UNIVERSIDAD NACIONAL DEL CAAGUAZÚ
FACULTAD DE CIENCIAS Y TECNOLOGÍAS

Desarrollo de un Clúster de alta disponibilidad con balanceo de carga en la nube para aplicaciones de la Facultad de Ciencias y Tecnologías UNCA en el año 2019

ELABORADO POR

Fernando Miguel Rojas Mosqueira - Juan Francisco Silvero Báez

TUTOR

Ing. Julio César Coronel Ramoa

Trabajo presentado a la Facultad de Ciencias y Tecnologías de la Universidad Nacional de Caaguazú, como requisito para la obtención del título de Ingeniero en Sistemas Informáticos.

CORONEL OVIEDO – PARAGUAY

Página de aprobación

Mesa examinadora de sustentación de tesis de grado

Carrera de Ingeniería en Informática

Título de la Tesis: Desarrollo de un Clúster de alta disponibilidad con balanceo de carga en la nube para aplicaciones de la Facultad de Ciencias y Tecnologías UNCA en el año 2019

Calificación obtenida: _____ (_____)

_____ Miembro

_____ Miembro

_____ Miembro

_____ Miembro

_____ Presidente

Acta N°: _____

Fecha: _____

Dedicatoria

A nuestras familias, amigos y profesores que siempre nos apoyaron en el transcurso de la carrera. Sin ustedes este trabajo no sería posible.
Gracias

Agradecimientos

Mi más cordial reconocimiento y agradecimiento a todos y cada uno de los Profesores de la Facultad, porque de alguna manera supieron brindarnos su gama de experiencia profesional.

Nuestro agradecimiento al tutor de tesis, por su colaboración y orientación en la realización del presente trabajo, ya que supo guiarnos de la mejor manera con su repertorio amplio de conocimientos.

A los docentes, funcionarios y directivos de la Facultad de Ciencias y Tecnologías (FCyT), por brindarnos siempre su apoyo.

Resumen

En el presente trabajo se describe la configuración necesaria para desarrollar un clúster de alta disponibilidad con balanceo de carga. El clúster contará con un servidor que actuará como gateway y firewall, con servidores nodos que se encargarán de todo el trabajo y también con servidores directores que repartirán el trabajo entre los servidores nodos equitativamente.

Los servidores utilizados para el clúster serán virtualizados, esto conlleva a una reducción en la inversión requerida y facilita el trabajo de configuración. Este trabajo no contempla la elección de un virtualizador en específico en vista de que el objetivo principal es utilizar una plataforma en la nube ya sea de la SENATICS o cualquier empresa privada que ofrezca los servicios básicos de computación en la nube.

El resultado final es la creación de un clúster de alta disponibilidad funcional que servirá de prototipo para su posible implementación en reemplazo a la estructura actual disponible en la FCyT.

Palabras clave: Alta Disponibilidad, Clúster, Nube.

Abstract

In the present work, the configuration necessary to implement a high availability cluster with load balancing is described. The cluster will have a server that will act as a gateway and firewall, with node servers that will be responsible for all the work and also with directing servers that will distribute the work among the node servers equally.

The servers used for the cluster will be virtualized, this leads to a reduction in the required investment and facilitates the configuration work. This work does not contemplate the choice of a specific virtualiser in view of the fact that the main objective is to use a platform in the cloud either from SENATICS or any private company that offers the basic services of computing in the cloud.

The final result is the creation of a functional high availability cluster that will serve as a prototype for its possible implementation in replacement of the current structure available in the FCyT.

Keywords: High Availability, Cluster, Cloud.

