar el despliegue de estas infraestructuras.

Designate es el servicio de gestión de DNS en OpenStack. Permite a los usuarios definir zonas y registros DNS, asegurando que las instancias y otros servicios puedan ser accesibles a través de nombres de dominios personalizados.

Octavia proporciona servicios de balanceo de carga, permitiendo distribuir el tráfico de manera equilibrada entre múltiples instancias para mejorar la disponibilidad y el rendimiento de las aplicaciones desplegadas.

Finalmente, el servicio de Yuyu proporciona a los usuarios una visión clara de los costos asociados al uso de recursos en la nube. Está configurado para monitorear los servicios de nova, cinder, neutron y keystone para recopilar datos de uso de la infraestructura, ayudando a gestionar presupuestos y optimizar gastos.

Cada región está dividida en zonas de disponibilidad que contienen nodos de cómputo dedicados a la ejecución de instancias. Las zonas de disponibilidad proporcionan aislamiento de fallos y aseguran que los recursos estén distribuidos. Los nodos de cómputo son responsables de gestionar las cargas de trabajo de los usuarios. En esta arquitectura, las zonas están compuestas por los siguientes nodos, compute 1 y compute 2 en la zona de disponibilidad 1, compute 3 y compute 4 en la zona de disponibilidad 2, estos nodos ejecutan las máquinas virtuales y aplicaciones, gestionando los recursos de cómputo de manera escalable y eficiente.

El almacenamiento distribuido es manejado por un clúster Ceph, que garantiza la escalabilidad y redundancia de los datos. Está compuesto por los siguientes nodos:

Ceph-admin es el nodo que centraliza la administración de todos los servicios del clúster Ceph, funcionando como el servidor principal para desplegar la solución. Adicionalmente, ejecuta ceph-mgr, un servicio clave para monitorear el rendimiento y manejar la administración del clúster.

Ceph-mon el nodo monitor tiene la responsabilidad de gestionar un mapa detallado del clúster, que incluye la supervisión de los OSD y todos los elementos que forman parte del clúster.

Ceph-osd1, ceph-osd2 y ceph-osd3 son los nodos de almacenamiento físico dentro del clúster Ceph. Estos nodos trabajan de manera redundante para asegurar la disponibilidad de los datos incluso en caso de fallos en alguno de ellos.

Finalmente, OpnSense nodo que se dedica a la seguridad y administración de red, este nodo funciona como firewall y gestor de red central, encargado de proteger la infraestructura y garantizar el flujo seguro de datos entre los diferentes servicios y usuarios.

CASO DE ESTUDIO

Las empresas dedicadas al desarrollo de software en el área metropolitana de El Salvador que están en creciente demanda de servicios tecnológicos por parte de sus clientes se ven en la necesidad de ofrecer servicios de alta disponibilidad, mantener la seguridad y la eficiencia de sistemas. El despliegue de una infraestructura de nube pública basada en OpenStack ofrece una solución flexible, escalable y segura para enfrentar los retos antes mencionados.

RAVN es una empresa de desarrollo de software en expansión que atiende a clientes de varias regiones de centroamérica. Se especializa en desarrollar soluciones personalizadas para automatización de ventas, plataformas de comercio electrónico, y herramientas de gestión de inventarios, enfrenta crecientes demandas de sus clientes, lo que la lleva a requerir servicios de alta disponibilidad, seguridad y eficiencia operativa. Sin embargo, tiene dificultades para utilizar soluciones preconfiguradas que simplifiquen el desarrollo de aplicaciones personalizadas. Para satisfacer estas necesidades, este prototipo ofrece una infraestructura basada en OpenStack que incluye servicios esenciales como creación de instancias, carga de imágenes y almacenamiento, entre otros, Además, el prototipo proporciona stacks que facilitan el desarrollo al ofrecer plantillas preconfiguradas, como por ejemplo Spring Boot con MySQL, y WordPress con MySQL, junto con herramientas como Git, para la gestión de repositorios remotos. De esta manera, RAVN puede mejorar su capacidad de respuesta a las necesidades del mercado, optimizar su proceso de desarrollo y asegurar una operación eficiente y segura.

ANEXO 3: Documento LLD

UNIVERSIDAD DE EL SALVADOR

FACULTAD DE INGENIERÍA Y ARQUITECTURA

DOCUMENTO DE DISEÑO DE BAJO NIVEL

LOW LEVEL DESIGN DOCUMENT

NOVIEMBRE 2024

VERSIÓN 1.0 17/11/2024

INDICE

INTRODUCCIÓN

El presente documento tiene como objetivo proporcionar un Diseño de Bajo Nivel (LLD) para el prototipo de infraestructura de nube pública. Este LLD complementa el documento de Diseño de Alto Nivel (HLD) previamente presentado, el cual estableció la visión general y los componentes esenciales del prototipo. En este documento, se explorarán los detalles técnicos necesarios para el despliegue del prototipo, describiendo las configuraciones requeridas y los pasos a seguir para llevar a cabo la construcción del prototipo. Además, se presentarán las pruebas realizadas para validar el correcto funcionamiento de la solución, asegurando que cumpla con los estándares y requisitos establecidos.

OBJETIVOS

OBJETIVO GENERAL

Proporcionar una guía detallada para la implementación del prototipo de infraestructura de nube pública, que incluya la descripción técnica de cada componente y las configuraciones necesarias.

OBJETIVOS ESPECIFICOS

Explicar los pasos necesarios para la construcción del prototipo de nube pública para asegurar el correcto funcionamiento de la infraestructura.

Detallar las configuraciones y archivos internos requeridos para cada componente de la infraestructura, proporcionando una guía de su configuración.

Configuraciones previas

Instalación de KVM en Ubuntu 22.04 LTS

Para la instalación es recomendado en actualizar todos los paquetes

```
sudo apt update && sudo apt upgrade -y
```

Antes de comenzar la instalación de KVM en Ubuntu 22.04 LTS, es importante asegurarse de que el hardware sea compatible con la virtualización. Para verificar si el procesador soporta virtualización, ejecutar el siguiente comando en la terminal.

```
egrep -c '(vmx|svm)' /proc/cpuinfo
```

Si el resultado es mayor a 0, significa que el hardware soporta la virtualización necesaria para KVM. Luego, es necesario comprobar si el sistema está listo para soportar KVM. Para ello, se debe de ejecutar el siguiente comando:

```
sudo kvm-ok
```

Si al ejecutar el comando se obtiene un mensaje “sudo: kvm-ok: comando no encontrado”, es porque esta herramienta no se encuentra instalada. Se puede instalar ejecutando el siguiente comando:

```
sudo apt install cpu-checker
```

Una vez instalada, ejecutar de nuevo, esto confirmará si el sistema es compatible a KVM

```
sudo kvm-ok
```

A continuación, se debe proceder a instalar KVM junto con sus dependencias asociadas, como virt-manager y bridge-utils, con el siguiente comando:

```
sudo apt install -y qemu qemu-kvm libvirt-daemon libvirt-
clients bridge-utils virt-manager
```

Cuando finalice la instalación, es importante comprobar que el servicio de libvirtd está corriendo correctamente. Para ello es de ejecutar:

```
sudo systemctl status libvirtd
```

Si el servicio está activo y en ejecución, asegurarse de que se inicie automáticamente

```
sudo systemctl enable --now libvirtd
```

Después de haber instalado KVM y configurado libvirt, se debe agregar el usuario al grupo kvm y libvirt.

```
sudo usermod -aG kvm $USER
sudo usermod -aG libvirt $USER
```

Creación de Máquinas Virtuales

Para crear una máquina virtual, primero se debe hacer clic en el icono de creación de máquinas virtuales en la interfaz de virt-manager.

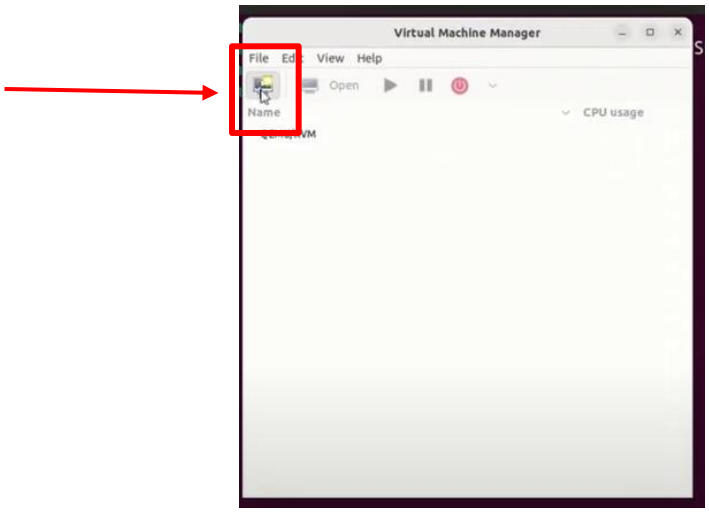


Figura 1: Crear una nueva máquina virtual en KVM

Se abrirá una ventana de creación donde se deberá seleccionar el método para crear la máquina virtual. En este caso, se elige la opción instalación local vía imagen ISO o CD-ROM.

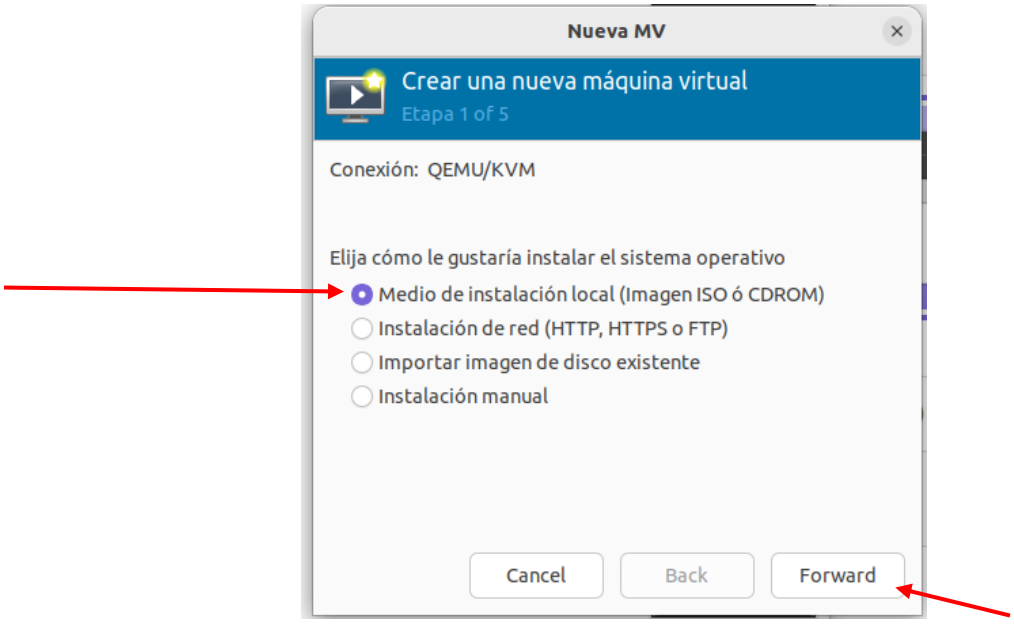


Figura 2: Opciones para crear máquina virtual.

El siguiente paso es seleccionar el medio de instalación, ya sea una unidad de CD o un archivo de imagen ISO. Como el método más común de instalación es mediante una imagen ISO, seleccionar el botón *Browse.*

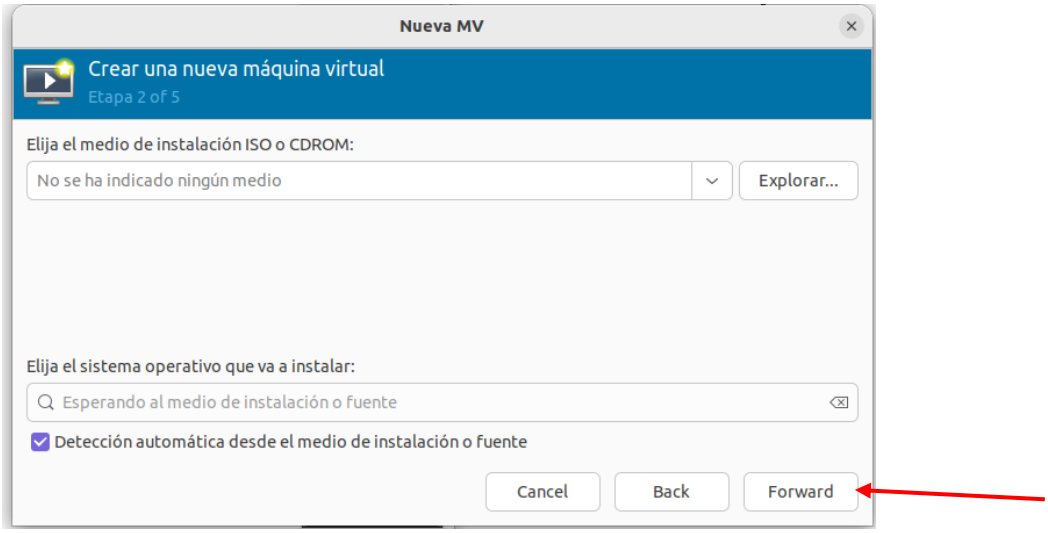


Figura 3: Elección de imagen en KVM

Seleccionar la imagen ISO del sistema operativo que se desea instalar. En este caso se usará la imagen de Ubuntu Server 22.04. El programa detectará automáticamente el sistema operativo de la imagen para crear el entorno.

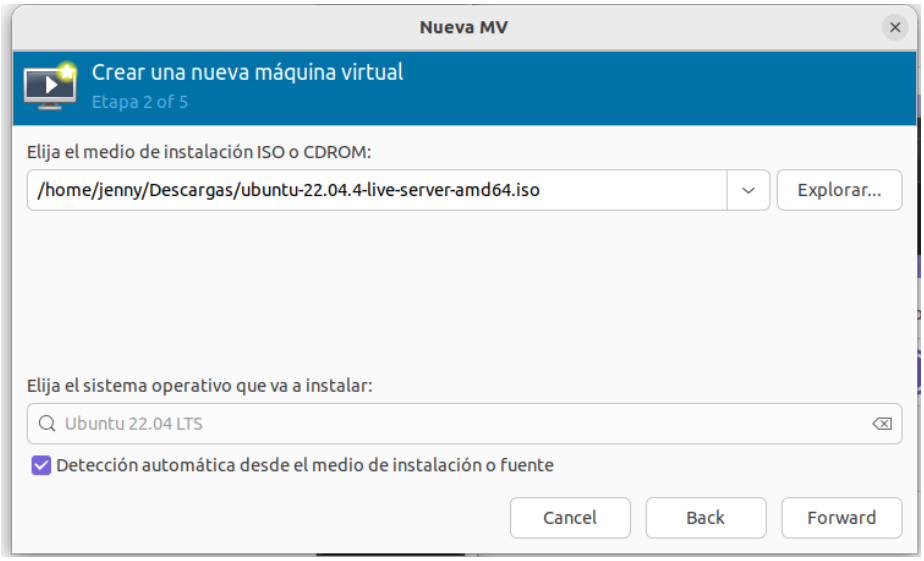


Figura 4: Elección de imagen en KVM

A continuación, se deben seguir las instrucciones para asignar hardware virtual, como la memoria RAM y el almacenamiento.

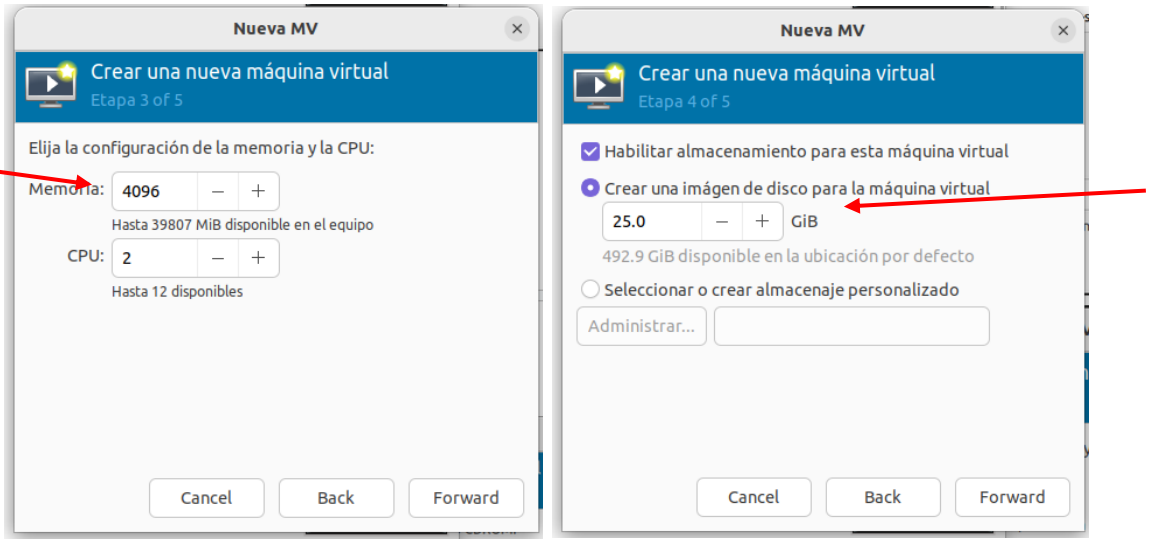


Figura 5: Elección de memoria RAM y almacenamiento

En la última pantalla, se configura el nombre de la máquina virtual y la red que utilizará su interfaz por defecto.

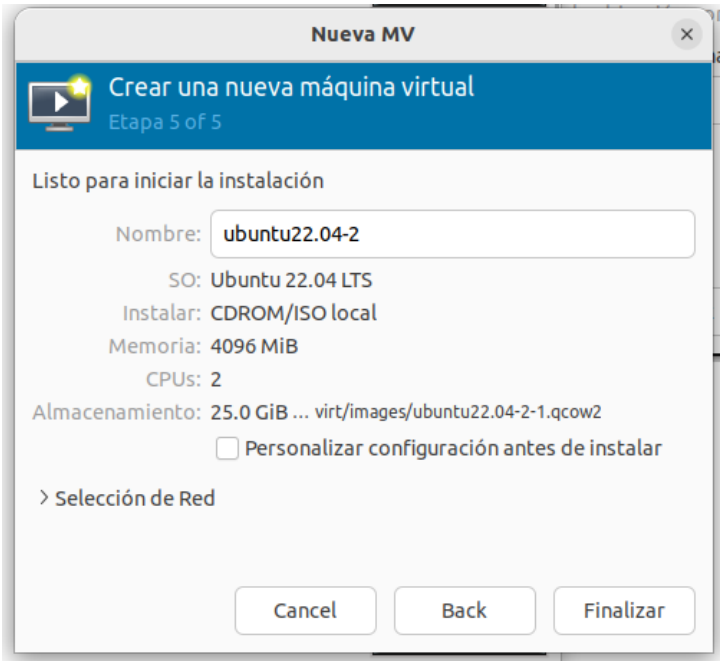


Figura 6: configuración nombre de maquina en KVM.

Abrir la ventana de edición de la máquina virtual, donde, además de modificar las características por defecto, se podrá agregar más hardware virtual.

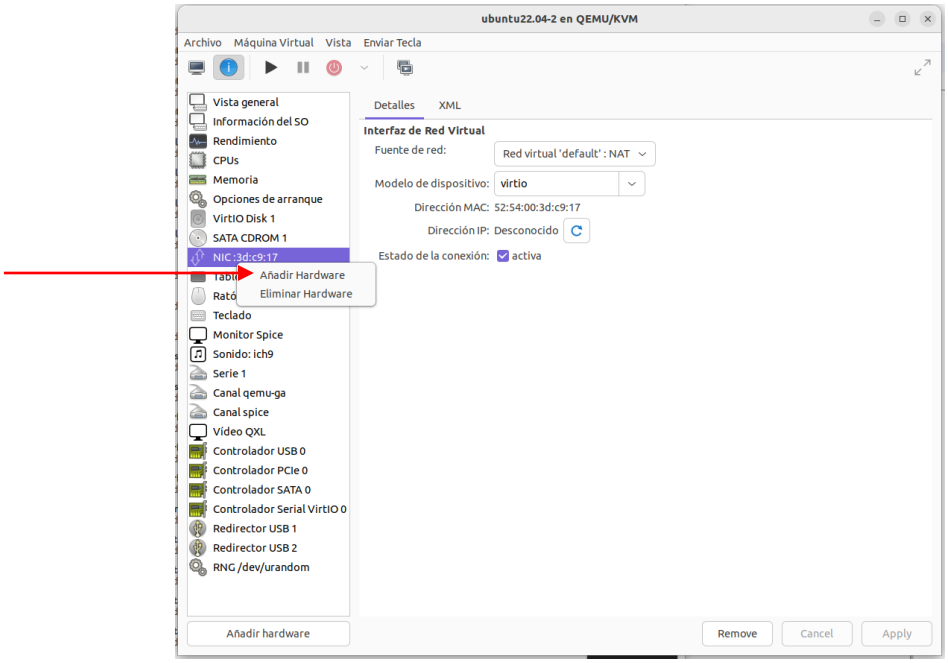


Figura 7: Agregar hardware

Al hacer clic en Agregar Hardware, aparecerá una nueva ventana para seleccionar el tipo de hardware virtual que se desea agregar y configurar sus parámetros.

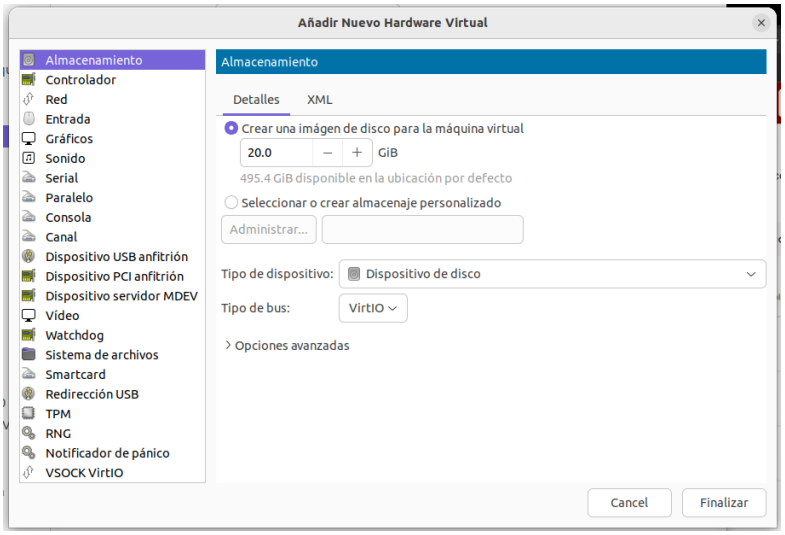


Figura 8: Agregando hardware almacenamiento

Se muestra una pantalla en la cual se puede ver la interfaz gráfica del sistema operativo a instalar. Los pasos siguientes depende de cada sistema operativo a instalar.