Índice

Índice de tablas

Índice de figuras

Introducción

La Facultad de Ciencias y Tecnologías UNCA hace uso de servidores privados virtuales (VPS, por sus siglas en inglés) para desplegar servicios de páginas web de la propia casa de estudios como también otras Facultades de la Universidad, además se utilizan para base de datos, sistemas informáticos académicos y sistemas informáticos administrativos. El esquema básico que para implementar los servidores es a través de un Firewall con IP estática y pública que redirecciona las peticiones que llegan desde el exterior hacia los servidores VPS con IP estática privada. De igual manera pero en sentido contrario, los servidores VPS con IP privada reciben las peticiones y devuelven las respuestas al firewall. Haciendo que todas las peticiones de las terminales de los clientes pasen por un Firewall.

Este esquema permite filtrar paquetes potencialmente peligrosos y soluciona la existencia de una sola IP estática pública. Sin embargo se tiene el inconveniente de "1 VPS, 1 servicio" y en caso de que ese servidor VPS por algún motivo deje de funcionar, todo el servicio cae y se vuelve inaccesible al cliente. El presente trabajo pretende solucionar ese problema colocando varios servidores VPS proporcionando el mismo servicio, otorgando redundancia y tolerancia a fallos.

Para realizar este trabajo se realizaron entrevistas a los encargados del Departamento de Informática de la Facultad, además nos facilitaron un gráfico del esquema utilizado para utilización de los servidores VPS. Pudimos corroborar los datos obtenidos de la entrevista haciendo instalaciones de servicios para la Facultad como parte de nuestra Pasantía,

constatamos que cada servicio debió adecuarse al esquema establecido para su correcto funcionamiento.

Verificamos los recursos que puede proveer la Facultad y consideramos que este proyecto es viable y si se llegara a implementar solucionará problemas de escalabilidad de servicios y permitirá la tolerancia a fallos.

El trabajo está divido en capítulos:

Capítulo I, se describe el planteamiento del problema de investigación, los objetivos Generales y específicos, la justificación y el alcance de proyecto.

Capítulo II, se expone el Marco Teórico con la descripción de los antecedentes de la investigación, las bases teóricas que sustentan teóricamente el Proyecto de Fin de Grado, también se presenta el Marco Conceptual, Operacional y el Marco Legal.

Capítulo III, se presenta el Marco Metodológico en él que se describen el tipo y diseño de la investigación, la población y muestra de estudio, además las técnicas e instrumentos utilizados para la recolección de datos y las fases metodológicas del Proyecto de fin de Grado.

Capítulo IV, presenta el Marco Analítico, que comprende los estudios de factibilidad técnica, económica y operacional, también los requerimientos del sistema y el análisis y diseño del mismo.

Capítulo V, en este capítulo se describe la instalación del clúster.

Capítulo VI, en este capítulo se muestran las Conclusiones y Recomendaciones, Además de las referencias bibliográficas, los anexos y el apéndice

CAPÍTULO I

1. Planteamiento del problema

Para cualquier empresa o institución, una interrupción en el funcionamiento de sus sistemas informáticos o aplicaciones web supone un serio problema. Esto puede darse debido a la alta exigencia que sufren sus servidores ocasionando la caída de sistemas que pueden darse por fallas causadas por varios factores, ocasionando desavenencias y retardos de trabajo tanto en los operadores como en los usuarios beneficiados, factores que para algunos sectores pueden significar serias pérdidas económicas.

En cuanto a las entidades educativas dentro del país se tiene el caso de las Universidades las cuales emplean los sistemas informáticos en el desarrollo de su trabajo cotidiano. En este caso al tomar en cuenta a la Universidad Nacional del Caaguazú y más específicamente la Facultad de Ciencias y Tecnologías, se puede destacar la utilidad indispensable de los servicios en la nube. Podemos citar la la página web informativa de la Facultad, el portal de consulta de notas, el repositorio de tesis y trabajos de investigación, bases de datos de sistemas informáticos, etc. Cada servicio depende exclusivamente de su respectivo servidor privado virtual (VPS) proveído gratuitamente por el Ministerio de Tecnologías de la Información y Comunicación. Si el servidor VPS falla, el servicio cae. Es por eso que se necesita de un arreglo o clúster de servidores para evitar que los servicios caigan por la caída de un servidor VPS. Precisamente se necesita de un clúster de Alta

Disponibilidad, que como su nombre lo dice, provee de alta disponibilidad a los servicios proveídos por la Facultad.

2. Pregunta de investigación

¿De qué manera un clúster de alta disponibilidad y balanceo de carga beneficiaría a las servicios en la nube de la Facultad de Ciencias y Tecnologías UNCA en el año 2019?

3. Delimitación y alcance

El proyecto consiste en el diseño de una estructura de Clúster de Alta disponibilidad con balanceo de carga para los servicios en la nube proveídos por la Facultad de Ciencias y Tecnologías UNCA.

El trabajo abarca desde la recolección de datos de la estructura actual de los servidores VPS, hasta la elaboración de una nueva estructura de servidores o mejor dicho, un clúster de servidores. Se pondrá en marcha el clúster y se identificará los beneficios o perjuicios que puede tener esta nueva estructura para su posible implementación en la Facultad de Ciencias y Tecnologías UNCA.

4. Objetivos

4.1. Objetivo General

Desarrollar un clúster de alta disponibilidad y balanceo de carga para los servicios en la nube de la facultad de Ciencias y Tecnologías UNCA en el año 2019.

4.2. Objetivos Específicos

Desarrollar la estructura base del clúster de Alta Disponibilidad con balanceo de carga.

Identificar las aplicaciones a ser instaladas para el Clúster.

Realizar monitoreo del Clúster.

5. Justificación

En la actualidad los sistemas informáticos en las instituciones públicas y privadas se han convertido en pieza fundamental. Es impensable administrar información sin ayuda de ordenadores.

La Facultad de Ciencias y Tecnologías UNCA posee una serie de servicios colocados en servidores privados virtuales (VPS). Estos servicios brindan base de datos, páginas web, archivos, etc. que son requeridos en el día a día para el correcto funcionamiento de las dependencias de la Facultad.

Este trabajo pretende desarrollar un clúster que mantenga los servicios siempre disponibles, seguros ante ataques de personas mal intencionadas, accesibles desde cualquier dispositivo conectado a la Internet (obviamente solo accesible a personas autorizadas).

También es importante mencionar que a través de este clúster, se sentarán las bases para nuevos servicios, fácilmente escalables que podrán ser administrados y monitoreados de una forma ordenada y segura.

CAPÍTULO II

MARCO TEÓRICO, CONCEPTUAL, OPERACIONAL Y LEGAL

1. Antecedentes

Una vez indagado en la Biblioteca así como en el repositorio digital de la Facultad de Ingeniería en Informática, se puede afirmar que no se ha encontrado un Proyecto de Trabajo de Graduación enfocado al desarrollo de un Clúster de alta disponibilidad para algún servicio de la Facultad. Pero hallamos trabajos con un tema similar al nuestro.

En Escuela Politécnica Nacional de Quito - Ecuador, se encontró el siguiente tema: "Configuración de un clúster de alta disponibilidad y balanceo de carga en linux para satisfacer gran demanda web y servicios de resolución de nombres", elaborado por el Sr. Andrés Gustavo Bustos Burbano, en el cual concluye lo siguiente: La alta disponibilidad en un ambiente de gran demanda Web y resolución de nombres tiene mucha importancia en diferentes aspectos, uno de ellos es la facilidad de realizar el mantenimiento de servidores. Una máquina de servidores se puede sacar de línea, apagarse o actualizarse o repararse sin comprometer los servicios que brinda el Clúster y sin afectar el desempeño del sistema en general.