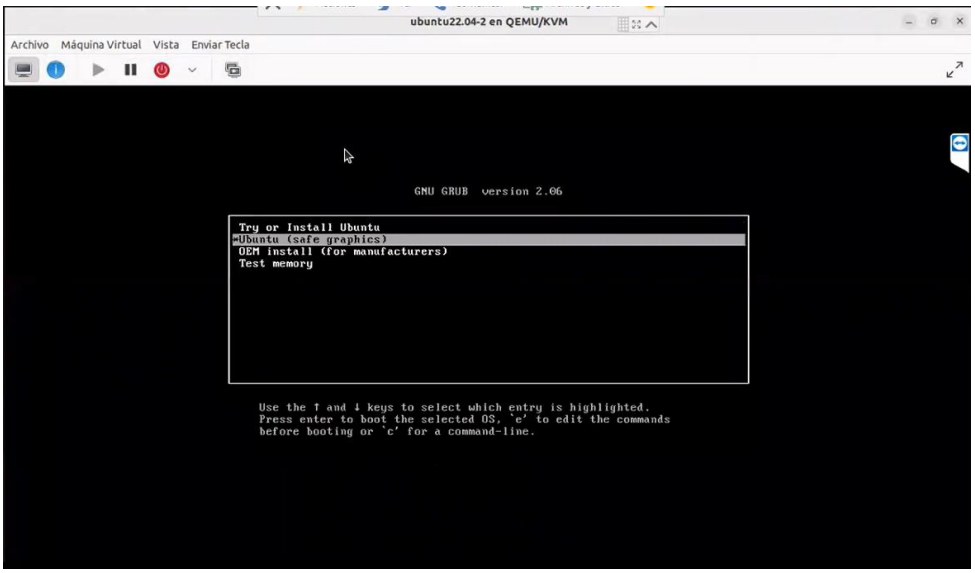


Figura 9: Instalación Ubuntu 22.04

Instalación y configuración de VPN

Para el despliegue del prototipo se configuro la red VPN con Zerotier para poder unificar todos los recursos de cómputo disponibles entre el equipo encargado del despliegue. ZeroTier es una empresa de software que facilita la creación y gestión de redes definidas por software (SDN), permitiendo conectar varios equipos en una red privada virtual (VPN) de forma sencilla y rápida.

Se utilizó la siguiente configuración de red la cual se debe configurar dentro del sitio web de Zerotier cuya URL es https://www.zerotier.com/. A continuación se debe crear una red en el sitio dando clic en el botón Create a Network.

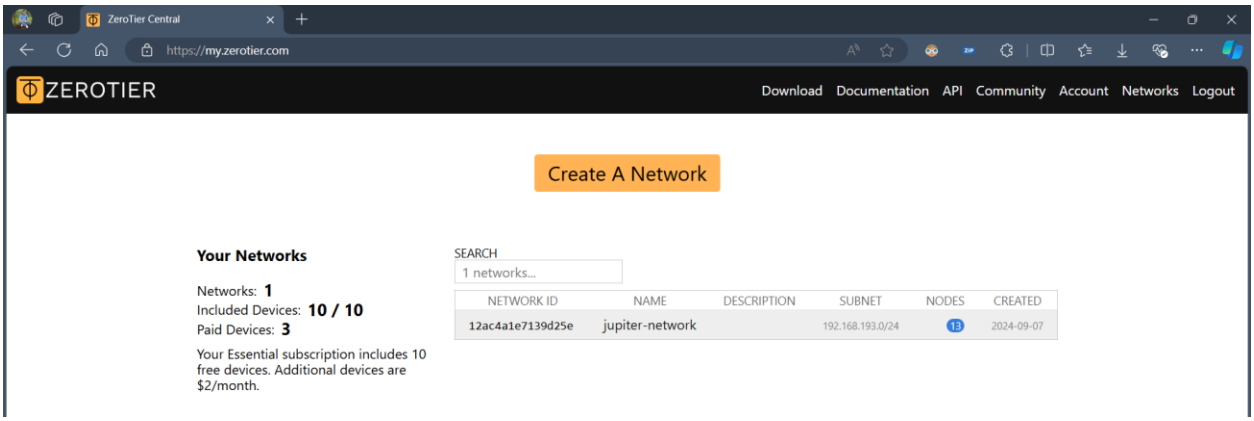


Figura 10: Creación de red en Zerotier

Se coloca un nombre a la red recientemente creada, para esto vamos a settings y en name colocamos el nombre correspondiente. En la sección access control elegimos private.

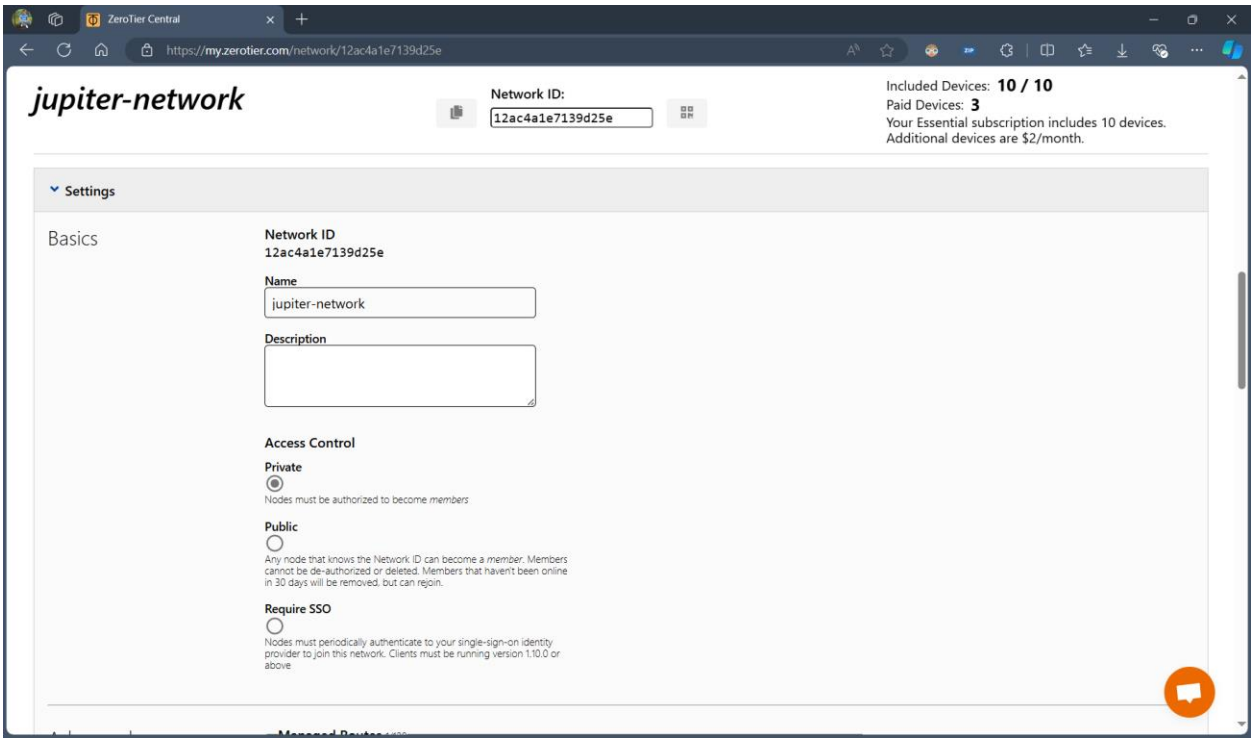


Figura 11: Nombramiento de red y configuración básica.

Se eligió la red 192.168.193.0/24 para desplegar el prototipo, por lo que se configuró en la sección de Advanced eligiendo la opción Easy sombreada con azul escogiendo la red

192.168.193.*. Con esto se finaliza la configuración por lo que se deberá dar clic en el botón save hasta el final de la página web.

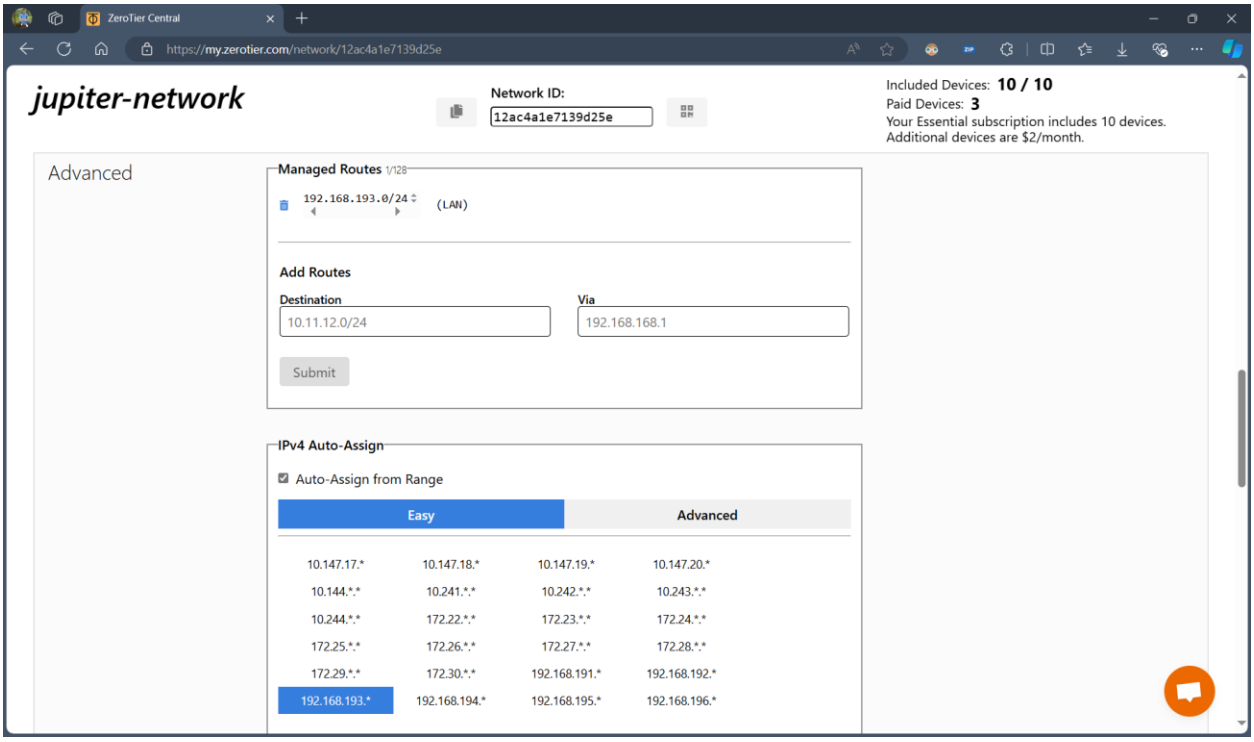


Figura 12: Elección de red VPN a utilizar en el despliegue del dispositivo.

Instalar Zerotier dentro de las máquinas virtuales para poder posteriormente ser parte de la red virtual VPN. Para instalar Zerotier se ocuparon los siguientes comandos:

```
curl -s
https://raw.githubusercontent.com/zerotier/ZeroTierOne/main/doc/contact%40zerotier.com.gpg' | gpg --import && \
     if z=$(curl -s 'https://install.zerotier.com/' | gpg);

then echo "$z" | sudo bash; fi
```

Desde la máquina virtual donde se instala Zerotier se obtiene el ID del nodo con el siguiente comando.

```
sudo zerotier-cli info
```

Lo cual nos dará el ID del nodo.

```
root@controller:/home/seferick# sudo zerotier-cli info
200 info 0eed3b3035 1.14.0 ONLINE
```

Se deberá ir agregando los ID de los nodos instalados dentro de las máquinas virtuales de Ubuntu server que utilizadas para el despliegue. Para esto se hará en la sección de advanced donde se agregará el nodo, después de digitado el ID del nodo se deberá dar clic al botón submit.

Figura 13: Agregar nodos del prototipo a la red VPN de Zerotier.

Luego se debe ejecutar el comando para poder ser parte de la VPN de Zerotier, teniendo en cuenta que en la parte superior del sitio web nos colocan el network ID como se visualiza en la imagen anterior, el cual utilizaremos con el siguiente comando.

```
sudo zerotier-cli join 12ac4a1e7139d25e
```

Finalmente, ya tendremos configurada nuestra red VPN para ser utilizada en el despliegue del prototipo. En la sección de members podremos ver los nodos agregados de la siguiente manera.

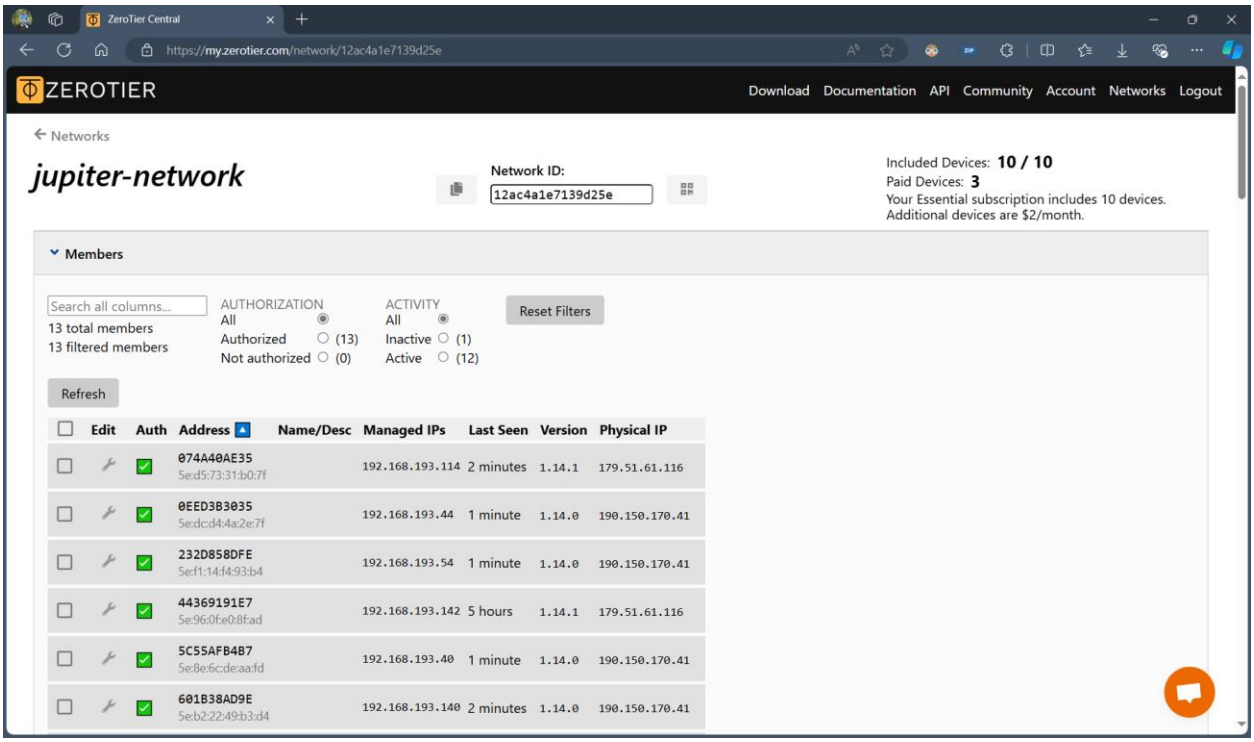


Figura 14: Nodos agregados a la red VPN de Zerotier.

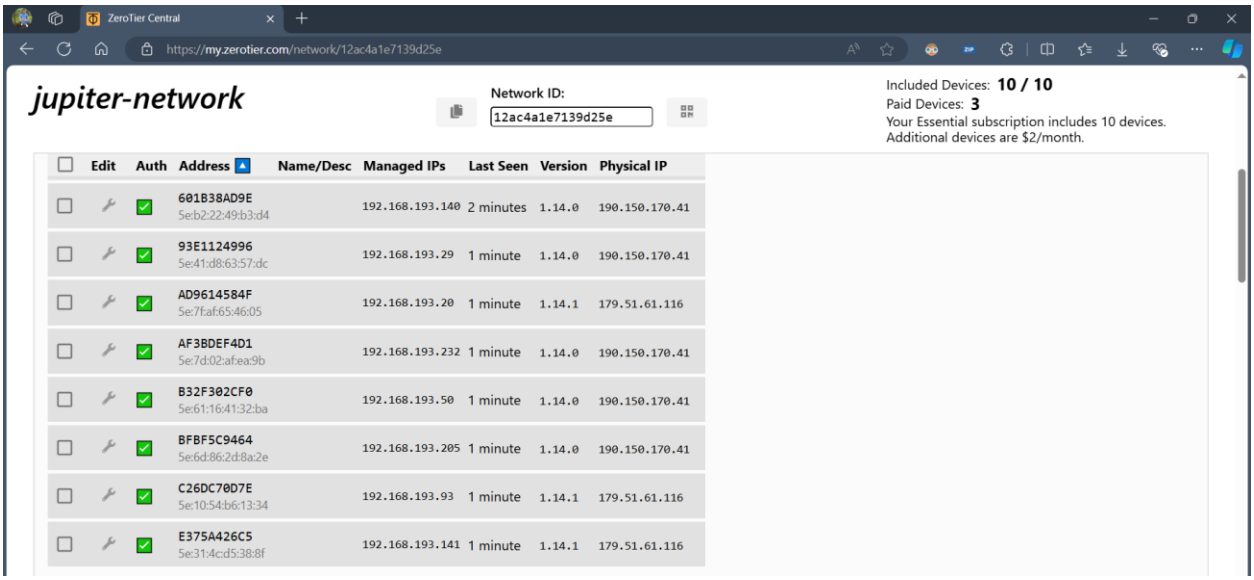


Figura 15: Nodos agregados a la red VPN de Zerotier.

Instalación y Configuración Clúster de CEPH

Para el clúster de CEPH se configuró 5 máquinas virtuales con los siguientes requerimientos.

Tabla 1

Recursos del Clúster CEPH

Cantidad	Nombre	Recursos
1	ceph-admin	2 GB RAM, 1 CPU, 100 GB Disk, 2 NIC
1	ceph-mon	2 GB RAM, 1 CPU, 100 GB Disk, 2 NIC
3	ceph-osd1, ceph-osd2, ceph-osd3	2GB RAM, 1CPU, 100GB Disk1, 300 GB Disk2, 2NIC

Tabla 2

Asignación de IP a los nodos.

Nodo	Interfaz	IP
ceph-admin	enp1s0	192.168.122.100
ceph-admin	enp9s0	192.168.3.50
ceph-admin	ztbtoxhtbp	192.168.193.232
ceph-mon	enp1s0	192.168.122.101
ceph-mon	enp9s0	192.168.3.51
ceph-mon	ztbtoxhtbp	192.168.193.205
ceph-osd1	enp1s0	192.168.122.102

ceph-osd1	enp9s0	192.168.3.52
ceph-osd1	ztbtoxhtbp	192.168.193.29
ceph-osd2	enp1s0	192.168.122.103
ceph-osd2	enp9s0	192.168.3.53
ceph-osd2	ztbtoxhtbp	192.168.193.40
ceph-osd3	enp1s0	192.168.122.104
ceph-osd3	enp9s0	192.168.3.54
ceph-osd3	ztbtoxhtbp	192.168.193.140

Configuraciones previas de los nodos

El hostname de los nodos debe ser actualizado mediante el siguiente comando:

```
admin@admin:~$ sudo hostnamectl set-hostname ceph-admin
admin@admin:~$ sudo exec bash
```

Nota: Se debe sustituir *ceph-admin* por el nombre correspondiente de cada nodo.

Después de cambiar el hostname, es necesario modificar el archivo /etc/hosts/ en cada nodo.

```
127.0.0.1 localhost
127.0.1.1 server1

192.168.3.50 ceph-admin
192.168.3.51 ceph-mon
192.168.3.52 ceph-osd1
192.168.3.53 ceph-osd2
192.168.3.54 ceph-osd3

# Config
10.0.0.12       Controller2
10.0.0.33       compute3
10.0.0.34       compute4
10.0.0.42       object2
192.168.122.11 controller
```

Para realizar las configuraciones de las direcciones IP de las máquinas virtuales, se editó el archivo netplan con el siguiente comando.

```
admin@ceph-admin:~$ sudo nano /etc/netplan/00-installer-
config.yaml
```

Se configura el netplan para el nodo ceph-admin con lo siguiente:

```
# This is the network config written by 'subiquity'
network:
  ethernets:
    enp1s0:
      dhcp4: false
      addresses: [192.168.122.100/24]
      gateway4: 192.168.122.1
      nameservers:
        addresses: [8.8.8.8,8.8.4.4]
    enp9s0:
      dhcp4: false
      addresses: [192.168.3.50/24]
    ztbtoxhtbp:
     addresses: [192.168.193.232/24]
     nameservers:
       addresses: [8.8.8.8,8.8.4.4]
     routes:
       - to: 10.0.0.12
         via: 192.168.193.141
         metric: 100
       - to: 10.0.0.33
         via: 192.168.193.93
         metric: 100
       - to: 10.0.0.34
         via: 192.168.193.114
         metric: 100
       - to: 10.0.0.42
         via: 192.168.193.142
         metric: 100
  version: 2
```

Nota: Realizar el mismo procedimiento en los demás nodos, ajustando la configuración de acuerdo a sus respectivas IP.

Instalación de paquetes

Se llevó a cabo la instalación de Chrony en todos los nodos del clúster.

```
admin@cep-admin:~$ sudo apt install chrony -y
```

Instalación de los paquetes requeridos para Docker.

```
admin@cep-admin:~$ sudo apt install apt-transport-https ca-
certificates curl gnupg-agent software-properties-common -y
```

Se registró la clave del repositorio para los paquetes.

```
admin@cep-admin:~$ curl -fsSL
https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add
-
```

Se agregó el repositorio de Docker a la lista de apt con el siguiente comando:

```
admin@cep-admin:~$ echo "deb [arch=amd64]
https://download.docker.com/linux/ubuntu $(lsb_release -sc)
stable" | sudo tee /etc/apt/sources.list.d/docker-ce.list
```

Se actualizaron los paquetes de los repositorios.

```
admin@cep-admin:~$ sudo apt update
```

Se instalaron los paquetes de Docker.

```
admin@cep-admin:~$ sudo apt install docker-ce docker-ce-cli
containerd.io -y
```

Configuración del servicio

Creación del Usuario

Se creó un usuario en el nodo ceph-admin para permitir conexiones SSH desde un cliente y facilitar la recuperación de archivos de configuración. Para lograrlo, se accedió a la cuenta root. El nuevo usuario es cephadmin y la contraseña establecida icc1152024.

```
admin@ceph-admin:~$ sudo su
root@ceph-admin:/home/admin# useradd -m -s /bin/bash cephadmin
```

```
root@ceph-admin:/home/admin# passwd cephadmin
New password:
Retype new password:
passwd: password updated successfully
```

Agregando el usuario al grupo sudo para tener permisos de super usuario.

```
root@ceph-admin:/home/admin# echo "cephadmin ALL=(ALL:ALL)
NOPASSWD:ALL" >> /etc/sudoers.d/cephadmin
root@ceph-admin:/home/admin# chmod 0440
/etc/sudoers.d/cephadmin
```

Habilitar acceso SSH para root en nodos y osds

Para permitir que el nodo admin se conecte a los nodos osd y monitor por SSH, fue necesario habilitar la conexión desde root y editar el archivo sshd_config en cada nodo.

```
admin@ceph-mon:~$ sudo su
root@ceph-mon:/home/admin# nano /etc/ssh/sshd_config
```

Buscar la línea *PermitRootLogin* y se establece en *yes.*

```
...
PermitRootLogin yes
...
```

Se guardaron los cambios, y se reinició el servicio.

```
admin@ceph-mon:/home/admin# nano /etc/init.d/ssh restart
```

Con la conexión SSH habilitada para el usuario root, se configuró la contraseña **icc1152024** en todos los nodos osd como en el nodo monitor.

```
admin@ceph-mon:/home/admin# passwd
New password:
Retype new password:
passwd: password updated successfully
```

Descarga de CEPH

En el nodo ceph-admin, Ceph fue instalado de manera manual. Para ello, se descargó la herramienta cephadm y se realizó la instalación desde root.

```
admin@ceph-admin:/home/admin# wget -q
https://github.com/ceph/ceph/raw/quincy/src/cephadm/cephadm -P
/usr/bin/
```

Se hizo cambio a los permisos a la utilidad.

```
admin@ceph-admin:/home/admin# chmod +x /usr/bin/cephadm
```

Despliegue del servicio

Se creó un clúster utilizando el comando de Bootstrap de cephadm. La dirección IP pública asignada al nodo ceph-admin. Por defecto, Ceph considera la red correspondiente a esta dirección IP como la red pública del clúster.

```
admin@ceph-admin:$ sudo cephadm bootstrap --cluster-network
192.168.3.0/24 --mon-ip 192.168.122.100
```

Esta ejecución puede tardar un poco debido a que realiza varias acciones sobre los nodos osd y monitor, al terminar la ejecución se mostrará lo siguiente:

```
Ceph Dashboard is now available at:
      URL: https://ceph-admin:8443/
      User: admin
     Password: mi1nc7trkr
Enabling client.admin keyring and conf on hosts with "admin"
label
Saving cluster configuration to /var/lib/ceph/7d0671fa-54f2-
11ef-8fb4-f5a38cf0151f/config directory
Enabling autotune for osd_memory_target
You can access the Ceph CLI as following in case of multi-
cluster or non-default config:
```

```
     sudo /usr/bin/cephadm shell --fsid 7d0671fa-54f2-11ef-
8fb4-f5a38cf0151f -c /etc/ceph/ceph.conf -k
/etc/ceph/ceph.client.admin.keyring
Or, if you are only running a single cluster on this host:
     sudo /usr/bin/cephadm shell
Please consider enabling telemetry to help improve Ceph:
     ceph telemetry on
For more information see:
     https://docs.ceph.com/docs/master/mgr/telemetry/
Bootstrap complete.
```

En la consola se puede ver que para acceder al servicio desde el navegador se debe de hacer *https://ceph-admin:8443*.