También se encontró el trabajo de fin de Máster del Señor Abraham Jaramillo Garófalo, "Configuración, optimización y evaluación de un servidor de alta disponibilidad con equilibrado de carga" de la Universitat Politècnica de València, donde se concluye lo

siguiente: Se logró implementar un repartidor de carga de alta disponibilidad que ofrece servicios web. El repartidor se compone de 2 nodos repartidores y 6 nodos servidores. Los nodos repartidores decidieron implementarse directamente sobre el hardware para alcanzar elevadas prestaciones, mientras que los nodos servidores se implementaron como máquinas virtuales para aprovechar su flexibilidad en la configuración y despliegue.

En la Universitat Oberta de Catalunya, se realizó un trabajo de Fin de carrera, titulado "Cluster Alta Disponibilidad sobre Plataforma GNU/LINUX" del Señor Santiago Barrio González donde se presentaron soluciones sobre como poder montar aplicaciones y sistemas de archivos sobre un cluster de alta disponibilidad en plataforma GNU/LINUX.

Así mismo se pueden citar cientos de trabajos en idiomas diferentes al español, verificamos que hay bibliografía suficiente para llevar a cabo este proyecto.

2. Bases teóricas

2.1. Desarrollar la estructura base del clúster de Alta Disponibilidad y balanceo de carga.

2.1.1. Clúster

Es un conjunto de computadoras construidas mediante la utilización de componentes de hardware que se comportan como si fuesen una única computadora (Buyya, 1999).

La tecnología de clúster ha evolucionado gracias al apoyo de actividades que van desde aplicaciones de supercómputo, software de misiones críticas, servidores web y comercio electrónico, hasta bases de datos de alto rendimiento, entre otros usos. El cómputo con clúster surge como resultado de la convergencia de varias tendencias actuales. Incluye disponibilidad de microprocesadores económicos de alto rendimiento y redes de alta

velocidad, desarrollo de herramientas de software para cómputo distribuido de alto rendimiento y la creciente necesidad de potencia computacional para aplicaciones que la requieran.

2.1.2. ¿Qué es un nodo?

Un nodo es cada uno de los ordenadores que forma parte de un cluster. Se excluye de esta definición a los hubs, switchs, routers y cualquier otro dispositivo de interconexión que no participe en la ejecución de los procesos ni se puedan migrar procesos a él. (S. Cortéz y M. Galarza, 2005)

Para este caso en particular, los ordenadores son Servidores Privados Virtuales (VPS) y serán los encargados de recibir peticiones y devolver resultados.

2.1.3. Características de un clúster

(S. Cortéz y M. Galarza, 2005) En su parte central, la tecnología de Clusters consta de 3 partes:

1. Un cluster debe estar compuesto de 2 o más nodos
2. El segundo componente, hace referencia al sistema operativo
3. Y el tercer componente es la interconexión de hardware

Un clúster debe estar compuesto por dos o más nodos que ofrezcan las mismas aplicaciones. No siempre un clúster es homogéneo, es decir, no necesariamente deben tener nodos con el mismo hardware o el mismo Sistema Operativo, ya que con la abstracción los nodos pueden acoplarse al mismo clúster.

Si bien no es necesario que el Sistema Operativo de cada uno de los nodos sea el mismo, sí es necesario que dichos Sistemas Operativos puedan adaptarse a un clúster.

En la actualidad hay interfaces de conexión muy eficientes, pero normalmente una red ethernet es la más utilizada por su facilidad de implementación y por el costo inferior a otras propuestas.

2.1.4. Clasificación

Los clústeres pueden clasificarse con base en sus características. Hay clústeres de alto rendimiento o High Performance Clúster (HPC), clusters de alta disponibilidad o High Availability (HA) y clústeres de alta eficiencia o High Throughput (HT). (Buyya, 1999).

High performance: Son clústeres en los cuales se ejecutan tareas que requieren una gran capacidad computacional, cantidades enormes de memoria o ambas a la vez. Llevar a cabo estas tareas puede comprometer los recursos del clúster por largos periodos (Oñate y Ortega, 2010).