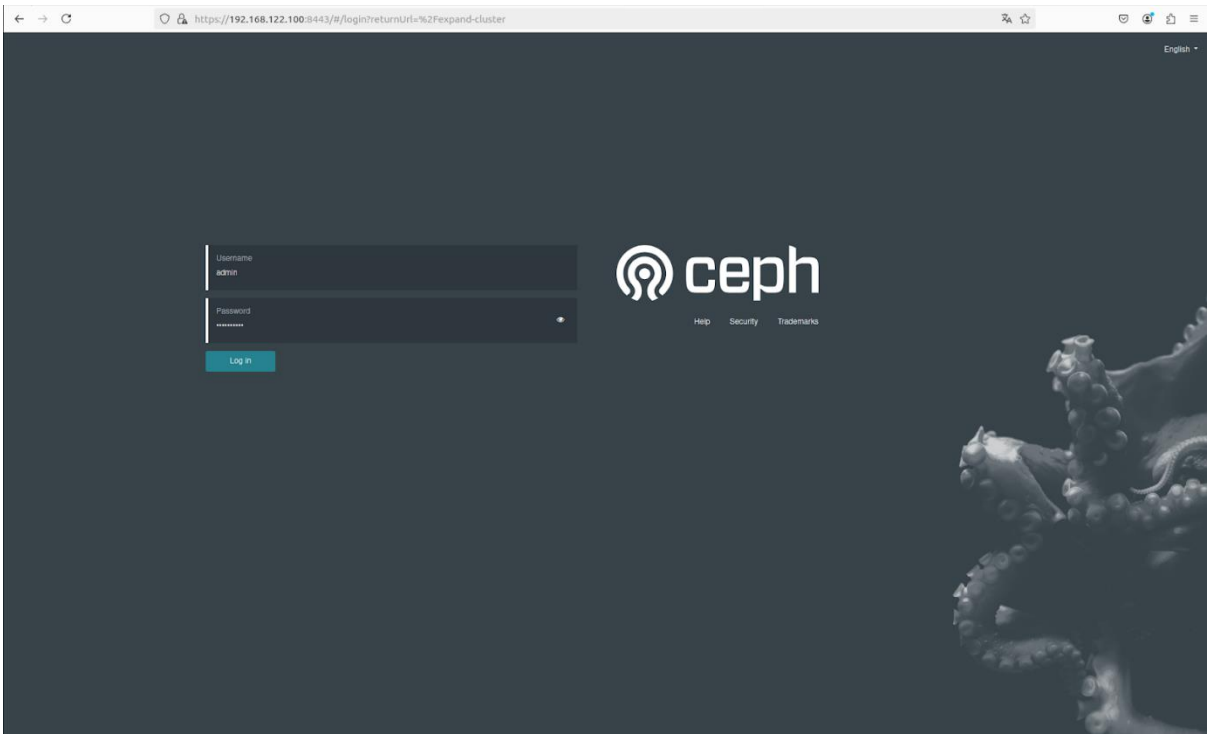


Figura 16: Login Ceph

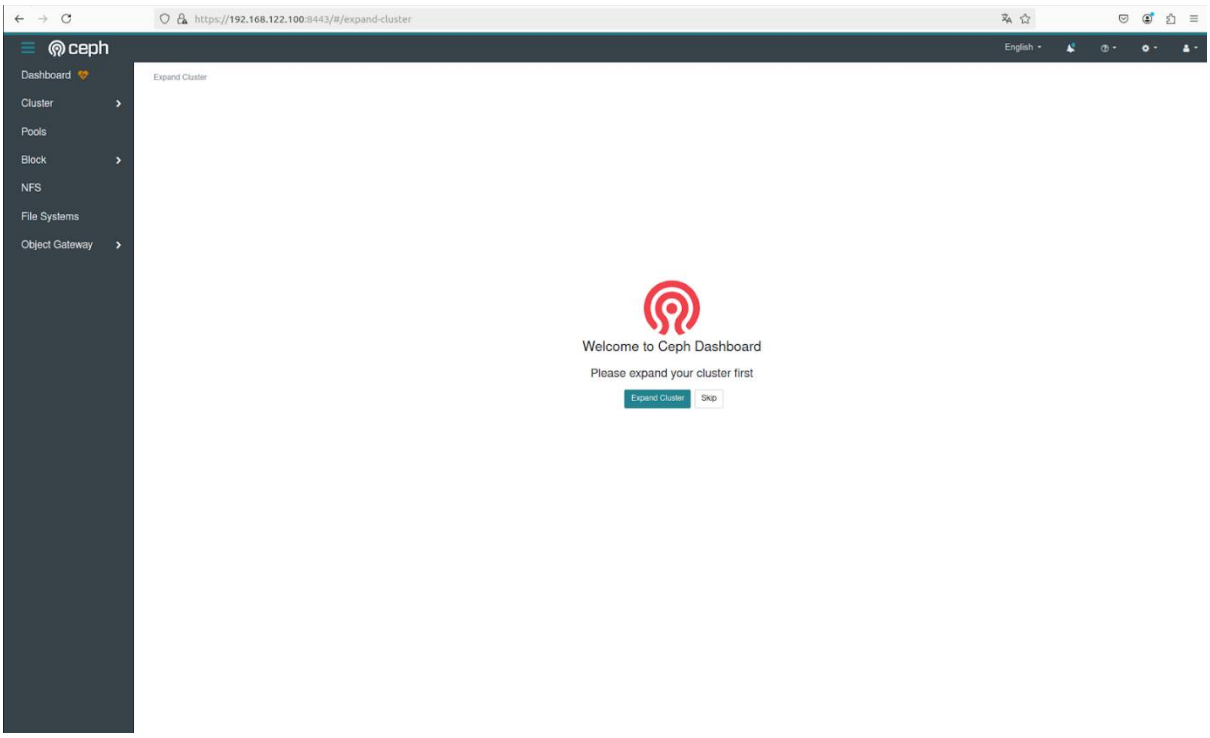


Figura 17: Dashboard Ceph

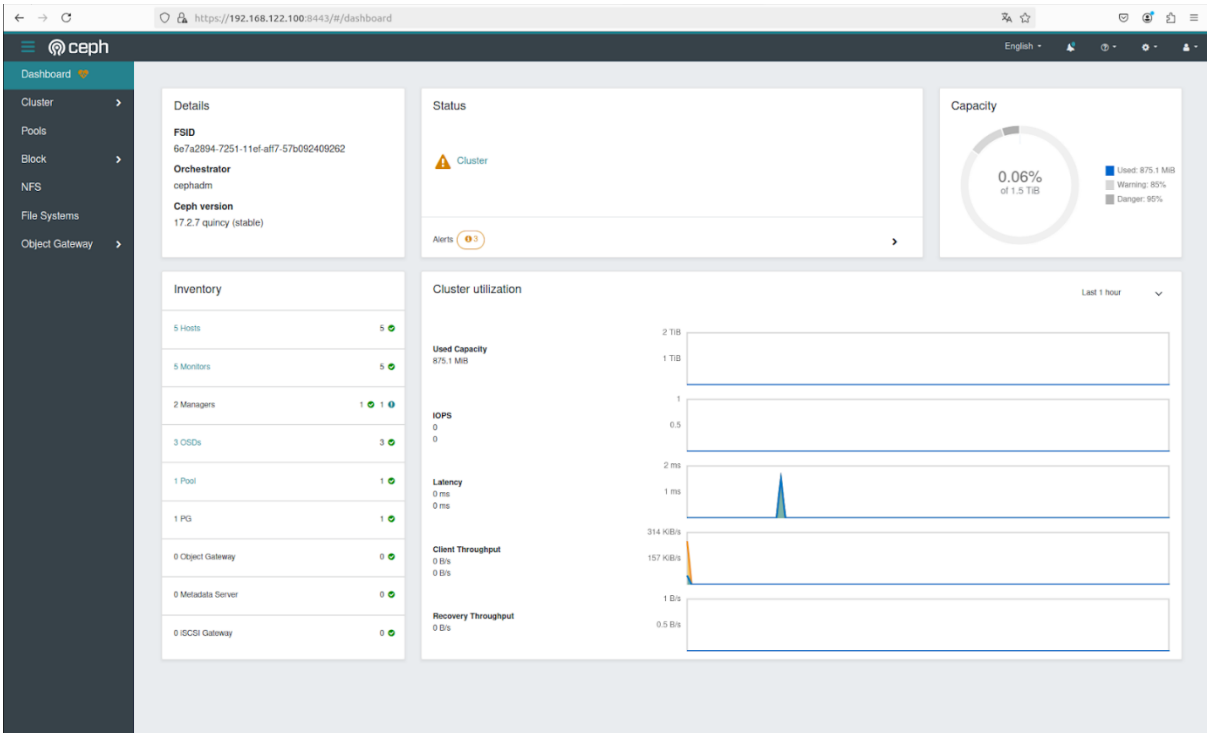


Figura 18: Dashboard Ceph

Una vez que el servicio está desplegado, se habilita Ceph CLI mediante el comando que muestra la salida del proceso de despliegue.

```
admin@ceph-admin:$ sudo /usr/bin/cephadm shell --fsid f959b65e-
91c2-11ec-9776-abbffb8a52a1 -c /etc/ceph/ceph.conf -k
/etc/ceph/ceph.client.admin.keyring
```

Para revisar la salud del Clúster se debe ejecutar el siguiente comando:

```
admin@ceph-admin:/home/admin# ceph -s
```

Se sugiere tener disponibles bibliotecas comunes para el acceso desde la terminal. Para ello, se ejecutaron los siguientes comandos.

Instalando ceph-common.

```
sudo cephadm add-repo --release quincy
sudo cephadm install ceph-common
sudo ceph -s
```

Para listar los componentes del clúster se debe de ejecutar.

```
sudo ceph orch host ls
```

Registro Nodo Monitor

Copiar llave al Nodo Monitor ceph-mon.

```
sudo ssh-copy-id -f -i /etc/ceph/ceph.pub root@ceph-mon
```

Agregándole el clúster de ceph.

```
sudo ceph orch host add ceph-mon
```

Agregar la etiqueta mod/osd al nodo ceph-mon recién agregado.

```
sudo ceph orch host label add ceph-mon mon/osd
```

Registro de los nodos OSD

El registro de los nodos OSD debe realizarse desde el nodo ceph-admin. Primero, se copia la clave pública desde el admin hacia los nodos cephs-osd.

Instalar el paquete lvm2 en todos los nodos que serán OSD.

```
sudo apt install lvm2
```

Copiar las llaves a cada uno de los nodos.

```
#for i in ceph-osd1 ceph-osd2 ceph-osd3; do sudo ssh-copy-id -f
-i /etc/ceph/ceph.pub root@$i; done
```

Registro de los nodos en el clúster.

```
sudo ceph orch host add ceph-osd1
sudo ceph orch host add ceph-osd2
sudo ceph orch host add ceph-osd3
```

Estableciendo las etiquetas que identificaran a cada uno de los nodos en el clúster.

```
#for i in ceph-osd1 ceph-osd2 ceph-osd3; do sudo ceph orch host
label add $i osd; done
```

Listando nuevamente los componentes del clúster.

```
sudo ceph orch host ls
```

Inflando el clúster

Para esto se debe de tener configurado un almacenamiento adicional en los nodos OSD.

Listando los discos disponibles en el host

```
sudo fdisk -l
```

Crear el volumen LVM vg01

```
vgcreate vg01 /dev/vdb
```

Creando volumen de disco lv01 en el grupo lvm vg01

```
lvcreate -L 49G -n lv01 vg01
```

Verificar si los volúmenes se han creado correctamente usando el siguiente comando

```
lvdisplay
```

Agregando volúmenes al clúster

Se deberá ejecutar en el nodo ceph-admin los siguientes comandos

```
sudo ceph orch daemon add osd ceph-osd1:vg01/lv01
sudo ceph orch daemon add osd ceph-osd2:vg01/lv01
sudo ceph orch daemon add osd ceph-osd3:vg01/lv01
```

Instalación y configuración de OPNsense

Primero que todo es necesario descargar la imagen ISO del sitio oficial de OPNsense en la siguiente URL https://opnsense.org/download/ configurando las siguientes opciones damos clic en descargar:

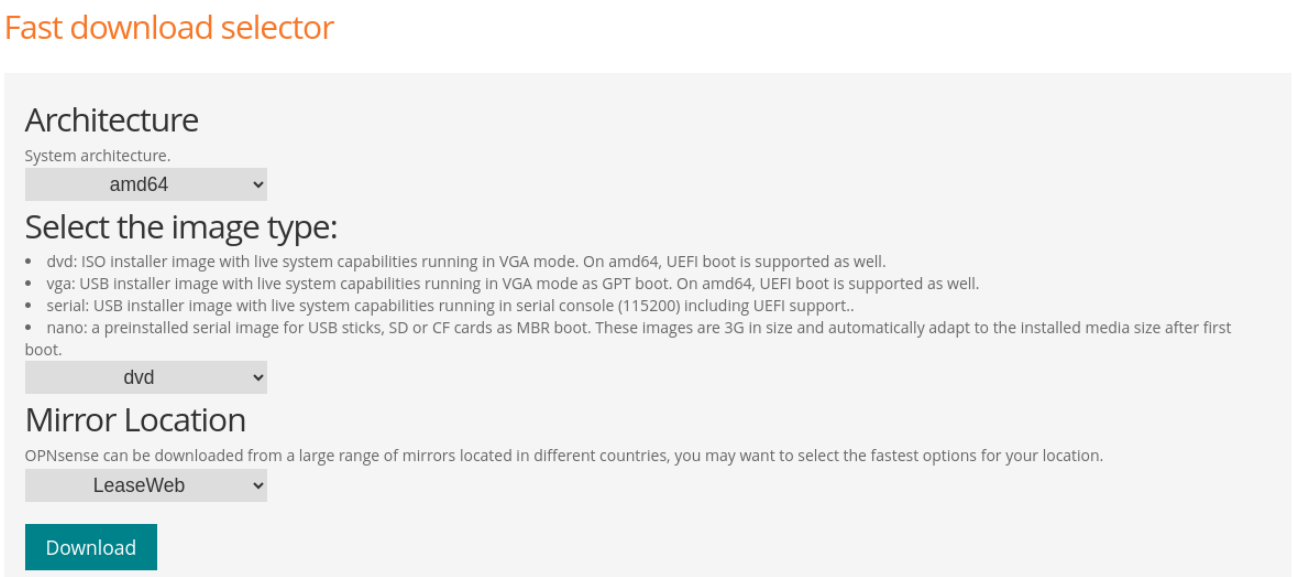


Figura 19: Sition oficial OPNsense para descargas.

Se crea una nueva máquina virtual a la cual le asignamos la ISO descargada.

Elegimos el sistema operativo a instalar FreeBSD 13.1.

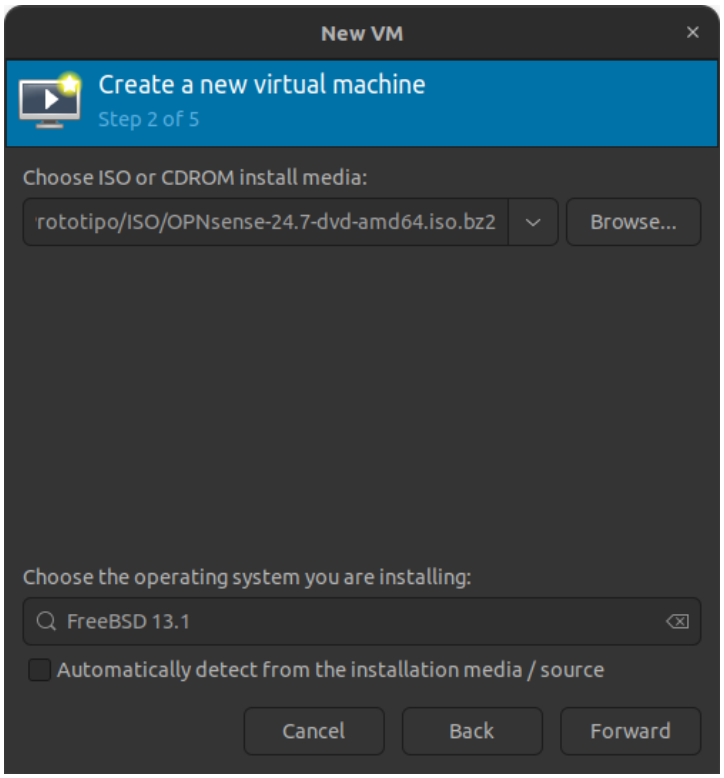


Figura 20: Configuración de máquina virtual.

Definimos las características que tendrá la máquina virtual para el nodo OPNsense.

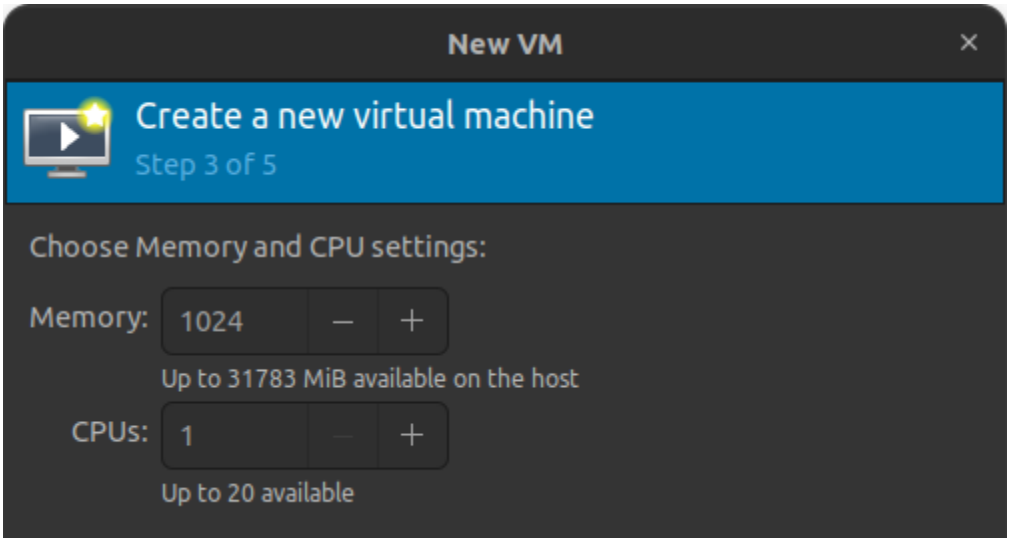


Figura 21: Configuración de RAM y CPUs.

Definimos la capacidad del disco de la máquina virtual.

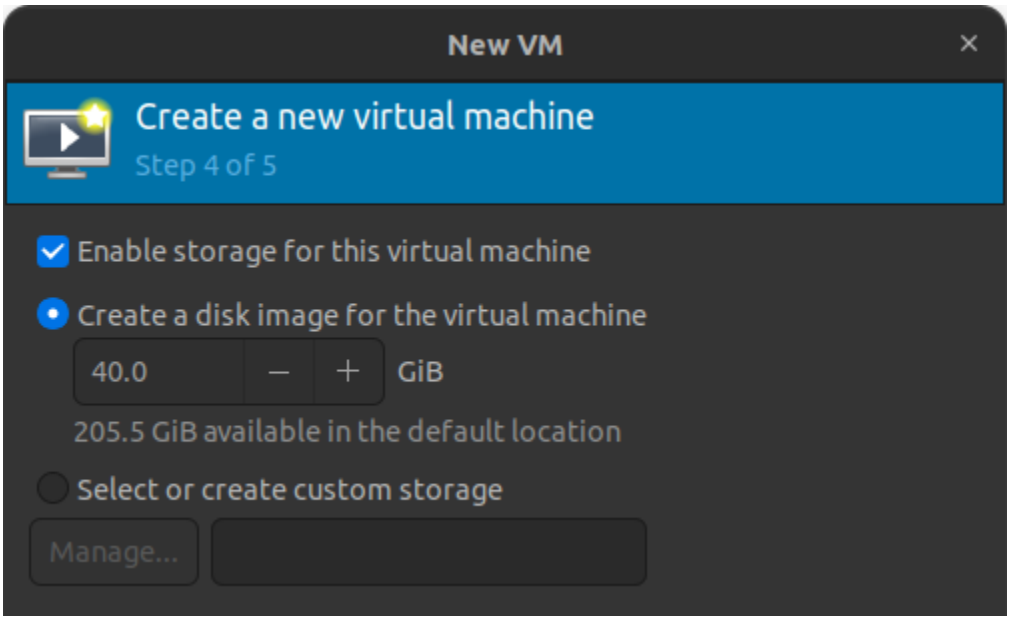


Figura 22: Configuración de disco de máquina virtual.

Seleccionamos el check box para customizar la máquina virtual antes de instalar.

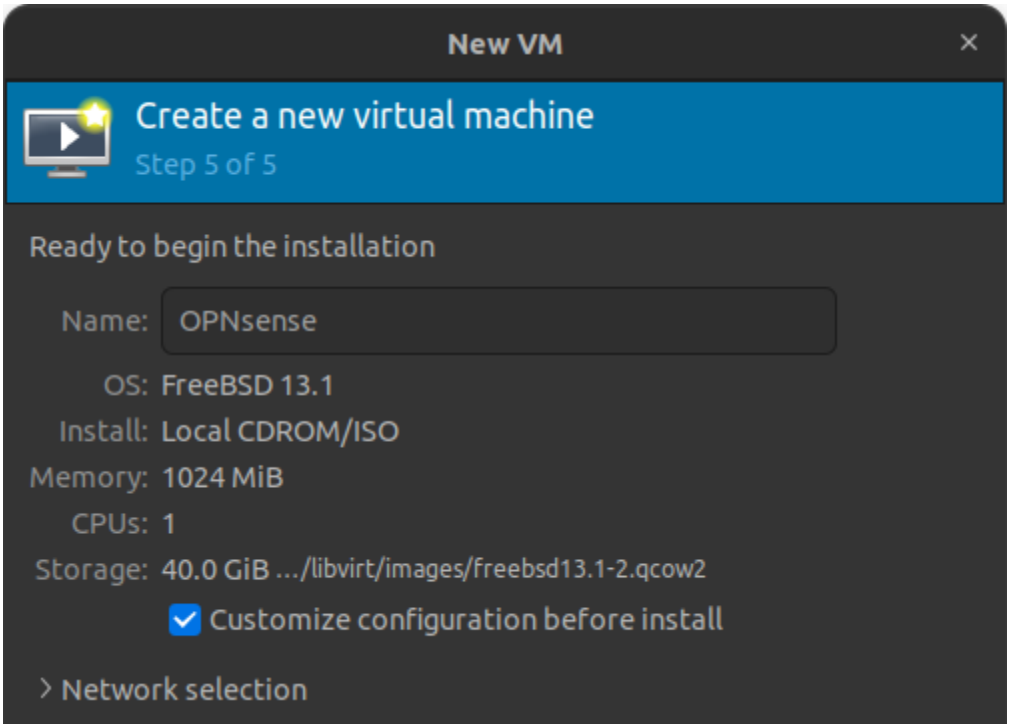


Figura 23: Configuración resumen.

Agregamos un hardware con la opción “Add hardware” y le añadimos una nueva tarjeta de red la cual configuramos como bridge de la siguiente manera.

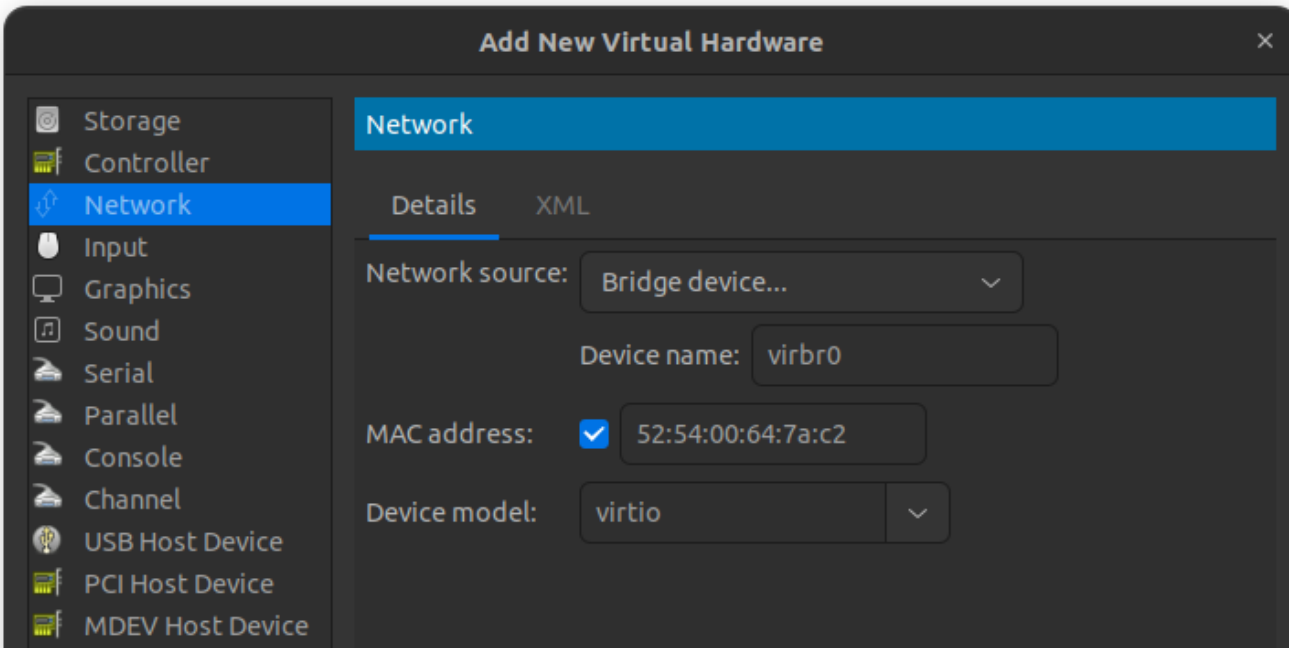


Figura 24: Configuración de interfaz de red.

Colocamos el archivo ISO del contiene el sistema operativo respectivo de OPNsense.

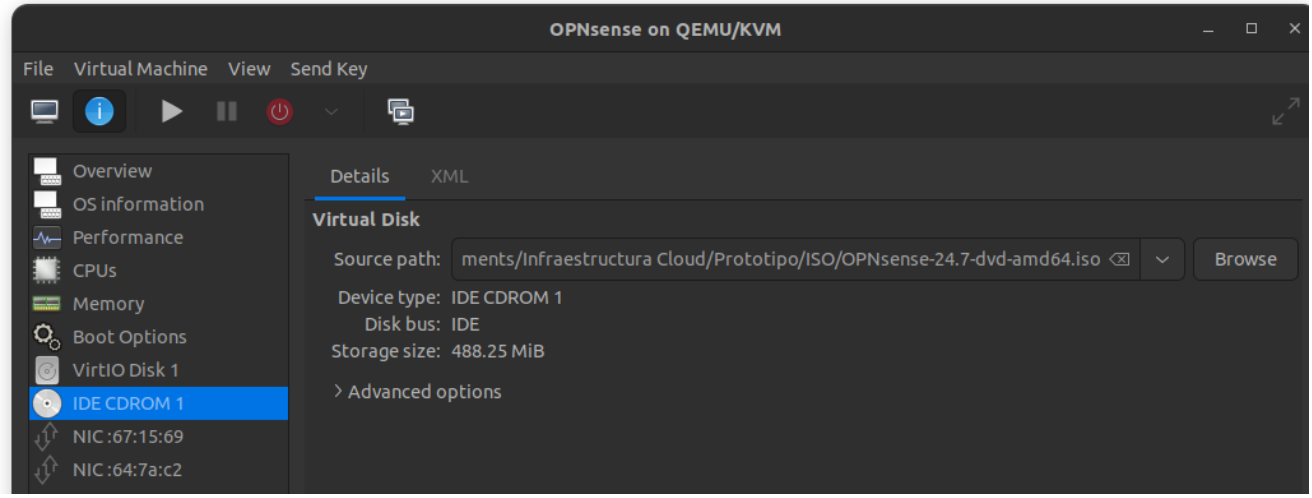


Figura 25: Configuración de imagen ISO.

Cuando cargue la pantalla y en el prompting nos permita ver el login ingresamos el usuario “installer” y en password “opnsense” para poder acceder a la configuración.

Figura 26: Acceso a configuración OPNsense.

A continuación dejamos la opción por default y presionamos enter.

Figura 27: Configuración de OPNsense en progreso.

Elegimos la opción “Install (UFS)” y presionamos enter sobre OK.

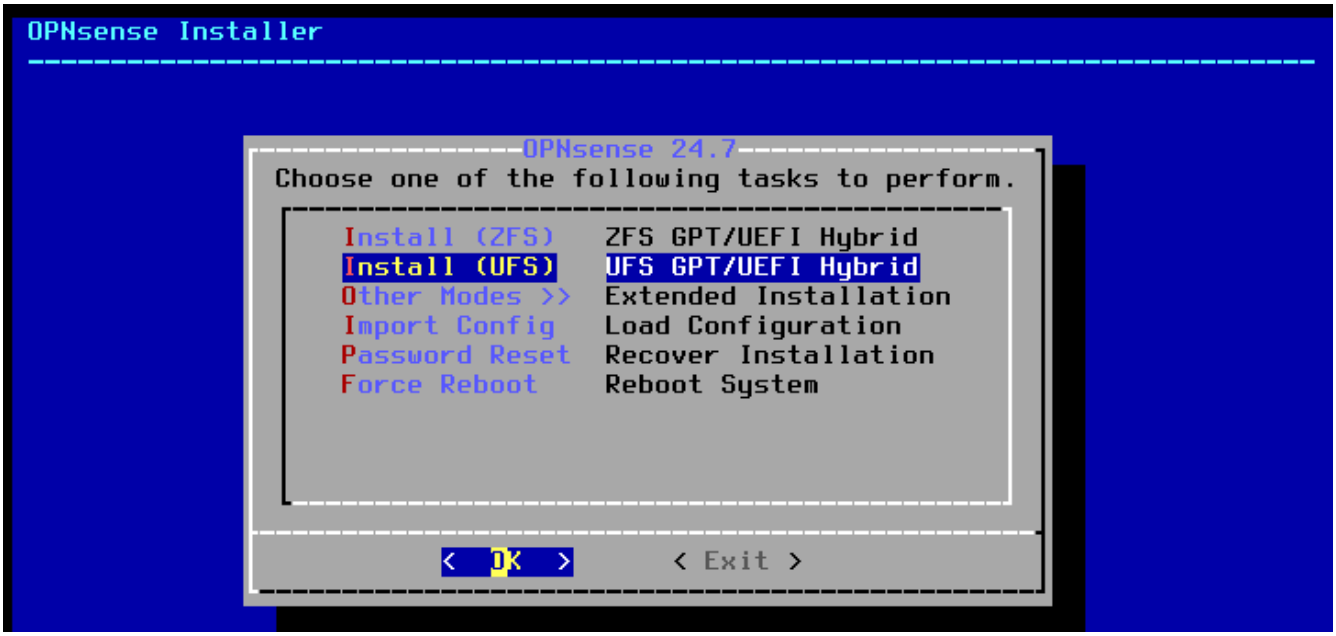


Figura 28: Configuración de OPNsense en progreso.

Elegimos el disco de 40GB que configuramos a nuestra máquina virtual.

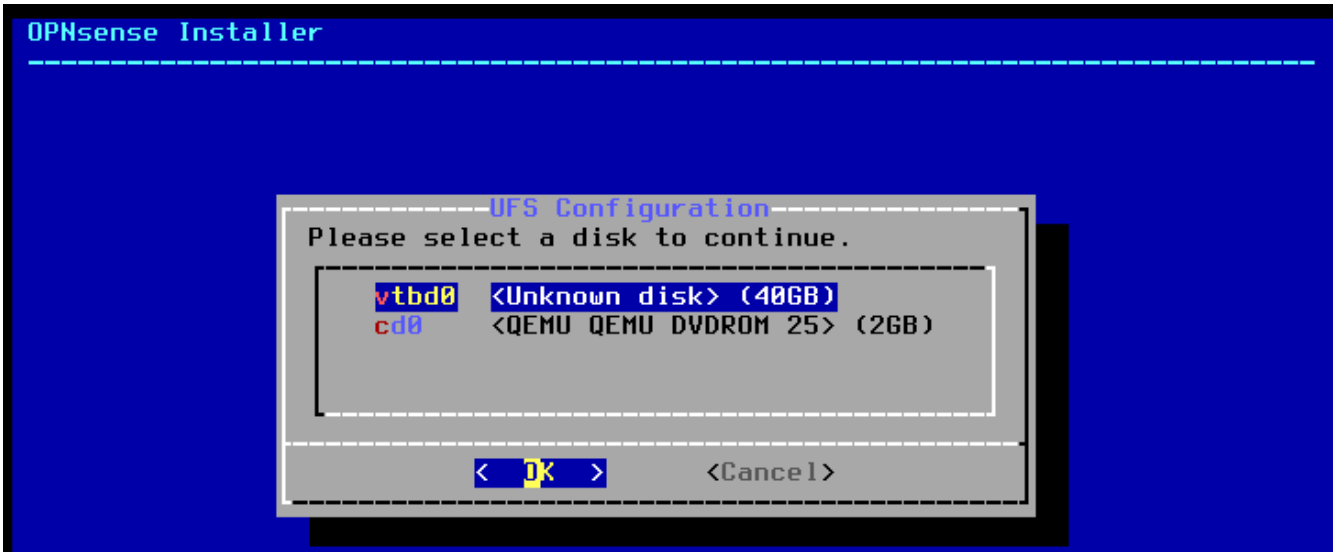


Figura 28: Configuración de OPNsense en progreso.

Presionamos enter sobre yes.

Figura 30: Configuración de OPNsense en progreso.

Nos preguntará si queremos eliminar el contenido actual del disco y le decimos “YES” y presionamos enter.

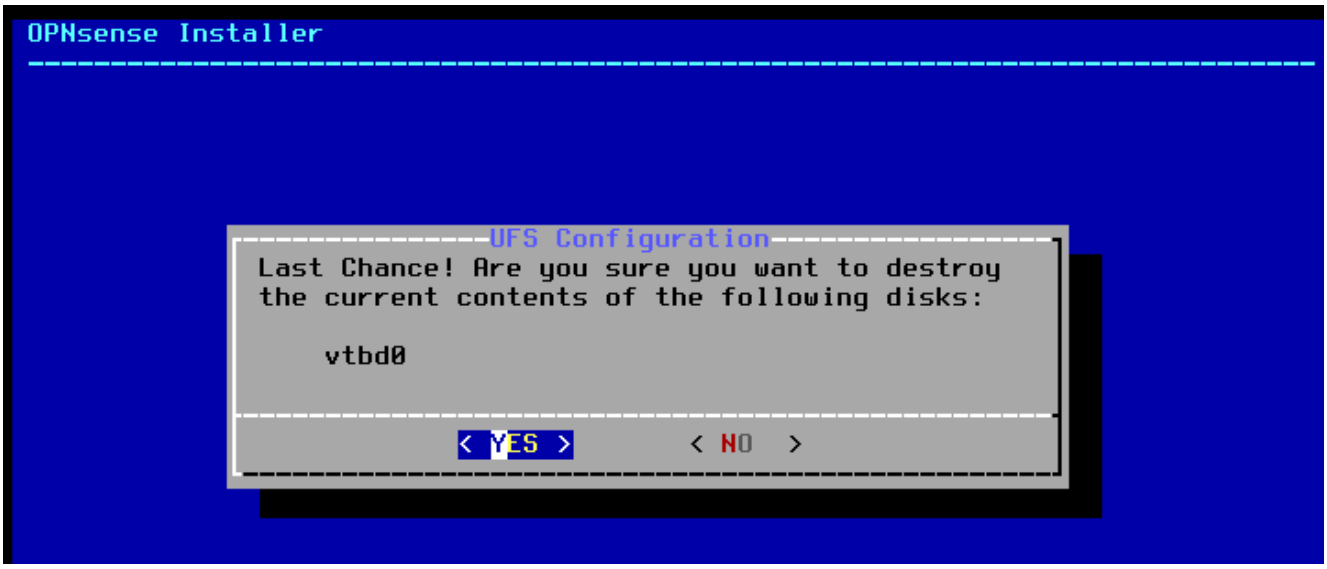


Figura 31: Configuración de OPNsense en progreso.

Comenzará a clonar el sistema operativo, tendremos que esperar a que finalice.

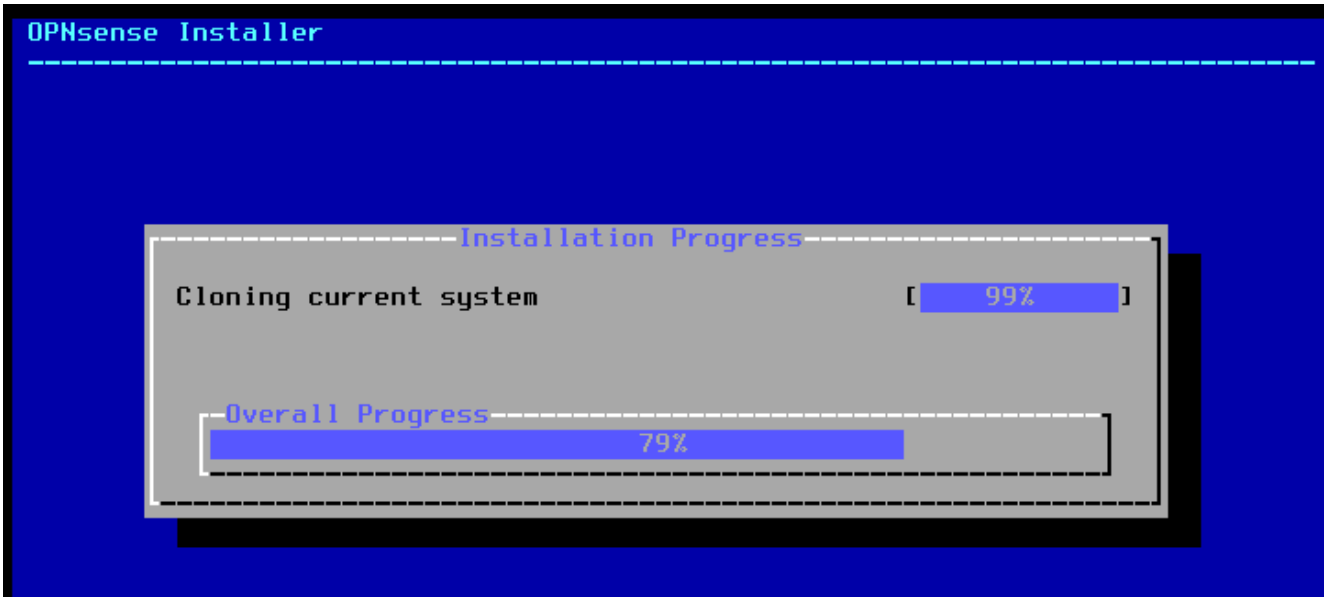


Figura 32: Configuración de OPNsense en progreso.

Ahora elegimos “Complete and install Exit and reboot” y presionamos enter.

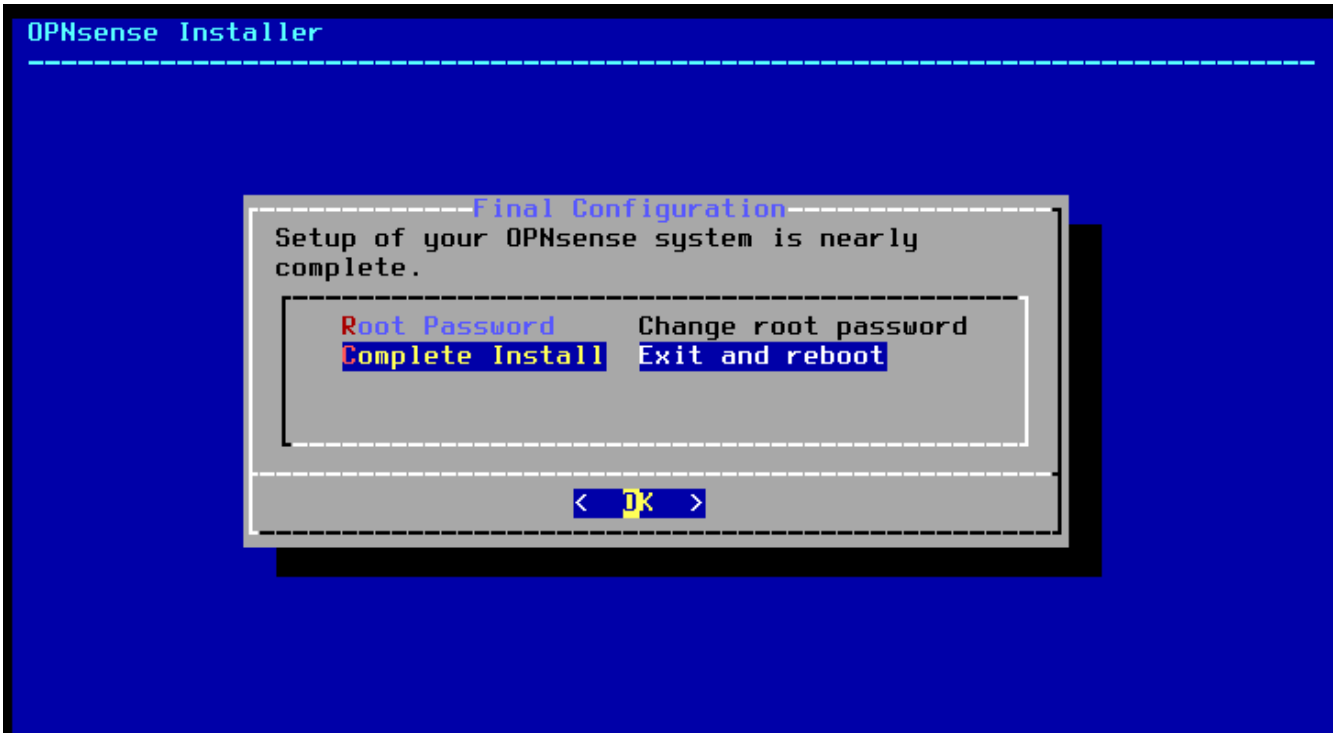


Figura 33: Configuración de OPNsense en progreso.

Ingresamos al sistema con el usuario “root” y contraseña “opnsense”.

Figura 34: Configuración de OPNsense en progreso.

Digitamos 1, seguidamente respondemos con “no” escribiendo la letra “n” en las dos preguntas que se nos muestran.

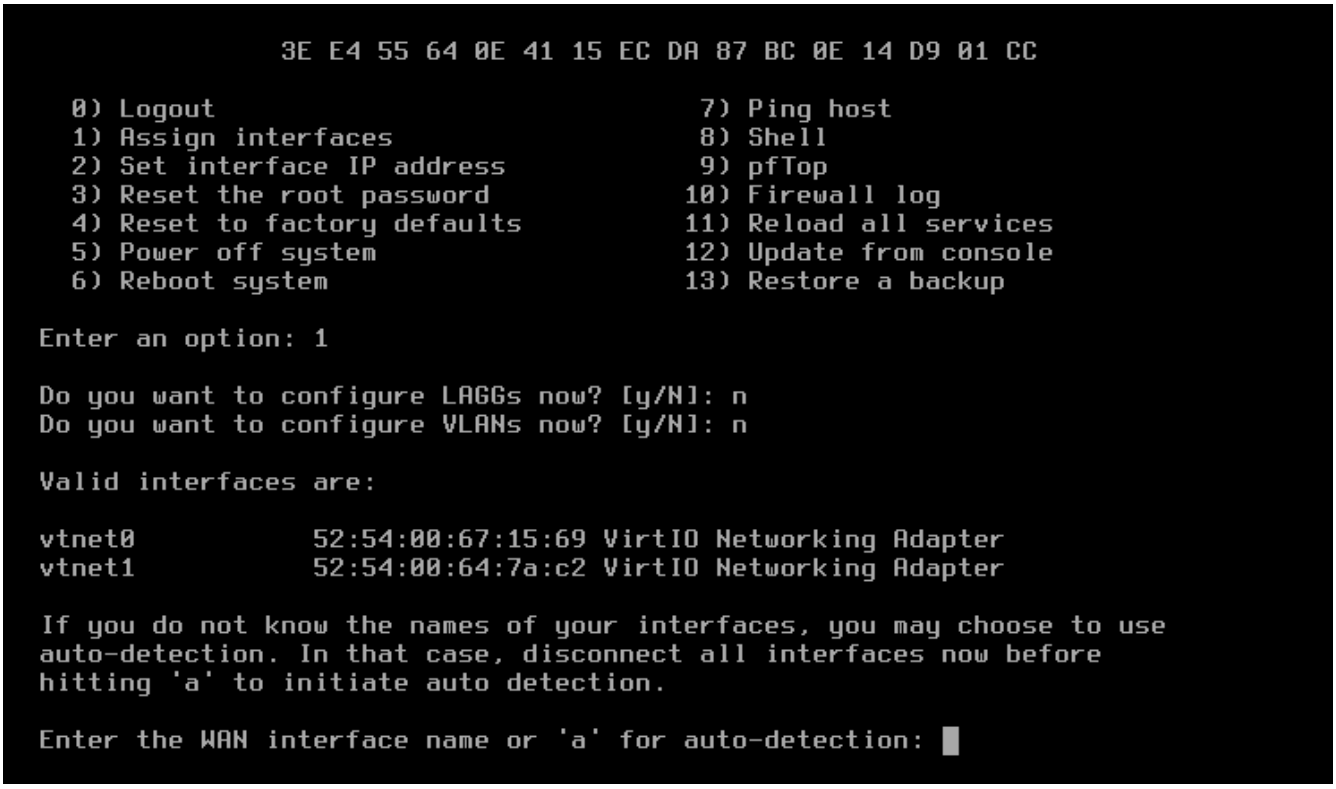


Figura 35: Configuración de OPNsense en progreso.