High availability: Son clústeres cuyo objetivo es proveer disponibilidad y confiabilidad. Estos clústeres tratan de brindar la máxima disponibilidad de los servicios que ofrecen. La confiabilidad se provee mediante un software que detecta fallos y permite recuperarse frente a ellos, mientras que en hardware se evita tener un único punto de fallos (Oñate y Ortega, 2010).

High throughput: Son clústeres cuyo objetivo de diseño es ejecutar la mayor cantidad de tareas en el menor tiempo posible; existe independencia de datos entre las tareas individuales. El retardo entre los nodos del clúster no es considerado un gran problema (Oñate y Ortega, 2010).

2.1.5. Escalabilidad

(A. Khare, Y. Huang, H. Doan y M. Sing, 2015) La escalabilidad se define a veces como "la facilidad con la que un sistema o componente puede modificarse para ajustarse al área problemática". Un sistema escalable tiene tres características simples:

1. El sistema puede acomodar un mayor uso.
2. El sistema puede acomodar un mayor conjunto de datos.
3. El sistema es mantenible y funciona con un rendimiento razonable.

Hay dos tipos de escalabilidad que deben tenerse en cuenta:

Escalabilidad Vertical: Ampliar las unidades para incrementar los recursos del hardware del sistema. Por ejemplo: reemplazar una unidad de 4 GB de RAM por una unidad de 8 GB.

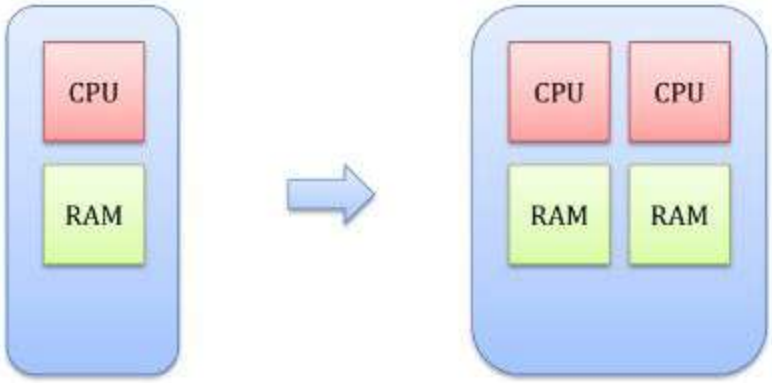


Figura 1: Escalabilidad vertical

Escalabilidad Horizontal: Agregar unidades para incrementar los recursos del hardware del sistema. Por ejemplo: Añadir una memoria de 4 GB de RAM al sistema.

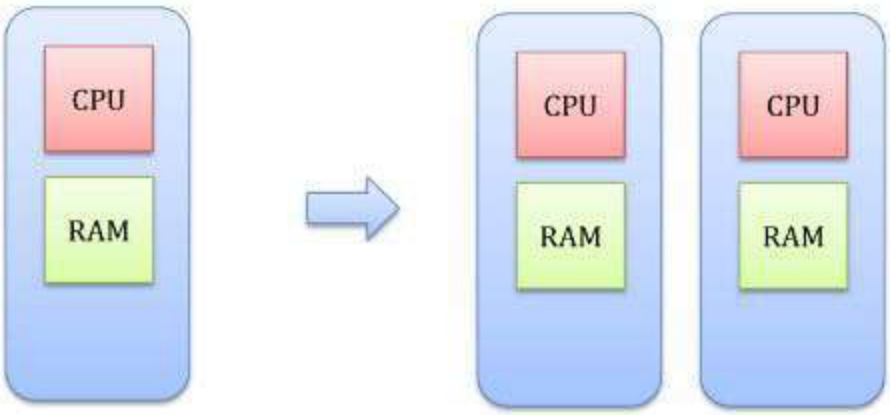


Figura 2: Escalabilidad Horizontal

2.1.6. Balanceo de carga

Según (T. Burke, 2001) es un proceso y tecnología que distribuye el tráfico entre varios servidores que utilizan un dispositivo basado en la red. Este dispositivo intercepta el tráfico destinado a un sitio y lo redirige a varios servidores. El proceso de equilibrio de carga es completamente transparente para el usuario final.