Configuramos WAN y LAN respectivamente de la siguiente manera.

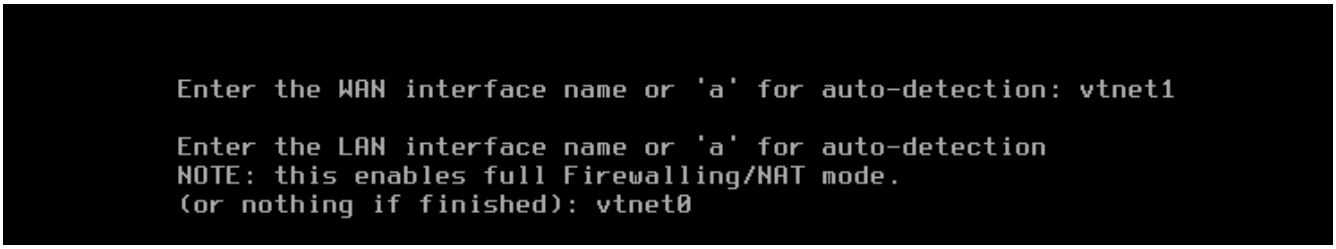


Figura 36: Configuración de OPNsense en progreso.

Digitamos “y” para aceptar la configuración

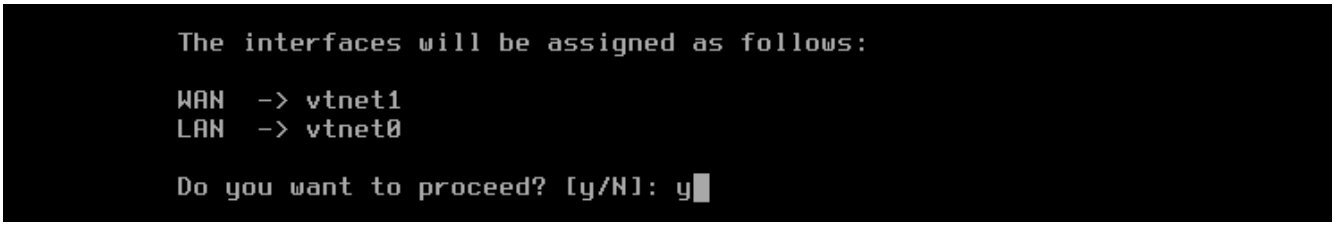


Figura 37: Configuración de OPNsense en progreso.

Digitamos 2 para configurar la interfaz 1, seguidamente digitamos 1 para configurar la interface LAN. Proseguimos también a configurar la IP 192.168.122.2. Definimos tambien "LAN IPv4 subnet" en 24 como se ve a continuación.

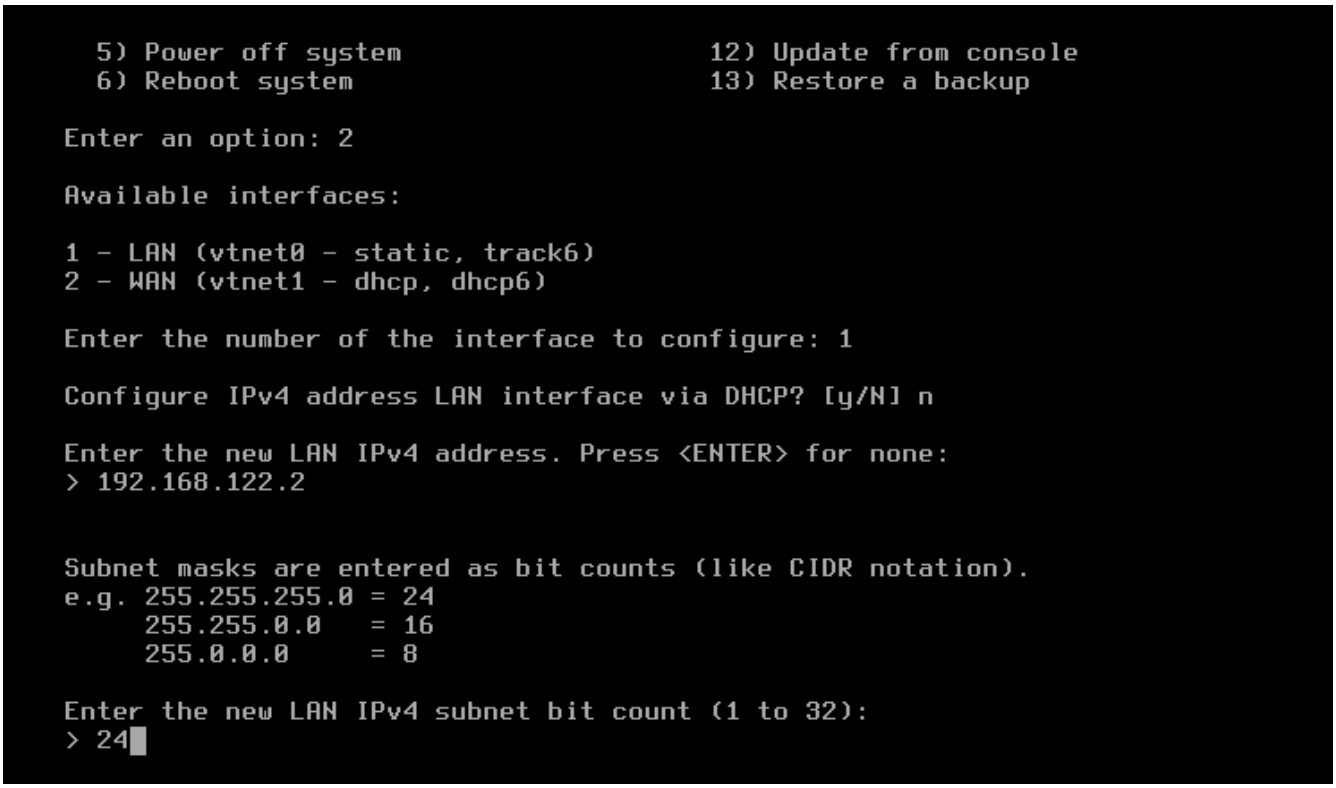


Figura 38: Configuración de OPNsense en progreso.

En la petición para la LAN por un "Ipv4 upstream gateway address" presionamos enter. En la parte de configuración de Ipv6 en ambas opciones digitamos "n" y en la solicitud de "new LAN Ipv6 address" presionamos enter. Cuando nos consulte si deseamos habilitar DHCP server en la LAN digitamos "y" para poder definir el rango de IP entre 192.168.122.10 – 192.168.122.200. Además, sobre el cambio de GUI protocol de HTTPS a HTTP digitamos "n", y para la generación de un nuevo "self-signed" certificado digitamos "y". Finalmente restauramos el "GUI access defaults" escribiendo "y" para confirmar y presionamos enter.

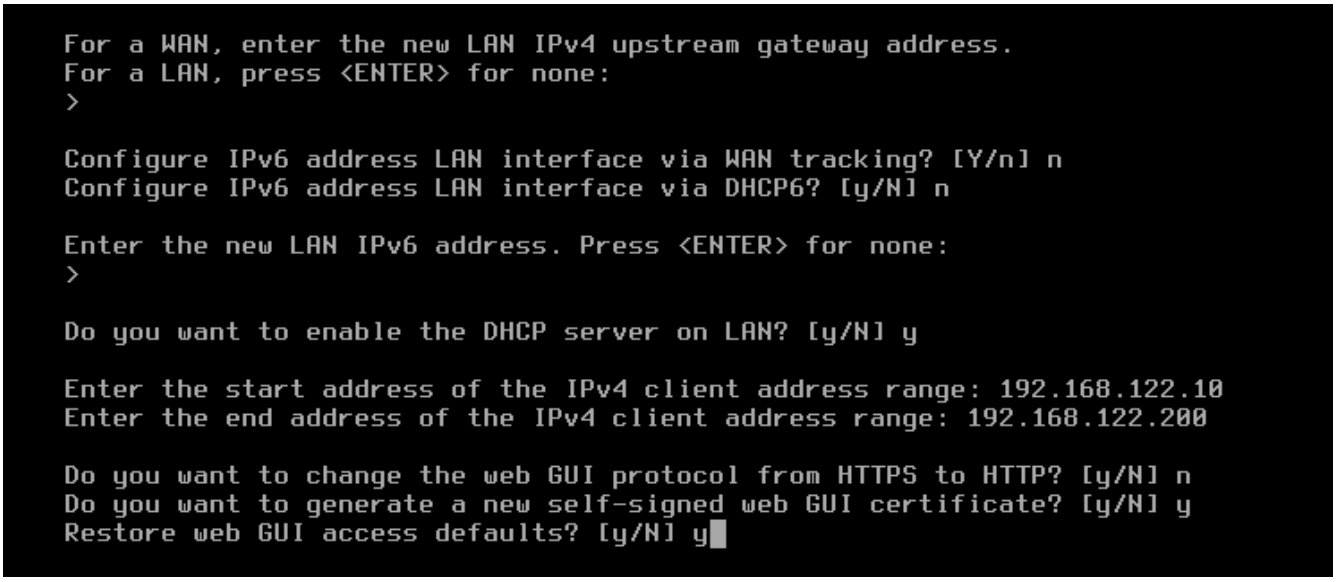


Figura 39: Configuración de OPNsense en progreso.

De esta manera ya podemos dirigirnos a nuestro navegador web y colocar la IP definida 192.168.122.2 y dar clic en la parte inferior en advance y clic en proceed. De esta manera ya nos mostrará el login de OPNsense donde podemos ingresar las credenciales de usuario “root” y contraseña “opnsense”.

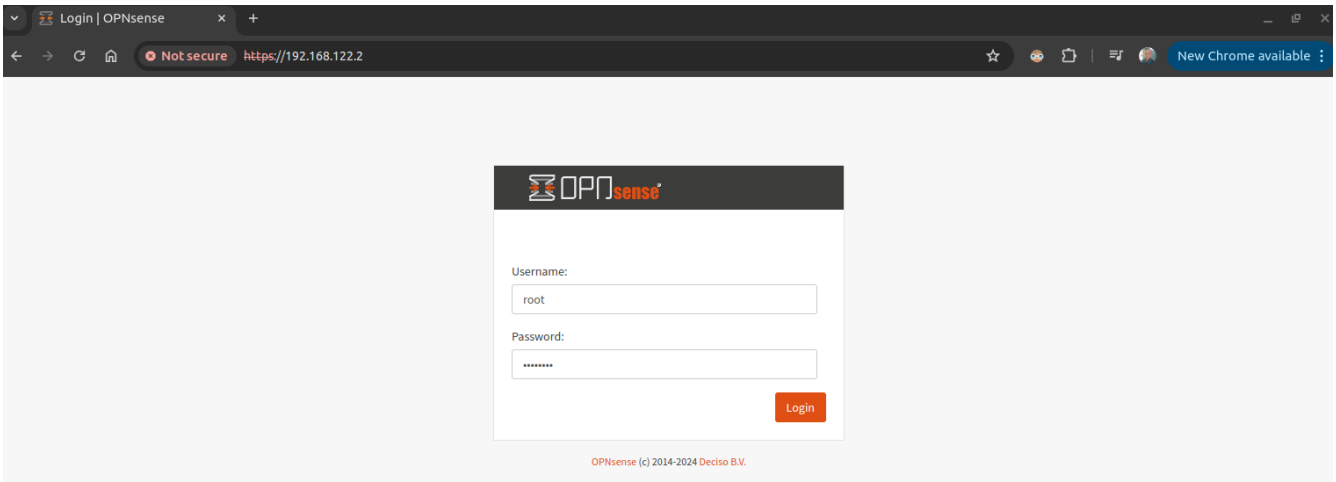


Figura 40: Configuración de OPNsense en progreso.

Seguidamente después de dar clic en login podemos ver el dashboard de OPNsense.

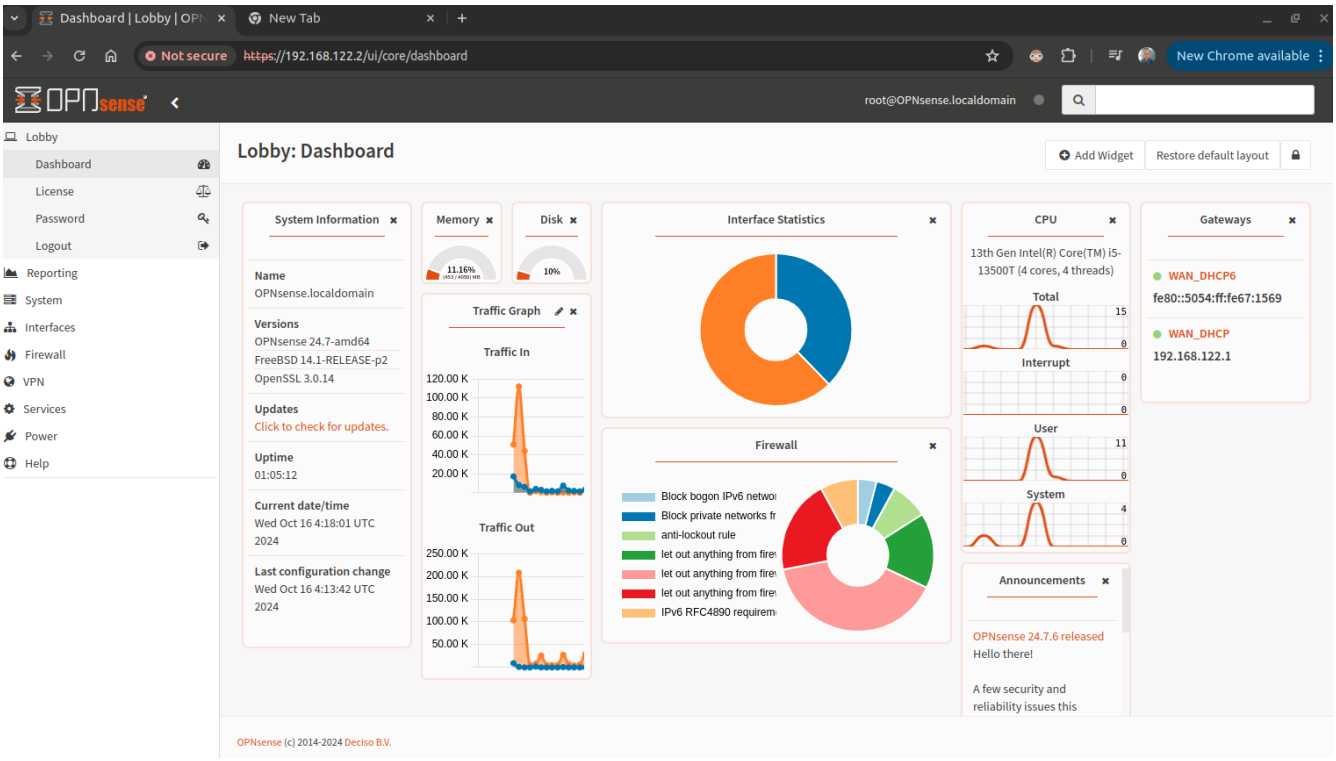


Figura 41: Configuración de OPNsense en progreso.

Instalación y configuración OpenStack

Requerimientos

Nodo	Nombre	Recursos
Controller 1 Región 1	Controller 1	Núcleos: 4 RAM: 8 GB NIC: 3 HDD: 100 GB
Compute 1 Región 1	Compute1	Núcleos: 10 RAM: 32 GB NIC: 3

		HDD: 100 GB
Compute 2 Región 1	Compute2	Núcleos: 10 RAM: 32 GB NIC: 3 HDD: 100 GB
Controller 2 Región 2	Controller 2	Núcleos: 5 RAM: 4 GB NIC: 3 HDD: 100GB
Compute 1 Región 2	Compute 1	Núcleos: 5 RAM: 10 GB NIC: 3 HDD: 100 GB
Compute 2 Región 2	Compute 2	Núcleos: 5 RAM: 10 GB NIC: 3 HDD: 100 GB

Asignación de IPs para el despliegue de prototipo

Host	**Región**	**Management**	**Provider**	**VPN**
Controller1	RegionOne	10.0.0.11	192.168.122.11	192.168.193.44
Compute1	RegionOne	10.0.0.31	192.168.122.31	

Compute2	RegionOne	10.0.0.32	192.168.122.32	192.168.193.214
Controller2	RegionTwo	10.0.0.12	192.168.122.12	192.168.193.141
Compute3	RegionTwo	10.0.0.33	192.168.122.33	192.168.193.93
Compute4	RegionTwo	10.0.0.34	192.168.122.34	192.168.193.114
Ceph-admin		192.168.3.50	192.168.122.100	192.168.193.232
Ceph-mon		192.168.3.51	192.168.122.101	192.168.193.205
Ceph-osd1		192.168.3.52	192.168.122.102	192.168.193.29
Ceph-osd2		192.168.3.53	192.168.122.103	192.168.193.40
Ceph-osd3		192.168.3.54	192.168.122.104	192.168.193.140
PC Region1	RegionOne			192.168.193.50
PC Region2	RegionTwo			192.168.193.20

Configuración del controller

El siguiente aparatado se muestra una serie de pasos para configurar la región uno y la región dos de la topología de red propuesta.

Para la configuración de todos los nodos involucrados en cada región primero se deben configurar máquinas virtuales en kvm, las cuales son controller, compute N, storage.

Configuración previa de las máquinas virtuales. Cuando se muestra un numeral # significa que se deben ejecutar los comandos como administrador. Cuando se muestra un signo de dólar $ se deben ejecutar los comandos como usuario normal.

Controller

Configuración inicial

Instalar Chrony

Se debe cambiar el hostname de la máquina virtual donde se instalarán los servicios del controller. Ejecutar los siguientes comandos.

```
# hostnamectl set-hostname controller
# exec bash
# apt update -y
# apt upgrade -y
```

Se deben agregar las direcciones IP de los nodos en el archivo *etc/*hosts.

```
# nano /etc/hosts
```

Luego agregar las siguientes líneas debajo de la IP 127.0.0.1.

```
10.0.0.11 controller
10.0.0.31 compute1
10.0.0.32 compute2
10.0.0.41 storage
```

Se debe configurar el archivo netplan para poder cambiar las direcciones IP de las máquinas. Al guardar el archivo ejecutar sudo netplan apply para guardar los cambios ejecutar el siguiente comando.

```
# nano /etc/netplan/00-installer-config.yaml
```

Agregar la siguiente configuración.

```
network:
  ethernets:
    enp1s0:
```

```
      dhcp4: false
      addresses: [192.168.122.12/24]
      gateway4: 192.168.122.1
      nameservers:
        addresses: [8.8.8.8,8.8.4.4]
    enp7s0:
      dhcp4: false
      addresses: [10.0.0.12/24]
    enp8s0:
      dhcp4: false
      addresses: [192.168.122.17/24]
      nameservers:
        addresses: [8.8.8.8,8.8.4.4]
    ztbtoxhtbp:
      addresses: [192.168.193.141/24]
      nameservers:
        addresses: [8.8.8.8,8.8.4.4]
  version: 2
```

Se debe instalar el servicio de chrony para sincronizar el reloj de los nodos compute con el nodo controller.

```
# apt install chrony –y
```

Se debe configurar el archivo de chrony localizado en /etc/chrony/chrony.conf.

```
# nano etc/chrony/chrony.conf
```

Agregar la siguiente línea en el archivo de configuración.

```
server ntp.ues.edu.sv iburst
```

Luego de guardar el archivo ejecutar.

```
# service chrony restart
```

Instalar la versión de openstack y openstack client. Ejecutar los siguientes comandos.

```
# add-apt-repository cloud-archive:bobcat
```

```
# apt install python3-openstackclient
```

Instalar la base de datos donde se almacenarán todos los servicios del controller. Ejecutar el siguiente comando.

```
# apt install mariadb-server python3-pymysql
```

Ejecutar el siguiente comando.

```
# nano /etc/mysql/mariadb.conf.d/99-openstack.cnf
```

Dentro de este mismo archivo agregar la siguiente línea. 10.0.0.11 es la IP management del controller.

```
[mysqld]
bind-address = 10.0.0.11

default-storage-engine = innodb
innodb_file_per_table = on
max_connections = 4096
collation-server = utf8_general_ci
character-set-server = utf8
```

Finalmente ejecutar.

```
# service mysql restart
```

Instalar RabbitMQ

Ejecutar para agregar el servicio de mensajería.

```
# apt install rabbitmq-server

# rabbitmqctl add_user openstack icc1152024

# rabbitmqctl set_permissions openstack ".*" ".*" ".*"
```

Instalar Memcached

Para poder instalar el servicio de memcached se debe ejecutar.

```
# apt install memcached python3-memcache
```

Configurar el archivo /etc/memcached.conf.

```
# nano /etc/memcached.conf
```

Buscar “-l 127.0.0.1” y reemplazarlo con -l 10.0.0.11. luego guardar y ejecutar el siguiente comando.

```
# service memcached restart
```

Instalar el servicio Etcd.

```
# apt install etcd
```

Luego configurar el archivo /etc/default/etcd.

```
# nano /etc/default/etcd
```

Agregar la siguiente configuración.

```
ETCD_NAME="controller"
ETCD_DATA_DIR="/var/lib/etcd"
ETCD_INITIAL_CLUSTER_STATE="new"
ETCD_INITIAL_CLUSTER_TOKEN="etcd-cluster-01"
ETCD_INITIAL_CLUSTER="controller=http://10.0.0.11:2380"
ETCD_INITIAL_ADVERTISE_PEER_URLS="http://10.0.0.11:2380"
ETCD_ADVERTISE_CLIENT_URLS="http://10.0.0.11:2379"
ETCD_LISTEN_PEER_URLS="http://0.0.0.0:2380"
ETCD_LISTEN_CLIENT_URLS="http://10.0.0.11:2379"
```