Cuando comenzamos a escalar horizontalmente, aparece un nuevo problema. Tenemos varios procesadores que residen en diferentes máquinas físicas, pero no tenemos un sistema de administración para distribuir las solicitudes entre ellos. Tenemos varias solicitudes que llegan a la misma IP, que queremos atender con varias máquinas.El problema es decidir qué máquina de aplicación o unidad de procesamiento respondería a qué solicitud.

La solución puede provenir de varios métodos, que podrían agruparse bajo la técnica de balanceo de carga. Existe un límite a lo que la escala vertical puede lograr para una aplicación en particular. Este límite puede ser determinado por el presupuesto de la compañía

para comprar hardware actualizado o la limitación técnica cuando la compañía ya está utilizando el mejor servidor disponible en el mercado. Estos sistemas necesitan escalar y equilibrar la carga. Por lo tanto, el equilibrio de carga ahora se ha convertido en una necesidad en casi cualquier arquitectura de servicio web, desempeñando un papel importante para garantizar la disponibilidad y escalabilidad de un sistema.

Un balanceador de carga acepta solicitudes de los usuarios y luego las dirige al servidor web correcto. El servidor "correcto" aquí se decide según ciertos criterios, principalmente considerados como la estrategia de equilibrio de carga. Hay muchas estrategias, las más comunes son:

- **Round Robin:** cada servidor toma turno para recibir solicitudes. Esta es la estrategia más simple, similar en espíritu a First In First Out aplicada en el almacenamiento en caché.
- **Menor número de conexiones:** el servidor con el menor número de conexiones será dirigido a la solicitud. Este es un intento de evitar una alta carga.
- **Tiempo de respuesta más rápido:** el servidor que tenga el tiempo de respuesta más rápido (ya sea recientemente o con frecuencia) será dirigido a la solicitud. Este es un intento de manejar solicitudes lo más rápido posible.
- **Ponderado:** esta estrategia puede ser altamente personalizada ya que la ponderación se puede configurar. Esta estrategia tiene en cuenta el escenario en el que los servidores del clúster pueden no tener la misma capacidad (capacidad de procesamiento, almacenamiento, etc.). Por lo tanto, los servidores más poderosos recibirán más solicitudes que los más débiles bajo la estrategia ponderada.

2.1.7. Balanceo de carga por Hardware

Según (T. Burke, 2001) los equilibradores de carga basados en conmutadores, también conocidos como equilibradores de carga basados en hardware, son dispositivos que se basan en chips de circuito integrado de aplicación específica (ASIC) para realizar las funciones de reescritura de paquetes.

La forma más directa de equilibrar las solicitudes entre varias máquinas en un grupo es usar un dispositivo de hardware. Lo enchufas, lo enciendes, configuras algunos ajustes y comienzas a servir el tráfico. El principio básico es que el tráfico de red se envía a una IP compartida en muchos casos llamada IP virtual (VIP) o IP de escucha. Esta VIP es una dirección que adjunta al equilibrador de carga. Una vez que el equilibrador de carga recibe una solicitud en este VIP, deberá tomar una decisión sobre dónde enviarlo, en función de su estrategia de equilibrio de carga.

2.1.8. Balanceo de carga por Software

Según (T. Burke, 2001) el balanceo de carga con software se realiza mediante un código de software que se ejecuta en la parte superior de la pila de red del sistema operativo del servidor.

El equilibrio de carga de software ofrece una alternativa económica a los equilibradores de carga de hardware. Hay varios softwares de equilibrio de carga disponibles en el mercado, como: Haproxy, Piranha (disponible sólo para RedHat), Nginx, etc.