Finalizar la instalación ejecutando.

```
systemctl enable etcd
systemctl restart etcd
```

Configuración de paquetes

Keystone

Paquete que permitirá la autenticación de usuarios. Se debe ejecutar.

```
# mysql
```

Crear las bases de datos y otorgar accesos.

```
CREATE DATABASE keystone;

GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'localhost'
IDENTIFIED BY 'icc1152024';

GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'%'  IDENTIFIED
BY 'icc1152024';
```

Una vez se ha configurado la base de datos, se debe instalar keystone.

```
# apt install keystone
```

Configurar el archivo /etc/keystone/keystone.conf.

```
# nano /etc/keystone/keystone.conf
```

Buscar las secciones entre corchetes con ctrl + W y reemplazar la información y guardar.

```
[database]
connection =
mysql+pymysql://keystone:icc1152024@controller/keystone

[token]
```

```
provider = fernet
```

Poblar la base de datos.

```
su -s /bin/sh -c "keystone-manage db_sync" keystone
```

Inicializar los repositorios fernet.

```
# keystone-manage fernet_setup --keystone-user keystone --
keystone-group keystone
# keystone-manage credential_setup --keystone-user keystone --
keystone-group keystone
```

Crear endpoint de autenticación.

```
# keystone-manage bootstrap --bootstrap-password icc1152024
  --bootstrap-admin-url http://controller:5000/v3/
  --bootstrap-internal-url http://controller:5000/v3/
  --bootstrap-public-url http://controller:5000/v3/
  --bootstrap-region-id RegionOne
```

Configurar el nombre de servidor de apache 2.

```
# nano /etc/apache2/apache2.conf
```

Agregar las siguientes lineas.

```
ServerName controller
```

Finalizar la instalación reiniciando apache2.

```
# service apache2 restart
```

Crear archivo de variables.

```
# nano admin-openrc
```

Agregar lo siguiente.

```
export OS_REGION_NAME=RegionOne
export OS_PROJECT_DOMAIN_NAME=Default
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=admin
export OS_USERNAME=admin
export OS_PASSWORD=icc1152024
export OS_AUTH_URL=http://controller:5000/v3
export OS_IDENTITY_API_VERSION=3
export OS_IMAGE_API_VERSION=2
```

Cargar las variables de entorno.

```
. admin-openrc
```

Crear el domino, proyecto, usuario y roles ejecutando los siguientes comandos. Ejecutar en secuencia los siguientes comandos.

```
$ openstack domain create --description "An Example Domain"
example

$ openstack project create --domain default   --description
"Service Project" service

$ openstack project create --domain default   --description
"Demo Project" myproject

$ openstack user create --domain default   --password-prompt
myuser

$ openstack role create myrole

$ openstack role add --project myproject --user myuser myrole
```

Glance

Servicio que permitirá la subida de imágenes.

```
# mysql
```

Crear las bases de datos y otorgar accesos.

```
CREATE DATABASE glance;

GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'localhost'
IDENTIFIED BY 'icc1152024';

GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'%'  IDENTIFIED BY
'icc1152024';
```

Cargar las variables de entorno.

```
$ . admin-openrc
```

Crear el usuario glance, asignar el rol y agregar los endpoints para ser consumidos.

```
$ openstack user create --domain default --password-prompt
glance

$ openstack role add --project service --user glance admin

$ openstack service create --name glance   --description
"OpenStack Image" image

$ openstack endpoint create --region RegionOne   image public
http://controller:9292

$ openstack endpoint create --region RegionOne   image admin
http://controller:9292

$ openstack endpoint create --region RegionOne   image internal
http://controller:9292
```

Iniciar la instalación de glance.

```
# apt install glance
```

Configurar el archivo /etc/glance/glance-api.conf.

```
# nano /etc/glance/glance-api.conf
```

Buscar las secciones entre corchetes con ctrl + W y reemplazar la información y guardar.

```
[DEFAULT]
use_keystone_limits = False
enabled_backends=fs:file

[database]

connection = mysql+pymysql://glance:icc1152024@controller/glance

[keystone_authtoken]

www_authenticate_uri  = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
region_name= RegionOne
project_domain_name = Default
user_domain_name = Default
project_name = service
username = glance
password = icc1152024


[paste_deploy]

flavor = keystone

[glance_store]

default_backend = fs

[fs]

filesystem_store_datadir = /var/lib/glance/images/

[oslo_limit]

auth_url = http://controller:5000
```

```
auth_type = password
user_domain_id = default
username = glance
system_scope = all
password = icc1152024
endpoint_id = 340be3625e9b4239a6415d034e98aace
region_name = RegionOne
```

Nota: en la sección [oslo_limit] la propiedad endpoint_id es el id que resulta del endopint public.

Agregar el role a glance.

```
$ openstack role add --user glance --user-domain Default --
system all reader
```

Poblar la base de datos.

```
# su -s /bin/sh -c "glance-manage db_sync" glance
```

Reiniciar glance.

```
service glance-api restart
```

Placement

Agregar la base de datos de placement al controller.

```
# mysql
```

Crear las bases de datos y otorgar accesos.

```
CREATE DATABASE placement;

GRANT ALL PRIVILEGES ON placement.* TO 'placement'@'localhost'
IDENTIFIED BY 'icc1152024';
```

```
GRANT ALL PRIVILEGES ON placement.* TO 'placement'@'%'
IDENTIFIED BY 'icc1152024';
```

Cargar las variables de entorno.

```
$ . admin-openrc
```

Crear el usuario de placement, asignar un rol a este y agregarlo a los servicios.

```
$ openstack user create --domain default --password-prompt
placement

$ openstack role add --project service --user placement admin

$ openstack service create --name placement   --description
"Placement API" placement
```

Crear los endpoints para placement.

```
$ openstack endpoint create --region RegionOne   placement
public http://controller:8778

$ openstack endpoint create --region RegionOne   placement
internal http://controller:8778

$ openstack endpoint create --region RegionOne   placement admin
http://controller:8778
```

Instalar placement.

```
# apt install placement-api
```

Configurar el archivo de placement /etc/placement/placement.conf.

```
# nano /etc/placement/placement.conf
```

Buscar las secciones entre corchetes con ctrl + W y reemplazar la información y guardar.

```
[placement_database]

connection =
mysql+pymysql://placement:icc1152024@controller/placement

[api]

auth_strategy = keystone

[keystone_authtoken]

auth_url = http://controller:5000/v3
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
region_name=RegionOne
project_name = service
username = placement
password = icc1152024
```

Poblar la base de datos.

```
# su -s /bin/sh -c "placement-manage db sync" placement
```

Finalizar la instalación.

```
service apache2 restart
```

Compute Service

Agregar las bases de datos de nova al controller.

```
# mysql
```

Crear las bases de datos y otorgar accesos.

```
CREATE DATABASE nova_api;

CREATE DATABASE nova;
```

```
CREATE DATABASE nova_cell0;

GRANT ALL PRIVILEGES ON nova_api.* TO 'nova'@'localhost'
IDENTIFIED BY 'icc1152024';
GRANT ALL PRIVILEGES ON nova_api.* TO 'nova'@'%'  IDENTIFIED BY
'icc1152024';

GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'localhost'  IDENTIFIED
BY 'icc1152024';
GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'%'  IDENTIFIED BY
'icc1152024';

GRANT ALL PRIVILEGES ON nova_cell0.* TO 'nova'@'localhost'
IDENTIFIED BY 'icc1152024';
GRANT ALL PRIVILEGES ON nova_cell0.* TO 'nova'@'%'  IDENTIFIED
BY 'icc1152024';
```

Cargar las variables de entorno.

```
$ . admin-openrc
```

Crear el usuario de nova y agregarlo al proyecto con su rol.

```
$ openstack user create --domain default --password-prompt nova
$ openstack role add --project service --user nova admin
$ openstack service create --name nova   --description
"OpenStack Compute" compute
```

Crear los endpoints para nova.

```
$ openstack endpoint create --region RegionOne  compute public
http://controller:8774/v2.1

$ openstack endpoint create --region RegionOne  compute admin
http://controller:8774/v2.1

$ openstack endpoint create --region RegionOne  compute internal
http://controller:8774/v2.1
```

Instalar los paquetes de nova.

```
# apt install nova-api nova-conductor nova-novncproxy nova-
scheduler
```

Configurar el archivo /etc/nova/nova.conf.

```
# nano /etc/nova/nova.conf
```

Buscar las secciones entre corchetes con ctrl + W y reemplazar la información y guardar.

```
[DEFAULT]

transport_url = rabbit://openstack:icc1152024@controller:5672/
my_ip = 10.0.0.11

[api_database]

connection = mysql+pymysql://nova:icc1152024@controller/nova_api

[database]

connection = mysql+pymysql://nova:icc1152024@controller/nova

[api]

auth_strategy = keystone

[keystone_authtoken]

www_authenticate_uri = http://controller:5000/
auth_url = http://controller:5000/
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = nova
region_name=RegionOne
password = icc1152024

[service_user]
send_service_user_token = true
auth_url = https://controller/identity
auth_strategy = keystone
region_name=RegionOne
auth_type = password
```

```
project_domain_name = Default
project_name = service
user_domain_name = Default
username = nova
password = icc1152024

[vnc]
enabled = true
server_listen = $my_ip
server_proxyclient_address = $my_ip

[glance]
api_servers = http://controller:9292

[oslo_concurrency]
lock_path = /var/lib/nova/tmp

[placement]
region_name = RegionOne
project_domain_name = Default
project_name = service
auth_type = password
user_domain_name = Default
auth_url = http://controller:5000/v3
username = placement
password = icc1152024
```

Poblar las 3 bases de datos

```
# su -s /bin/sh -c "nova-manage api_db sync" nova
# su -s /bin/sh -c "nova-manage cell_v2 map_cell0" nova
# su -s /bin/sh -c "nova-manage db sync" nova
```

Finalizar la instalación reiniciando los servicios.

```
service nova-api restart
service nova-scheduler restart
service nova-conductor restart
service nova-novncproxy restart
```

Networking Service

Agregar la base de datos de neutron al controller.

```
# mysql
```

Crear las bases de datos y otorgar accesos.

```
CREATE DATABASE neutron;

GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'localhost'
IDENTIFIED BY 'icc1152024';

GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'%'  IDENTIFIED
BY 'icc1152024';
```

Cargar las variables de entorno.

```
. admin-openrc
```

Crear el usuario de neutron, agregar su rol y asignarlo a un grupo.

```
$ openstack user create --domain default --password-prompt
neutron

$ openstack role add --project service --user neutron admin

$ openstack service create --name neutron   --description
"OpenStack Networking" network
```

Crear los endpoints de neutron.

```
$ openstack endpoint create --region RegionOne  network public
http://controller:9696

$ openstack endpoint create --region RegionOne  network admin
http://controller:9696

$ openstack endpoint create --region RegionOne  network internal
http://controller:9696
```

Para el presente proyecto se optó por Self Service provider utilizando el ml2 plugin

instalar openvswitch.

```
# apt install neutron-server neutron-plugin-ml2 neutron-
openvswitch-agent neutron-l3-agent neutron-dhcp-agent  neutron-
metadata-agent
```

Configurar el archivo /etc/neutron/neutron.conf.

```
# nano /etc/neutron/neutron.conf
```

Buscar las secciones entre corchetes con ctrl + W y reemplazar la información y guardar.

```
[DEFAULT]
transport_url = rabbit://openstack:icc1152024@controller
core_plugin = ml2
service_plugins = router
auth_strategy = keystone
notify_nova_on_port_status_changes = true
notify_nova_on_port_data_changes = true

[database]

connection =
mysql+pymysql://neutron:icc1152024@controller/neutron


[keystone_authtoken]

www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
region_name= RegionOne
project_domain_name = Default
user_domain_name = Default
project_name = service
username = neutron
password = icc1152024

[nova]

auth_url = http://controller:5000
auth_type = password
project_domain_name = Default
```

```
user_domain_name = Default
region_name = RegionOne
project_name = service
username = nova
password = icc1152024

[oslo_concurrency]

lock_path = /var/lib/neutron/tmp
```

Configurar el archivo /etc/neutron/plugins/ml2/ml2_conf.ini.

```
# nano /etc/neutron/plugins/ml2/ml2_conf.ini
```

Buscar las secciones entre corchetes con ctrl + W y reemplazar la información y guardar.

```
[ml2]
type_drivers = flat,vlan,vxlan
tenant_network_types = vxlan
mechanism_drivers = openvswitch,l2population
extension_drivers = port_security
vni_ranges = 1:1000
```

Configurar el archivo /etc/neutron/plugins/ml2/openvswitch_agent.ini.

```
# nano /etc/neutron/plugins/ml2/openvswitch_agent.ini
```

Buscar las secciones entre corchetes con ctrl + W y reemplazar la información y guardar.

```
[ovs]

bridge_mappings = provider:br-provider

[vxlan]
local_ip = 10.0.0.11
l2_population = true

[agent]
tunnel_types = vxlan
l2_population = true
```

```
[securitygroup]
enable_security_group = true
firewall_driver = openvswitch
```

Configurar el archivo /etc/neutron/l3_agent.ini.

```
# nano /etc/neutron/l3_agent.ini
```

Buscar las secciones entre corchetes con ctrl + W y reemplazar la información y guardar.

```
[DEFAULT]

interface_driver = openvswitch
```

Configurar el archivo /etc/neutron/dhcp_agent.ini.

```
# nano /etc/neutron/dhcp_agent.ini
```

Buscar las secciones entre corchetes con ctrl + W y reemplazar la información y guardar.

```
[DEFAULT]
interface_driver = openvswitch
dhcp_driver = neutron.agent.linux.dhcp.Dnsmasq
enable_isolated_metadata = true
```

Configurar el archivo /etc/neutron/metadata_agent.ini.

```
# nano /etc/neutron/metadata_agent.ini
```

Buscar las secciones entre corchetes con ctrl + W y reemplazar la información y guardar.

```
[DEFAULT]
nova_metadata_host = controller
metadata_proxy_shared_secret = METADATA_SECRET
```

Configurar el archivo /etc/nova/nova.conf.

```
# nano /etc/nova/nova.conf
```

Buscar las secciones entre corchetes con ctrl + W y reemplazar la información y guardar.

```
[neutron]
auth_url = http://controller:5000
auth_type = password
project_domain_name = Default
user_domain_name = Default
region_name = RegionOne
project_name = service
username = neutron
password = icc1152024
service_metadata_proxy = true
metadata_proxy_shared_secret = METADATA_SECRET
```

Poblar la base de datos.

```
#  su -s /bin/sh -c "neutron-db-manage --config-file
/etc/neutron/neutron.conf   --config-file
/etc/neutron/plugins/ml2/ml2_conf.ini upgrade head" neutron
```

Reiniciar los servicios.

```
# service nova-api restart
# service neutron-server restart
# service neutron-openvswitch-agent restart
# service neutron-dhcp-agent restart
# service neutron-metadata-agent restart
# service neutron-l3-agent restart
```

Cinder

Servicio de almacenamiento en bloques de OpenStack, que proporciona volúmenes persistentes para instancias de máquinas virtuales.

Conectarse como usuario root:

```
Mysql
```

Crear la base de datos Cinder con la siguiente sentencia:

```
CREATE DATABASE cinder;
```

Otorgar los privilegios necesarios a la base de datos:

```
GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'localhost'
IDENTIFIED BY 'CINDER_DBPASS'; GRANT ALL PRIVILEGES ON cinder.*
TO 'cinder'@'%' IDENTIFIED BY 'CINDER_DBPASS';
```

Nota: Reemplazar CINDER_DBPASS con una contraseña adecuada.

Salir del cliente de acceso a la base de datos.

Obtener las credenciales de administrador necesarias para acceder a los comandos de CLI:

```
$ . admin-openrc
```

Para crear las credenciales del servicio, ejecutar los siguientes pasos:

Crear un usuario para Cinder con el siguiente comando, agregar la contraseña cuando se te solicite:

```
$ openstack user create --domain default --password-prompt
cinder
```

Se debe de agregar el rol de administrador al usuario Cinder:

```
$ openstack role add --project service --user cinder admin
```

Crear la entidad de servicio Cinder:

```
$ openstack service create --name cinderv3 --description
"OpenStack Block Storage" volumev3
```

Crear los puntos finales de API para Cinder:

```
$ openstack endpoint create --region RegionOne volumev3 public
http://controller:8776/v3/%\(project_id\)s
$ openstack endpoint create --region RegionOne volumev3
internal http://controller:8776/v3/%\(project_id\)s
$ openstack endpoint create --region RegionOne volumev3 admin
http://controller:8776/v3/%\(project_id\)s
```

Se debe de ejecutar el siguiente comando para instalar Cinder:

```
apt install cinder-api cinder-scheduler
```

Editar el archivo /etc/cinder/cinder.conf y realizar las siguientes configuraciones

En la sección **[database]**, añadir la conexión a la base de datos:

```
[database]
# ...
connection =mysql+pymysql://cinder:CINDER_DBPASS@controller/cinder
```

Nota: reemplazar CINDER_DBPASS, con la contraseña que se eligió al crear la base de datos.

En la sección [DEFAULT], se debe de configurar RabbitMQ que es el acceso a la cola de mensajes:

```
[DEFAULT]
# ...
transport_url = rabbit://openstack:RABBIT_PASS@controller
```

Nota: se debe de reemplazar RABBIT_PASS por la contraseña que se eligió para OpenStack.

Configurar el acceso de servicio de identidad estos se hacen en las secciones [DEFAULT] y [keystone_authtoken].

```
[DEFAULT]
auth_strategy = keystone

[keystone_authtoken]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
```

```
project_domain_name = default
user_domain_name = default
project_name = service
username = cinder
password = CINDER_PASS
```

Nota: se debe de reemplazar CINDER_PASS con la clave para el usuario de identidad de cinder.

En la sección [DEFAULT], se configuró la opción de my_ip para utilizar la dirección IP de la interfaz de administración del nodo controlador:

```
[DEFAULT]
my_ip = 10.0.0.11
```

Configuración de la ruta de bloqueo, esto se agrega en la sección de [oslo_concurrency].

```
[oslo_concurrency]
lock_path = /var/lib/cinder/tmp
```

Poblar la base de datos con el siguiente comando.

```
# su -s /bin/sh -c "cinder-manage db sync" cinder
```

Se editó el archivo /etc/nova/nova.conf con la siguiente configuración.

```
[cinder]
os_region_name = RegionOne
```

Se debe reiniciar los siguientes servicios.

```
# service nova-api restart
# service cinder-scheduler restart
# service apache2 restart
```

Trove

A continuación se describe como realizar la instalación y configuración del servicio de base de datos Trove, antes de seguir con el proceso se debe de asegurar que cumpla con los prerrequisitos.

Se debe crear una base de datos, credenciales del servicio y puntos finales de API.

Se debe de conectar al servidor de base de datos como usuario root.

```
Mysql
```

Ejecutar el siguiente script para crear la base de datos trove.

```
CREATE DATABASE trove;
```

Seguidamente se debe otorgar los permisos necesarios a la base de datos trove.

```
GRANT ALL PRIVILEGES ON trove.* TO 'trove'@'localhost' \
  IDENTIFIED BY 'TROVE_DBPASS';
GRANT ALL PRIVILEGES ON trove.* TO 'trove'@'%' \
  IDENTIFIED BY 'TROVE_DBPASS';
```

Nota: se debe de reemplazar **TROVE_DBPASS** con una clave segura.

Salir del cliente de acceso a la base de datos.

```
Exit
```

Para poder acceder a los comandos CLI exclusivos para administradores se deben de cargar las credenciales con el siguiente comando.

```
$ . admin-openrc
```

A continuación, se deben de crear las credenciales para el servicio Trove.

Creación de usuario trove.

```
$ openstack user create --domain default --password-prompt
trove
```

Nota: se debe de introducir y confirmar la credencial que se solicite.

Añadir el rol admin al usuario trove:

```
$ openstack role add --project service --user trove admin
```

Ahora se procede a crear la entidad del servicio para Trove.

```
$ openstack service create --name trove \ --description
"Database" database
```

Configuración de los puntos finales de la API para el servicio de base de datos

Creación del punto final público.

```
$ openstack endpoint create --region RegionOne \ database
public http://controller:8779/v1.0/%\(tenant_id\)s
```

Creación del punto final interno.

```
$ openstack endpoint create --region RegionOne \
  database internal http://controller:8779/v1.0/%\(tenant_id\)s
```

Creación del punto final de administración.

```
$ openstack endpoint create --region RegionOne \
  database admin http://controller:8779/v1.0/%\(tenant_id\)s
```

Debe de instalar los siguientes paquetes.

```
# apt-get update
# apt-get install python-trove trove-common trove-api trove-
taskmanager trove-conductor
# pip3 install python-troveclient
```

Luego de haber realizado la instalación de los paquetes debe de configurar el siguiente archive, **/etc/trove/trove.conf**.

```
[DEFAULT]
```

```
network_driver = trove.network.neutron.NeutronDriver
management_networks = ef7541ad-9599-4285-878a-e0ab62032b03
management_security_groups = d0d797f7-11d4-436e-89a3-
ac8bca829f81
cinder_volume_type = lvmdriver-1
nova_keypair = trove-mgmt
default_datastore = mysql
taskmanager_manager = trove.taskmanager.manager.Manager
trove_api_workers = 5
transport_url =
rabbit://stackrabbit:password@192.168.1.34:5672/
control_exchange = trove
reboot_time_out = 300
usage_timeout = 900
agent_call_high_timeout = 1200
use_syslog = False
debug = True


[keystone_authtoken]
memcached_servers = localhost:11211
cafile = /devstack/stack/data/ca-bundle.pem
project_domain_name = Default
project_name = service
user_domain_name = Default
password = password
username = trove
auth_url = http://192.168.1.34/identity
auth_type = password

[service_credentials]
auth_url = http://192.168.1.34/identity/v3
region_name = RegionOne
project_name = service
password = password
project_domain_name = Default
user_domain_name = Default
username = trove

[database]
connection =
mysql+pymysql://root:password@127.0.0.1/trove?charset=utf8

[mariadb]
tcp_ports = 3306,4444,4567,4568

[mysql]
tcp_ports = 3306
```

```
[postgresql]
tcp_ports = 5432
```

Se debe de editar el archivo /etc/trove/trove-guestagent.conf para que los agentes futuros de Trove pueda conectarse al entorno de OpenStack con la siguiente configuración.

```
[DEFAULT]
log_file = trove-guestagent.log
log_dir = /var/log/trove/
ignore_users = os_admin
control_exchange = trove
transport_url = rabbit://stackrabbit:password@172.24.5.1:5672/
command_process_timeout = 60
use_syslog = False
debug = True

[service_credentials]
auth_url = http://192.168.1.34/identity/v3
region_name = RegionOne
project_name = service
password = password
project_domain_name = Default
user_domain_name = Default
username = trove
```

Luego se debe de poblar la base de datos con el siguiente comando.

```
# su -s /bin/sh -c "trove-manage db_sync" trove
```

Reiniciar servicios.

```
# service trove-api restart
# service trove-taskmanager restart
# service trove-conductor restart
```

Magnum

Magnum facilita la creación y administración de clústeres de contenedores con tecnologías como kuberetes y Docker a continuación se muestra los pasos a realizar para configurar este servicio.

Prerrequisitos

Primero se necesita conectar al servidor como usuario root.

```
#  mysql
```

Luego, crear la base de datos que usará magnum con el siguiente comando:

```
CREATE DATABASE magnum;
```

A continuación, se debe de otorgar los permisos adecuados para acceder a la base de datos. Se debe de ejecutar el siguiente comando, tomar en cuenta que se debe de reemplazar MAGNUM_DBPASS por una contraseña segura.

```
GRANT ALL PRIVILEGES ON magnum.* TO 'magnum'@'localhost' IDENTIFIED BY
'MAGNUM_DBPASS';
GRANT ALL PRIVILEGES ON magnum.* TO 'magnum'@'%' IDENTIFIED BY
'MAGNUM_DBPASS';
```

Salir del cliente de MySQL.

```
Exit
```

Antes de continuar, se debe cargar las credenciales del administrador para poder usar los comandos de OpenStack:

```
$ . admin-openrc
```

Crear un usuario llamado magnum dentro del dominio predeterminado:

```
$ openstack user create --domain default \ --password-prompt magnum
```

Se debe de agregar y confirmar la contraseña del usuario. Después, se debe asginar el rol de administrador al usuario magnum en el proyecto service:

```
$ openstack role add --project service --user magnum admin
```

Se debe de crear la entidad de servicio magnum con el siguiente comando

```
$ openstack service create --name magnum \ --description "OpenStack Container
Infrastructure Management Service" \ container-infra
```

Crear los puntos de acceso a la API para el servicio con los siguientes commandos.

```
$ openstack endpoint create --region RegionOne \ container-infra public
http://CONTROLLER_IP:9511/v1
$ openstack endpoint create --region RegionOne \ container-infra internal
http://CONTROLLER_IP:9511/v1
$ openstack endpoint create --region RegionOne \ container-infra admin
http://CONTROLLER_IP:9511/v1
```

Magnum necesita información adicional en el servicio de identidad para gestionar los clústeres, para realizar esta configuración se debe de seguir los siguientes pasos:

Crear el dominio magnum que contenga Proyecto y usuarios

```
$ openstack domain create --description "Owns users and projects \ created by magnum"
magnum
```

Crear el usuario magnum_domain_admin para gestionar proyectos y usuarios en el dominio magnum.

```
$ openstack user create --domain magnum --password-prompt \ magnum_domain_admin
```

Seguidamente se debe asignar el rol de administrador al usuario magnum_domain_admin en el dominio magnum para habilitar privilegios de gestion administrativa.

```
$ openstack role add --domain magnum --user-domain magnum --user \
magnum_domain_admin admin
```

Instalar y configurar components

Se debe de instalar los paquetes communes y también las librerías.

```
# DEBIAN_FRONTEND=noninteractive apt-get install magnum-api magnum-conductor
python3-magnumclient
```

Modificar el archivo /etc/magnum/magnum.conf:

En la seccion [api], establece la dirección del host

```
[api]
...
```

```
host = IP_DEL_CONTROLADOR
```

Sustituye IP_DEL_CONTROLADOR por la dirección IP donde deseas que el aPI de magnum esté disponible.

Para usar barbican para almacenar certificados:

```
[certificates]
...
cert_manager_type = barbican
```

Nota: Barbican es preferable para entornos de producción

En la sección [cinder_client], se dede definir el nombre de la región:

```
[cinder_client]
...
region_name = RegionOne
```

Seguidamente en la sección de [database], se configura el acceso a la base de datos:

```
[database]
...
connection = mysql+pymysql://magnum:MAGNUM_DBPASS@controller/magnum
```

Nota: sustituir **MAGNUM_DBPASS** por la contraseña que elegiste para la base de datos de magnum.

En las secciones de [keystone_authtoken] y [trust], configura el acceso al servicio de identidad:

```
[keystone_authtoken]
...
memcached_servers = controller:11211
auth_version = v3
www_authenticate_uri = http://controller:5000/v3
project_domain_id = default
```

```
project_name = service
user_domain_id = default
password = MAGNUM_PASS
username = magnum
auth_url = http://controller:5000
auth_type = password
admin_user = magnum
admin_password = MAGNUM_PASS
admin_tenant_name = service

[trust]
...
trustee_domain_name = magnum
trustee_domain_admin_name = magnum_domain_admin
trustee_domain_admin_password = DOMAIN_ADMIN_PASS
trustee_keystone_interface = KEYSTONE_INTERFACE
```

Se sustituye CONTRASEÑA_MAGNUM por la contraseña que se elegió para el usuario de magnum en el servidor de identidad y CONTRASEÑA_ADMIN_DOMINIO por la contraseña que se establecio el usuario magum_domain_admin.

También sustituir INTERFAZ_KEYSTONE por **public** o **internal** según la configuración de la red. En la sección [oslo_messaging_notifications], configurar el controlador:

```
[oslo_messaging_notifications]

driver = messaging
```

En la sección [DEFAULT], configura el acceso a la cola de mensajes RabbitMQ:

```
[DEFAULT]

transport_url = rabbit://openstack:RABBIT_PASS@controller
```

Sustituir CONTRASEÑA_RABBIT por la contraseña que elegiste para cuenta openstack en RabbitMQ.

Seguido se debe de inicializar la base de datos Magnum:

```
# su -s /bin/sh -c "magnum-db-manage upgrade" magnum
```

Finalización de la instalación

Reiniciar los servicios de Gestión de Infraestructura de contenedores:

```
# service magnum-api restart

# service magnum-conductor restart
```

Heat

Para desplegar el servicio de orquestación se deben realizar las siguientes configuraciones.

Crear una base de datos mysql.

```
# mysql
```

Se deben crear las bases de datos que almacenaran toda la información del servicio de orquestación heat.

```
GRANT ALL PRIVILEGES ON heat.* TO 'heat'@'localhost'
IDENTIFIED BY icc1152024';
GRANT ALL PRIVILEGES ON heat.* TO 'heat'@'%'   IDENTIFIED BY
icc1152024';
```

Cargar las variables de entorno para crear los servicios.

```
$ . admin-openrc
```

Se debe crear el usuario de heat. Pedirá contraseña la cual será icc1152024.

```
$ openstack user create --domain default --password-prompt heat
```

Se debe asignar un rol al usuario.

```
$ openstack role add --project service --user heat admin
```

Se debe crear el servicio.

```
$ openstack service create --name heat  --description
"Orchestration" orchestration
$ openstack service create --name heat-cfn  --description
"Orchestration"  cloudformation
```

Crear los endpoint para el servicio de heat.

```
$ openstack endpoint create --region RegionOne  orchestration
public http://controller:8004/v1/%\(tenant_id\)s
$ openstack endpoint create --region RegionOne  orchestration
admin http://controller:8004/v1/%\(tenant_id\)s
$ openstack endpoint create --region RegionOne  orchestration
internal http://controller:8004/v1/%\(tenant_id\)s

$ openstack endpoint create --region RegionOne cloudformation
public http://controller:8000/v1
$ openstack endpoint create --region RegionOne cloudformation
admin http://controller:8000/v1
$ openstack endpoint create --region RegionOne cloudformation
internal http://controller:8000/v1
```

Configurar el dominio de Openstack para heat.

```
$ openstack domain create --description "Stack projects and
users" heat
$ openstack user create --domain heat --password-prompt
heat_domain_admin
$ openstack user create --domain heat --password-prompt
heat_domain_admin
$ openstack role create heat_stack_owner
$ openstack role add --project demo --user demo
heat_stack_owner
$ openstack role create heat_stack_user
```

Instalar los paquetes para activar heat.

```
# apt-get install heat-api heat-api-cfn heat-engine
```

Configurar el archivo localizado en **/etc/heat/heat.conf**.

```
# nano /etc/heat/heat.conf
```

Buscar y reemplazar las secciones en corchetes, sino existen agregarlas.

```
 [DEFAULT]
transport_url = rabbit://openstack:icc1152024@controller
heat_metadata_server_url = http://controller:8000
heat_waitcondition_server_url = http://controller:8000/v1/waitcondition
stack_domain_admin = heat_domain_admin
stack_domain_admin_password = icc1152024
stack_user_domain_name = heat

[database]
connection = mysql+pymysql://heat: :icc1152024@controller/heat

[keystone_authtoken]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = default
user_domain_name = default
project_name = service
username = heat
región_name = RegionOne
password = :icc1152024

[trustee]
auth_type = password
auth_url = http://controller:5000
username = heat
password = :icc1152024
user_domain_name = default

[clients_keystone]
auth_uri = http://controller:5000
```

Poblar la base de datos.

```
su -s /bin/sh -c "heat-manage db_sync" heat
```

Finalmente reiniciar los servicios

```
# service heat-api restart
# service heat-api-cfn restart
# service heat-engine restart
```

Desginate

En el nodo controller realizar las siguientes configuraciones. Se deben cargar las variables de entorno.

```
$ . admin-openrc
```

Se debe configurar el domino para el servicio de designate.

```
$ openstack user create --domain default --password-prompt
designate
$ openstack role add --project service --user designate admin
$ openstack service create --name designate --description "DNS"
dns
```

Se debe crear el respectivo endpoint para designate.

```
$ openstack endpoint create --region RegionOne  dns public
http://controller:9001/
```

Se deben instalar los paquetes de designate.

```
# apt-get install designate
```

Se debe crear la respectiva base de datos designate.

```
# mysql
CREATE DATABASE designate;
```

```
GRANT ALL PRIVILEGES ON designate.* TO 'designate'@'localhost'
IDENTIFIED BY icc1152024';
GRANT ALL PRIVILEGES ON designate.* TO 'designate'@'%'
IDENTIFIED BY icc1152024';
```

Instalar bind9.

```
# apt-get install bind9 bind9utils bind9-doc
```

Crear llave RNDC.

```
rndc-confgen -a -k designate -c /etc/designate/rndc.key -r
/dev/urandom
```

Configurar el archivo **/etc/bind/named.conf.options.**

```
include "/etc/designate/rndc.key";

options {
    ...
    allow-new-zones yes;
    request-ixfr no;
    listen-on port 53 { 127.0.0.1; };
    recursion no;
    allow-query { 127.0.0.1; };
};

controls {
  inet 127.0.0.1 port 953
    allow { 127.0.0.1; } keys { "designate"; };
};
```

Reiniciar bind9.

```
# systemctl restart bind9.service
```

Configurar el archivo localizado en /etc/designate/designate.conf.

```
# nano /etc/designate/designate.conf
```

Reemplazar p agregar las secciones siguientes.

```
[DEFAULT]
transport_url = rabbit://openstack:icc1152024@controller:5672/

[service:api]
listen = 0.0.0.0:9001
auth_strategy = keystone
enable_api_v2 = True
enable_api_admin = True
enable_host_header = True
enabled_extensions_admin = quotas, reports


[keystone_authtoken]
auth_type = password
username = designate
password = icc1152024
región_name = RegionOne
project_name = service
project_domain_name = Default
user_domain_name = Default
www_authenticate_uri = http://controller:5000/
auth_url = http://controller:5000/
memcached_servers = controller:11211

[storage:sqlalchemy]
connection =
mysql+pymysql://designate:icc1152024@controller/designate
```

Poblar la base de datos.

```
# su -s /bin/sh -c "designate-manage database sync" designate
```

Reiniciar los servicios.

```
#systemctl start designate-central designate-api
# systemctl enable designate-central designate-api
```

Crear el archivo /etc/designate/pools.yaml.

```
# nano /etc/designate/pools.yaml
```

```
- name: default
  # The name is immutable. There will be no option to change
the name after
  # creation and the only way will to change it will be to
delete it
  # (and all zones associated with it) and recreate it.
  description: Default Pool

  attributes: {}

  # List out the NS records for zones hosted within this pool
  # This should be a record that is created outside of
designate, that
  # points to the public IP of the controller node.
  ns_records:
    - hostname: ns1-1.example.org.
      priority: 1

  # List out the nameservers for this pool. These are the
actual BIND servers.
  # We use these to verify changes have propagated to all
nameservers.
  nameservers:
    - host: 127.0.0.1
      port: 53

  # List out the targets for this pool. For BIND there will be
one
  # entry for each BIND server, as we have to run rndc command
on each server
  targets:
    - type: bind9
      description: BIND9 Server 1

      # List out the designate-mdns servers from which BIND
servers should
      # request zone transfers (AXFRs) from.
      # This should be the IP of the controller node.
      # If you have multiple controllers you can add multiple
masters
      # by running designate-mdns on them, and adding them
here.
      masters:
        - host: 127.0.0.1
          port: 5354

      # BIND Configuration options
```

```
      options:
        host: 127.0.0.1
        port: 53
        rndc_host: 127.0.0.1
        rndc_port: 953
        rndc_key_file: /etc/designate/rndc.key
```

Actualizar los pools.

```
# su -s /bin/sh -c "designate-manage pool update" designate
```

Instalar designate worker.

```
# apt install designate-worker designate-producer designate-
mdns
```

Restart Services.

```
# systemctl start designate-worker designate-producer
designate-mdns
# systemctl enable designate-worker designate-producer
designate-mdns
```

Octavia

A continuación, se describe cómo instalar y configurar el servicio de balanceador de carga para Ubuntu 22.04 (LTS) proporcionando paso a paso su configuración. Aclarar que los pasos siguientes se deben realizar en los nodos controller de cada región. De tal manera de la instalación y configuración siguiente aplica para la región uno y así mismo para la región dos. Los siguientes pasos se centran en la configuración de la región uno para ejemplificar su configuración e instalación.

Se debe crear la base de datos a utilizar para el servicio. Se debe ejecutar como usuario root.

```
# mysql
```

Crear base de datos llamada octavia.

```
# CREATE DATABASE octavia;
```

Darle los respectivos permisos a la base de datos octavia.

```
# GRANT ALL PRIVILEGES ON octavia.* TO 'octavia'@'localhost' \
IDENTIFIED BY 'icc1152024';
# GRANT ALL PRIVILEGES ON octavia.* TO 'octavia'@'%' \
IDENTIFIED BY 'icc1152024';
```

Salir de la base de datos.

```
# exit;
```

Cargar las credenciales de entorno para administradores.

```
$ . admin-openrc
```

Crear credenciales de servicio de octavia. Solicitará clave y repetirla, para la cual ingresar icc1152024.

```
$ openstack user create --domain default --password-prompt
Octavia
```

Agregar rol al usuario de octavia.

```
$ openstack role add --project service --user octavia admin
```

Crear entidades de servicio de octavia.

```
$ openstack service create --name octavia --description
```

```
"OpenStack Octavia" load-balancer
```

Crear los respectivos endpoints para la región.

```
$ openstack endpoint create --region RegionOne \
  load-balancer public http://controller:9876
```

```
$ openstack endpoint create --region RegionOne \
  load-balancer internal http://controller:9876
```

```
$ openstack endpoint create --region RegionOne \
  load-balancer admin http://controller:9876
```

Crear el archive octavia-openrc.

```
$ cat << EOF >> $HOME/octavia-openrc
export OS_PROJECT_DOMAIN_NAME=Default
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=service
export OS_USERNAME=octavia
export OS_PASSWORD=icc1152024
export OS_AUTH_URL=http://controller:5000
export OS_IDENTITY_API_VERSION=3
export OS_IMAGE_API_VERSION=2
export OS_VOLUME_API_VERSION=3
EOF
```

Creación de imagen de anphora. La imagen debe estar previamente descargada para este paso.

```
$ openstack image create --disk-format qcow2 --container-format
bare \
  --private --tag amphora \
  --file <path to the amphora image> amphora-x64-haproxy
```

Crear sabor para la imagen.

```
$ openstack flavor create --id 200 --vcpus 1 --ram 1024 \
  --disk 2 "amphora" --private
```

Instalación de componentes.

```
$ apt install octavia-api octavia-health-manager octavia-
housekeeping \
  octavia-worker python3-octavia python3-octaviaclient
```

Crear certificados a utilizar.

```
$ git clone https://opendev.org/openstack/octavia.git
$ cd octavia/bin/
$ source create_dual_intermediate_CA.sh
$ sudo mkdir -p /etc/octavia/certs/private
$ sudo chmod 755 /etc/octavia -R
$ sudo cp -p etc/octavia/certs/server_ca.cert.pem
/etc/octavia/certs
$ sudo cp -p etc/octavia/certs/server_ca-chain.cert.pem
/etc/octavia/certs
$ sudo cp -p etc/octavia/certs/server_ca.key.pem
/etc/octavia/certs/private
$ sudo cp -p etc/octavia/certs/client_ca.cert.pem
/etc/octavia/certs
$ sudo cp -p etc/octavia/certs/client.cert-and-key.pem
/etc/octavia/certs/private
```

Cargar variables de entorno de octavia.

```
$ . octavia-openrc
```

Creación de los grupos de seguridad y reglas.

```
$ openstack security group create lb-mgmt-sec-grp
$ openstack security group rule create --protocol icmp lb-mgmt-
sec-grp
$ openstack security group rule create --protocol tcp --dst-port
22 lb-mgmt-sec-grp
$ openstack security group rule create --protocol tcp --dst-port
9443 lb-mgmt-sec-grp
```

```
$ openstack security group create lb-health-mgr-sec-grp
$ openstack security group rule create --protocol udp --dst-port
5555 lb-health-mgr-sec-grp
```

Creación de llave keypair para uso del servicio de octavia.

```
$ openstack keypair create --public-key ~/.ssh/id_rsa.pub mykey
```

Crear el respectivo archivo dhclient.conf para el dhclient.

```
$ cd $HOME
$ sudo mkdir -m755 -p /etc/dhcp/octavia
$ sudo cp /etc/dhcp/dhclient.conf /etc/dhcp/octavia
```

Creación de red a utilizar. Cuando se ejecuten los siguientes comandos deberá guardar BRNAME y MGMT_PORT_MAC para su posterior utilización.

```
$ OCTAVIA_MGMT_SUBNET=172.16.0.0/12
$ OCTAVIA_MGMT_SUBNET_START=172.16.0.100
$ OCTAVIA_MGMT_SUBNET_END=172.16.31.254
$ OCTAVIA_MGMT_PORT_IP=172.16.0.2

$ openstack network create lb-mgmt-net
$ openstack subnet create --subnet-range $OCTAVIA_MGMT_SUBNET --
allocation-pool \
  start=$OCTAVIA_MGMT_SUBNET_START,end=$OCTAVIA_MGMT_SUBNET_END
\
  --network lb-mgmt-net lb-mgmt-subnet

$ SUBNET_ID=$(openstack subnet show lb-mgmt-subnet -f value -c
id)
$ PORT_FIXED_IP="--fixed-ip subnet=$SUBNET_ID,ip-
address=$OCTAVIA_MGMT_PORT_IP"

$ MGMT_PORT_ID=$(openstack port create --security-group \
  lb-health-mgr-sec-grp --device-owner Octavia:health-mgr \
  --host=$(hostname) -c id -f value --network lb-mgmt-net \
  $PORT_FIXED_IP octavia-health-manager-listen-port)

$ MGMT_PORT_MAC=$(openstack port show -c mac_address -f value \
  $MGMT_PORT_ID)
```

```
$ sudo ip link add o-hm0 type veth peer name o-bhm0
$ NETID=$(openstack network show lb-mgmt-net -c id -f value)
$ BRNAME=brq$(echo $NETID|cut -c 1-11)
$ sudo brctl addif $BRNAME o-bhm0
$ sudo ip link set o-bhm0 up

$ sudo ip link set dev o-hm0 address $MGMT_PORT_MAC
$ sudo iptables -I INPUT -i o-hm0 -p udp --dport 5555 -j ACCEPT
$ sudo dhclient -v o-hm0 -cf /etc/dhcp/octavia
```

Configuración de archivos. Se deberá editar el archivo **/etc/systemd/network/o-hm0.network.**

```
[Match]
Name=o-hm0

[Network]
DHCP=yes
```

Edición del archivo **/etc/systemd/system/octavia-interface.service.**

```
[Unit]
Description=Octavia Interface Creator
Requires=neutron-linuxbridge-agent.service
After=neutron-linuxbridge-agent.service

[Service]
Type=oneshot
RemainAfterExit=true
ExecStart=/opt/octavia-interface.sh start
ExecStop=/opt/octavia-interface.sh stop

[Install]
WantedBy=multi-user.target
```

Además se deberá editar el archivo **/opt/octavia-interface.sh.**

```
#!/bin/bash

set -ex
```

```
MAC=$MGMT_PORT_MAC
BRNAME=$BRNAME

if [ "$1" == "start" ]; then
  ip link add o-hm0 type veth peer name o-bhm0
  brctl addif $BRNAME o-bhm0
  ip link set o-bhm0 up
  ip link set dev o-hm0 address $MAC
  ip link set o-hm0 up
  iptables -I INPUT -i o-hm0 -p udp --dport 5555 -j ACCEPT
elif [ "$1" == "stop" ]; then
  ip link del o-hm0
else
  brctl show $BRNAME
  ip a s dev o-hm0
fi
```

Edición del archivo **/etc/octavia/octavia.conf.** Configurando el acceso a la base de datos Octavia.

```
[database]
connection =
mysql+pymysql://octavia:icc1152024@controller/octavia
```

Configurar la URL de transporte RabbitMQ.

```
[DEFAULT]
transport_url = rabbit://openstack:icc1152024@controller
```

Configurar dirección URL de transporte y el nombre del tema en **[oslo_messaging].**

```
[oslo_messaging]
topic = octavia_prov
```

Configurar el Puerto a utilizar y la IP del host.

```
[api_settings]
bind_host = 0.0.0.0
bind_port = 9876
```

Configuración del acceso a keystone.

```
[keystone_authtoken]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = octavia
password = icc1152024
```

Configurar las credenciales en la sección respectiva para utilizar los servicios de OpenStack.

```
[service_auth]
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = octavia
password = icc1152024
```

Se deberá configurar ruta absoluta del certificado CA, su clave privada y contraseñas.

```
[certificates]
server_certs_key_passphrase = insecure-key-do-not-use-this-key
ca_private_key_passphrase = not-secure-passphrase
ca_private_key = /etc/octavia/certs/private/server_ca.key.pem
ca_certificate = /etc/octavia/certs/server_ca.cert.pem
```

Configurar certificado del cliente de la siguiente manera.

```
[haproxy_amphora]
server_ca = /etc/octavia/certs/server_ca-chain.cert.pem
client_cert = /etc/octavia/certs/private/client.cert-and-key.pem
```

Configuración de la IP de la sección **[health_manager].**

```
[health_manager]
bind_port = 5555
bind_ip = 172.16.0.2
controller_ip_port_list = 172.16.0.2:5555
```

Configurar el worker de la siguiente manera.

```
[controller_worker]
amp_image_owner_id = <id of service project>
amp_image_tag = amphora
amp_ssh_key_name = mykey
amp_secgroup_list = <lb-mgmt-sec-grp_id>
amp_boot_network_list = <lb-mgmt-net_id>
amp_flavor_id = 200
network_driver = allowed_address_pairs_driver
compute_driver = compute_nova_driver
amphora_driver = amphora_haproxy_rest_driver
client_ca = /etc/octavia/certs/client_ca.cert.pem
```

Se procede a poblar la base de datos de octavia.

```
# octavia-db-manage --config-file /etc/octavia/octavia.conf
upgrade head
```

Se deben restablecer los servicios involucrados.

```
# systemctl restart octavia-api octavia-health-manager octavia-
housekeeping octavia-worker
```

OpenLDAP

Para instalar OpenLDAP, comenzamos actualizando el sistema con el siguiente comando.

```
$ sudo hostnamectl set-hostname ldap.identity.net
```

Instalar sladp y ldap-utils.

```
$ sudo apt install slapd ldap-utils
```

Durante la instalación, se te solicitará que establezca una contraseña para el administrador del directorio LDAP.