Los equilibradores de carga de software generalmente se clasifican en dos categorías: capa 4 y capa 7, según la información de capa de red que utilizan para el equilibrio de carga.

Los equilibradores de carga de capa 4 utilizan la información proporcionada por TCP (protocolo de control de transmisión) en la capa de red. El equilibrador de carga captura la solicitud en esta capa y utiliza la información contenida en la secuencia TCP: la dirección IP y el puerto de origen y destino, que es suficiente para enrutar la solicitud. Dada esta información, podemos dirigir la conexión al puerto correcto en el backend. Dado que la conexión debe establecerse entre el cliente y el servidor en un transporte orientado a la conexión antes de enviar el contenido de la solicitud, el equilibrador de carga generalmente selecciona un servidor sin mirar el contenido de la solicitud.

Los equilibradores de carga de la capa 7, por otro lado, inspeccionan el mensaje correctamente hasta la capa de aplicación, examinando la solicitud HTTP en sí. Pueden ver la solicitud y sus encabezados y utilizarlos como parte de la estrategia de equilibrio. Por lo tanto, las solicitudes se pueden equilibrar en función de la información en la cadena de consulta, en las cookies o en cualquier encabezado que elijamos, así como la información normal de la capa 4, incluidas las direcciones de origen y destino. Por ejemplo, un elemento de uso frecuente para el equilibrio de la capa 7 es la propia URL de solicitud HTTP. Al equilibrar en función de la URL, se puede garantizar que todas las solicitudes de un recurso específico vayan a un servidor específico. Los equilibradores de carga de capa 7 pueden proporcionar requisitos de calidad de servicio para diferentes tipos de contenido y mejorar el rendimiento general del clúster.

2.1.9. Servidor Balanceador de Carga

Según (Kent Roberts, 2018) el equilibrio de carga del servidor es la práctica de dividir el trabajo de manera uniforme entre varios servidores.

Un Servidor Balanceador de carga (SLB por sus siglas en inglés) es un dispositivo que distribuye la carga entre varias máquinas. Como se discutió anteriormente, tiene el efecto de hacer que varias máquinas aparezcan como una sola.

2.1.10. Virtual IP (VIP)

La IP virtual, también llamada interfaz sin circuito o loopback, es una función poderosa que proporciona una forma de asignar una o más direcciones al sistema sin la necesidad de vincular la dirección a una interfaz física. (IBM Knowledge Center, s.f.)

Los usos para VIP incluyen la traducción de direcciones de red (especialmente, NAT de uno a muchos), tolerancia a fallas y movilidad.

2.1.11. Redundancia

En ingeniería, la redundancia es la duplicación de componentes o funciones críticas de un sistema con la intención de aumentar la confiabilidad del sistema, generalmente en forma de respaldo o a prueba de fallas, o para mejorar el rendimiento real del sistema. (J. R. Sklaroff, 1976)

La redundancia como concepto es simple: si un dispositivo falla, otro ocupará su lugar y funcionará, con poco o ningún impacto en las operaciones en su conjunto. Hay varias formas de lograr esta funcionalidad. Por lo general, se implementan dos dispositivos. Un dispositivo usa un protocolo para verificar la salud de su compañero. . En algunos escenarios, ambos dispositivos están activos y aceptan tráfico, mientras que en otros, solo se usa un dispositivo mientras el otro espera en caso de falla.

Escenario Activo/En espera

El escenario de redundancia Activo/En espera es el más fácil de entender e implementar. Un dispositivo toma el tráfico mientras que el otro espera en caso de falla.