A continuación, editamos el archivo de configuración de hosts.

```
$ sudo nano /etc/hosts
```

En este archivo, añadimos la siguiente línea:

```
10.0.0.11 ldap.identity.net ldap
```

Finalmente, ejecutamos la reconfiguración de sladp para ajustar las opciones necesarias:

```
$ sudo dpkg-reconfigure slapd
```

Figura 42: Configuracion de slapd para ldap

Cuando pregunte si deseas omitir la configuración del servidor OpenLDAP, selecciona “NO”.

Dejamos como esta y seleccionamos OK

Figura 43: Configuracion dns de slapd para ldap

Cuando pida ingresar la contraseña se debe de escribir la misma que se configuró anteriormente.

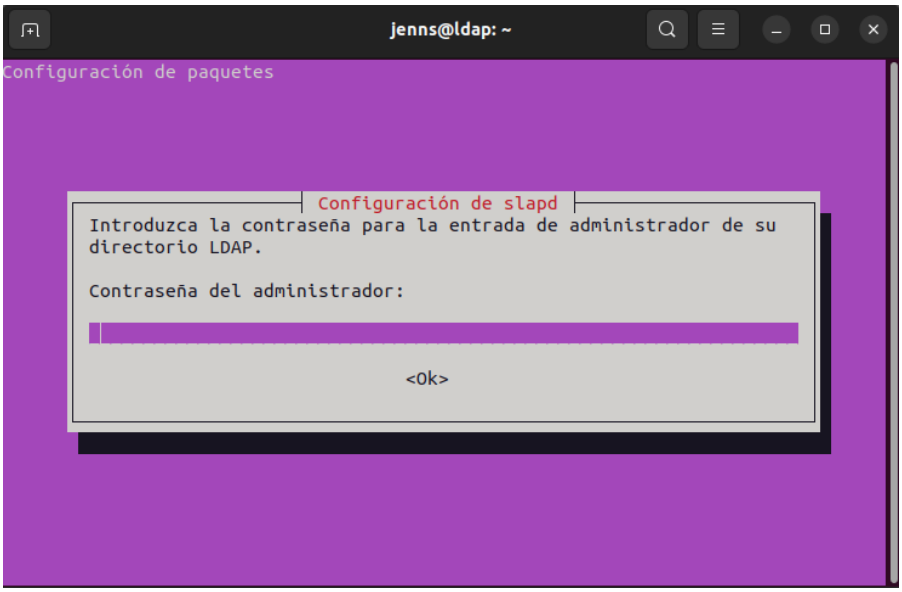


Figura 45: Configuracion de contraseña slapd para ldap

En la pantalla donde pregunta si desea borrar la base de datos es de seleccionar NO.

Figura 44: Configuracion de base de datos de slapd para ldap

Después de los pasos anteriores, nos preguntará si deseamos mover la base de datos, en te caso seleccionamos que si

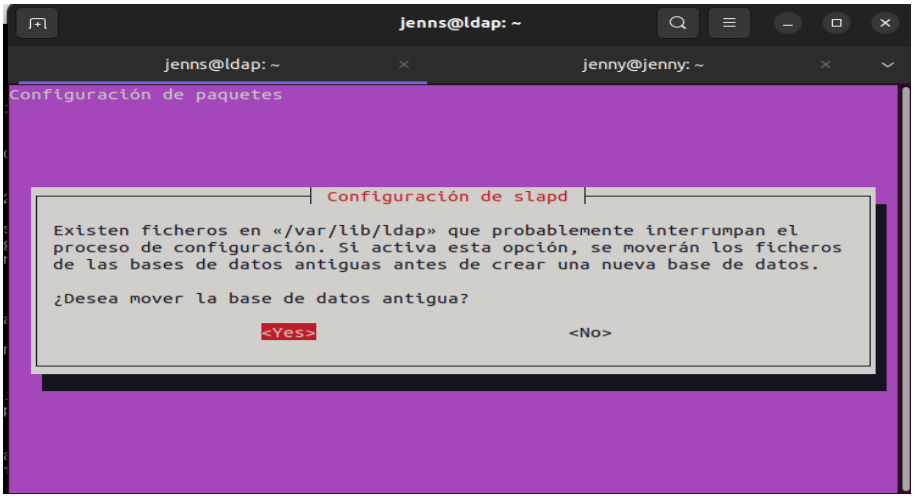


Figura 46: Configuracion de respaldo de base de datos en slapd para ldap

Ejecutar el siguiente comando para reiniciar el servicio de OpenLDAP y aplicar cambios

```
$ sudo systemctl restart slapd
```

Activar el siguiente servicio ufw

```
$ sudo ufw enable
```

Permitir el tráfico en los puertos requeridos por LDAP y otros servicios con los siguientes comandos.

```
$ sudo ufw allow ldap
$ sudo ufw allow ldaps
$ sudo ufw allow ssh
$ sudo ufw reload
$ sudo ufw status
```

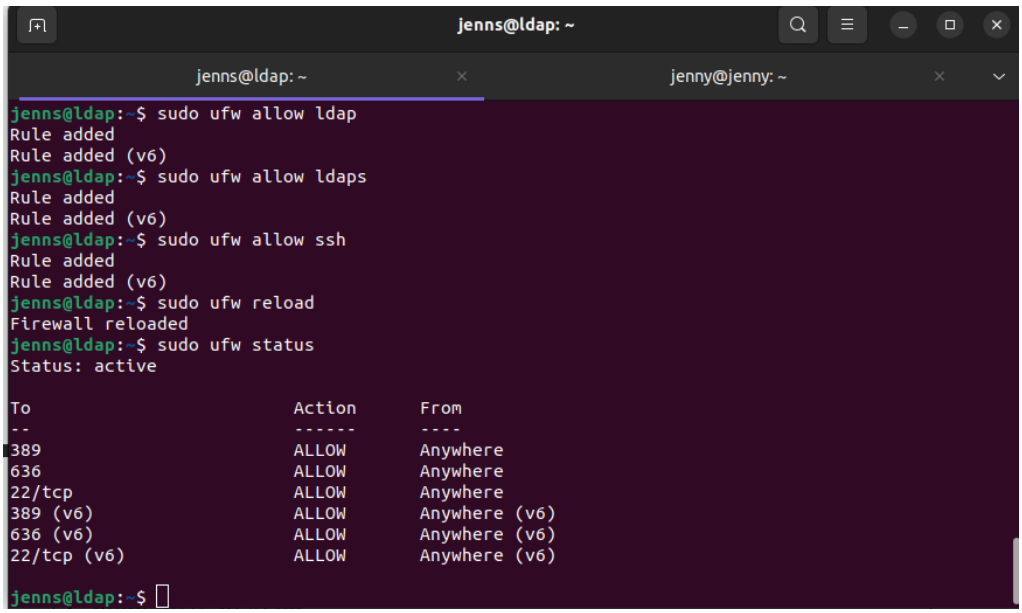


Figura 47: Configuracion de puertos en slapd para ldap

Creación de Árbol de direcciones

Primero, creamos un archivo llamado tree.ldif y agregamos el siguiente contenido.

```
dn: cn=accounts,dc=identity,dc=net
objectClass: organizationalRole
cn: accounts

dn: cn=users,cn=accounts,dc=identity,dc=net
objectClass: organizationalRole
```

```
cn: users

dn: cn=groups,cn=accounts,dc=identity,dc=net
objectClass: organizationalRole
cn: groups
```

Luego, ejecutamos este comando para agregar la estructura al servicio LDAP.

```
# ldapadd -x -D "cn=admin,dc=identity,dc=net" -W -f tree.ldif
```

Nota: al ejecutar el comando, se te pedirá que ingreses la contraseña que estableciste en este caso sería icc1152024.

Creación de Usuario Lookup

Crea un archivo llamado usuario.ldif y añade el siguiente contenido:

Se debe de generar una nota encriptada y eso se hace con el siguiente comando.

```
slappasswd -s PASSWORD
```

La contraseña es icc1152024. El contenido del archivo usuario.ldif seria.

```
dn: uid=svc-ldap,cn=users,cn=accounts,dc=identity,dc=net
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
uid: svc-ldap
sn: svc-ldap
cn: Service LDAP
userPassword: {SSHA}vYeyCBOSo2W5XwLU2LGAin/xF5WppAU8

dn: cn=grp-openstack,cn=groups,cn=accounts,dc=identity,dc=net
objectClass: top
objectClass: groupOfNames
cn: grp-openstack
member: uid=svc-ldap,cn=users,cn=accounts,dc=identity,dc=net
```

Después de crear el archivo, ejecutar el siguiente comando para agregar el usuario al servicio LDAP.

```
# ldapadd -x -D "cn=admin,dc=identity,dc=net" -W -f
usuario.ldif
```

Creación de Usuarios Normales

Crear otro archivo llamado usuarios.ldif y añade los siguientes dos usuarios.

```
dn: uid=johndoe,cn=users,cn=accounts,dc=identity,dc=net
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
uid: johndoe
cn: John Doe
sn: Doe
givenName: John
mail: johndoe@example.com
userPassword: {SSHA}vYeyCBOSo2W5XwLU2LGAin/xF5WppAU8

dn: uid=janedoe,cn=users,cn=accounts,dc=identity,dc=net
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
uid: janedoe
cn: Jane Doe
sn: Doe
givenName: Jane
mail: janedoe@example.com
userPassword: {SSHA}vYeyCBOSo2W5XwLU2LGAin/xF5WppAU8
```

Una vez que se ha creado este archivo, ejecutar el siguiente comando para agregar los usuarios al servidor LDAP.

```
# ldapadd -x -D "cn=admin,dc=identity,dc=net" -W -f
usuario.ldif
```

Configuración del Certificado

Al final del archivo de configuración de LDAP, se encuentra la ruta del certificado. En el caso de Ubuntu 22.04, la ruta predeterminada es /etc/ssl/certs/ca-certificates.crt.

Ahora, copiemos este archivo al servidor donde sen encuentra Keystone. Para hacer esto, se hará uso del comando scp para transferir el archivo:

```
# scp /etc/ssl/certs/ca-certificates.crt
USUARIO@IP_KEYSTONE_SERVER:/home/USUARIO
```

Para habilitar la conexión ssh, es necesario modificar la configuración. Abre el archivo /etc/ssh/sshd_config y localiza la siguiente línea.

```
PermitRootLogin without-password
#Cambiarla por
PermitRootLogin yes
```

Después de hacer este cambio, reiniciar el servicio ssh con el siguiente comando.

```
# service ssh restart
```

Una vez que el archivo esté en la carpeta del usuario en el servicio de Keystone, debes convertirlo en formato PEM. Para ello, usa el siguiente comando.

```
# openssl x509 -in ca-certificates.crt -out cacertificates.pem
-outform PEM
```

Luego, mueve ambos archivos al directorio adecuado en el servidor, que normalmente es el directorio de llaves:

```
# sudo cp ca-certificates.pem /usr/local/share/cacertificates/
# sudo cp ca-certificates.crt /usr/local/share/cacertificates/
```

Finalmente, actualizar los certificados del sistema con este comando.

```
# sudo update-ca-certificates
```

Configuración Nuevo Dominio

En el nodo donde se encuentra Keystone, creamos un nuevo directorio para alojar las configuraciones del nuevo dominio con los siguientes comandos.

```
$ sudo mkdir /etc/keystone/domains/
$ sudo chown keystone /etc/keystone/domains/
```

Luego, añadir las configuraciones necesarias al archivo de configuración de Keystone ubicado en /etc/keystone/keystone.conf. Las siguientes líneas deben agregarse.

```
[assignment]
driver = sql

[identity]
domain_specific_drivers_enabled = true
domain_config_dir = /etc/keystone/domains
```

Para crear el nuevo dominio usando la CLI de OpenStack, ejecutamos el siguiente comando (asegurándonos de haber iniciado sesión correctamente).

```
# openstack domain create LAB
```

A continuación, creamos un archivo de configuración específico para el nuevo dominio. Este archivo será usado para establecer la conexión con LDAP y debe ubicarse en /etc/keystone/domains/keystone.LAB.conf. Dentro de este archivo, añadimos la siguiente configuración.

```
[ldap]
url = ldap://10.0.0.11
user = uid=svc-ldap,cn=users,cn=accounts,dc=identity,dc=net
password = icc115
user_tree_dn = cn=users,cn=accounts,dc=identity,dc=net
user_objectclass = inetOrgPerson
user_id_attribute = uid
user_name_attribute = uid
user_mail_attribute = mail
```

```
user_pass_attribute =
user_allow_create = False
user_allow_update = False
user_allow_delete = False
tls_cacertfile = /usr/local/share/ca-
certificates/cacertificates.crt

[identity]
driver = ldap
```

Cambiamos los permisos del archivo de configuración recién creado para que el usuario keystone sea el propietario.

```
$ sudo chown keystone /etc/keystone/domains/keystone.LAB.conf
```

Ahora, debemos otorgar acceso al usuario admin para el nuevo dominio LAB. Para ello, necesitamos obtener ciertos datos.

El ID del dominio LAB.

```
# openstack domain show LAB
```

El ID del usuario admin.

```
# openstack user list --domain default | grep admin
```

El ID del rol admin.

```
# openstack role list
```

Con los IDS obtenidos, asignamos el rol admin al usuario admin sobre el dominio LAB usando el siguiente comando.

```
# openstack role add --domain ID_DOMAIN --user ID_USER_ADMIN
ID_ROLE_ADMIN
```

Reiniciamos el servicio Apache para que las nuevas configuraciones entren en vigor.

```
# service apache2 restart
```

Podemos listar los usuarios que se encuentra en LDAP para el dominio LAB usando el siguiente comando.

```
# openstack user list --domain LAB
```

Nota: Si surge algún error al ejecutar los comandos, puede que necesitemos instalar Python y actualizar los paquetes.

```
# apt install python3-pip
# apt update
```

Asignación de permisos a los usuarios de LDAP para que puedan acceder al panel de OpenStack.

Primero, obtenemos el ID de uno de los usuarios listado en el dominio LAB.

```
# openstack user list --domain LAB
```

Una vez que tenemos el ID del usuario, se le debe de asignar un rol específico.

```
# openstack role add --user ID_JANE --domain default member
```

El comando final podría verse así.

```
# openstack role add --user
43bbf55baa47d2f99d07f332680bcb758e3c18378f65d85ef7f99cef8ac04
82c --domain default member
```

Si queremos otorgarle permisos sobre un proyecto específico dentro del dominio, como por ejemplo el proyecto admin, usamos este comando para asignar el rol member.

```
# openstack role add --user
43bbf55baa47d2f99d07f332680bcb758e3c18378f65d85ef7f99cef8ac04
82c --project admin member
```

Finalmente, el usuario ahora podrá iniciar sesión en el panel de Horizon utilizando sus credenciales, contraseña y seleccionando el dominio LAB.

Yuyu

Instalación de la API de Yuyu

Clonar el código fuente más reciente, dirígete al directorio donde deseas colocar el código, en este ejemplo será /var.

```
cd /var
git clone https://github.com/Yuyu-billing/yuyu.git
cd yuyu
```

Crear y activar un entorno virtual, configurar el entorno virtual para que use Python 3.8.

```
virtualenv env --python=python3.8
source env/bin/activate
pip install -r requirements.txt
```

Instalar las dependencias, ejecutar la instalación de las dependencias necesarias usando pip.

```
pip install -r requirements.txt
```

Configurar el archivo local de ajustes, copia el archivo de configuración de ejemplo y edítalo.

```
cp yuyu/local_settings.py.sample yuyu/local_settings.py
vi yuyu/local_settings.py
```

Añadir la siguiente configuración.

```
YUYU_NOTIFICATION_URL =
"rabbit://openstack:password@127.0.0.1:5672//"
YUYU_NOTIFICATION_TOPICS = ["notifications"]
```

Ejecutar la migración de la base de datos, se debe de aplicar las migraciones necesarias en la base de datos.

```
python manage.py migrate
```

Instalar y activa la API de Yuyu, para eso se debe de ejecutar el siguiente script.

```
./bin/setup_api.sh
systemctl enable yuyu_api
systemctl start yuyu_api
systemctl status yuyu_api
```

Instalar y activar el monitor de eventos de Yuyu, configurar el monitor de eventos usando el siguiente script.

```
./bin/setup_event_monitor.sh
systemctl enable yuyu_event_monitor
systemctl start yuyu_event_monitor
systemctl status yuyu_event_monitor
```

Instalación de crontab.

```
crontab -e
```

Añadir esta línea para que se procese la facturación el día 1 de cada mes a la medianoche.

```
1 0 1 * * /var/yuyu/bin/process_invoice.sh
```

Para finalizar, desactiva el entorno virtual.

```
Deactivate
```

Panel de control de Yuyu

Clonar el repositorio del panel de control, ve al directorio /var y clona el respositorio del panel de Yuyu.

```
cd /var
git clone https://github.com/Yuyu-billing/yuyu_dashboard.git
cd yuyu_dashboard
```

Configurar el panel de control, para ello se debe de ejecutar el siguiente script.

```
./setup_yuyu.sh
```

Durante el proceso, pedirá que se ingrese la ruta de la instalación de Horizon. Por ejemplo:

```
/var/www/html/horizon
```

Instalar las dependencias de Yuyu, usa pip para instalar las dependencias.

```
pip3 install -r requirements.txt
```

Agregar configuración de horizon local_settings..py.

```
Vi /var/www/html/horizon/openstack_dashboard/local/local_settings.py
```

Agregar las siguientes líneas.

```
YUYU_URL="http://yuyu_server_url:8182"
CURRENCIES = ('IDR',)
DEFAULT_CURRENCY = "IDR"
```

Después de hacer los cambios, reiniciar el servicio de Apache.

```
systemctl restart apache2
```

Si notas que la vista de inicio de sesión del panel no se carga correctamente, reinicia el servicio de mencached.

```
systemctl restart memcached
```

Configuración del compute

Instalar Chrony.

Se debe cambiar el hostname de la máquina virtual donde se instalarán los servicios.

```
# hostnamectl set-hostname controller

# exec bash

# apt update -y

# apt upgrade –y
```

Se deben agregar las direcciones IP de los nodos en el archivo etc/hosts.

```
# nano /etc/hosts
```

Luego agregar las siguientes líneas debajo de la IP 127.0.0.1.

```
10.0.0.11 controller

10.0.0.31 compute1

10.0.0.32 compute2

10.0.0.41 storage
```

Se debe configurar el archivo netplan para poder cambiar las direcciones IP de las máquinas. Al guardar el archivo ejecutar sudo netplan apply para guardar los cambios.

Ejecutar el siguiente comando.

```
# nano /etc/netplan/00-installer-config.yaml
```

Agregar la siguiente configuración.

```
network:
  ethernets:
    enp1s0:
      dhcp4: false
      addresses: [192.168.122.31/24]
      gateway4: 192.168.122.1
      nameservers:
        addresses: [8.8.8.8,8.8.4.4]
    enp7s0:
      dhcp4: false
      addresses: [10.0.0.31/24]
    enp8s0:
      dhcp4: true
    ztbtoxhtbp:
      addresses: [192.168.193.44/24]
      nameservers:
        addresses: [8.8.8.8,8.8.4.4]
  version: 2
```

Se debe instalar el servicio de chrony para sincronizar el reloj de los nodos con el nodo compute.

```
# apt install chrony –y
```

Se debe configurar el archivo de chrony localizado en etc/chrony/chrony.conf.

```
# nano etc/chrony/chrony.conf
```

Agregar la siguiente línea en el archivo de configuración.

```
server controller iburst
```

Luego de guardar el archivo ejecutar.

```
# service chrony restart
```

Instalar la version de Openstack y openstackclient. Ejecutar los siguientes comandos.

```
# add-apt-repository cloud-archive:bobcat

# apt install python3-openstackclient
```

Instalar nova en los compute

```
# apt install nova-compute
```

Configurar el archivo localizado en /etc/nova/nova.conf

```
# nano /etc/nova/nova.conf
```

Reemplazar las secciones en corchetes por los valores siguientes.

```
[DEFAULT]

transport_url = rabbit://openstack:icc1152024@controller
my_ip = 10.0.0.11
```

```
[api]
auth_strategy = keystone

[keystone_authtoken]
www_authenticate_uri = http://controller:5000/
auth_url = http://controller:5000/
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
region_name = RegionOne
username = nova
password = icc1152024

[service_user]
send_service_user_token = true
auth_url = https://controller:5000/identity
auth_strategy = keystone
auth_type = password
project_domain_name = Default
project_name = service
region_name = RegionOne
user_domain_name = Default
username = nova
password = icc1152024

[vnc]
enabled = true
server_listen = 0.0.0.0
server_proxyclient_address = $my_ip
novncproxy_base_url = http://controller:6080/vnc_auto.html

[glance]
api_servers = http://controller:9292

[oslo_concurrency]
lock_path = /var/lib/nova/tmp

[placement]
region_name = RegionOne
project_domain_name = Default
project_name = service
auth_type = password
user_domain_name = Default
auth_url = http://controller:5000/v3
username = placement
```

```
password = icc1152024
```

En el controller ejecutar los siguientes comandos.

Cargar las variables de entorno y listar los computes disponibles

```
$ . admin-openrc
openstack compute service list --service nova-compute
```

Descubrir los compute.

```
# su -s /bin/sh -c "nova-manage cell_v2 discover_hosts --
verbose" nova
```

Para que automáticamente se carguen los compute en el archivo /etc/nova/nova.conf:

```
# nano /etc/nova/nova.conf:
```

Buscar y reemplazar.

```
[scheduler]
discover_hosts_in_cells_interval = 300
```

Instalar neutron en los nodos compute

Instalar openvswitch en el compute.

```
# apt install neutron-openvswitch-agent
```

Ejecutar el archivo localizado en /etc/neutron/neutron.conf.

```
# nano /etc/neutron/neutron.conf
```

```
[DEFAULT]
transport_url = rabbit://openstack:icc1152024@controller
```

```
auth_strategy = keystone

[keystone_authtoken]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = neutron
password = icc1152024

[oslo_concurrency]
lock_path = /var/lib/neutron/tmp
```

Configurar self service provider. Configurar el archivo localizado en /etc/neutron/plugins/ml2/openvswitch_agent.ini.

```
# nano /etc/neutron/plugins/ml2/openvswitch_agent.ini
```

Buscar y reemplazar las secciones en corchetes sino existen agregarlas.

```
[ovs]
bridge_mappings = provider:br-provider

[vxlan]
local_ip = 10.0.0.11
l2_population = true

[securitygroup]
enable_security_group = true
firewall_driver = openvswitch

[agent]
tunnel_types = vxlan
l2_population = true
```

Ejecutar los siguientes comandos.

```
# ovs-vsctl add-br br-provider
```

```
# ovs-vsctl add-port br-provider enp7s0 (en netplan dhcp4 está
en true)
```

Configurar el archivo de nova localizado en /etc/nova/nova.conf.

```
# nano /etc/nova/nova.conf
```

Reemplazar la sección en corchetes.

```
[neutron]
auth_url = http://controller:5000
auth_type = password
project_domain_name = Default
user_domain_name = Default
region_name = RegionOne
project_name = service
username = neutron
password = icc1152024
```

Ejecutar los siguientes comandos.

```
# service nova-compute restart
# service neutron-openvswitch-agent restart
```