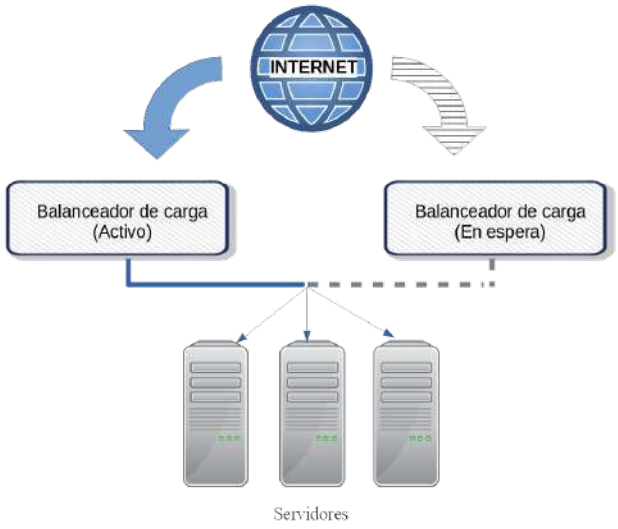


Figura 3: Escenario Activo/En espera

En caso de que el balanceador de carga Activo falle

Si el primer balanceador de carga fallara, el otro dispositivo se haría cargo del tráfico.

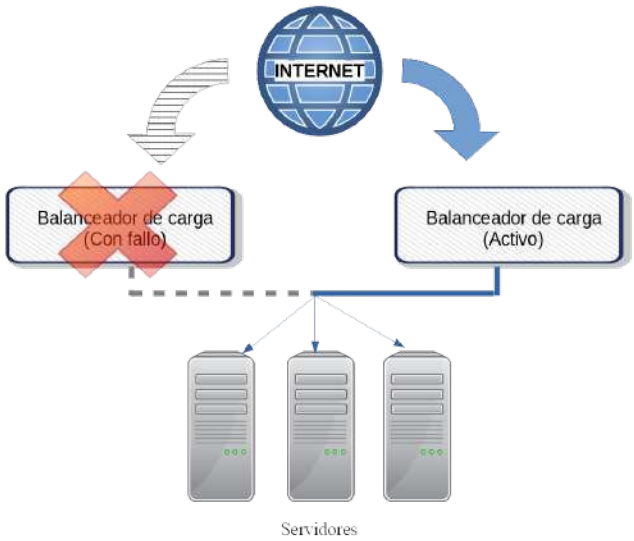


Figura 4: Escenario Activo/En espera con fallo

Escenario Activo/Activo

En este caso, ambas unidades aceptan tráfico.

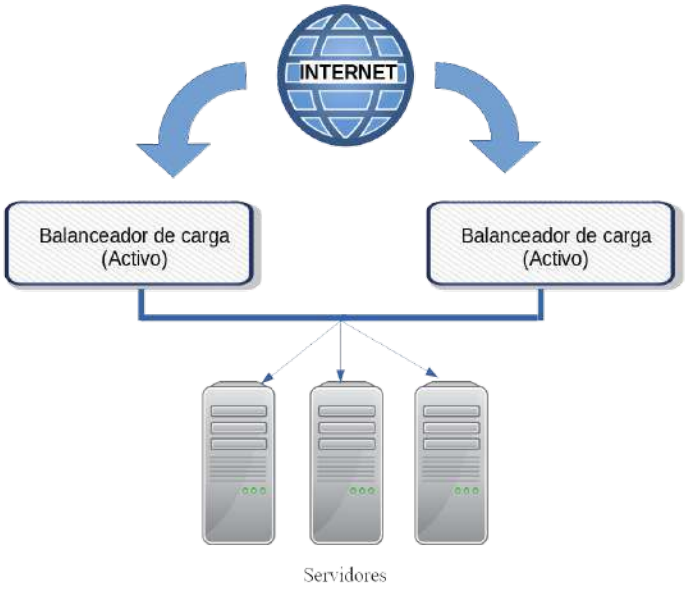


Figura 5: Escenario Activo/Activo

2.1.12. Replicación

La replicación de datos es la práctica de crear una o más copias redundantes de una base de datos u otro almacén de datos con el propósito de tolerancia a fallas. ("What is Data Replication: Definition | Informatica India", s.f.)

Pueden surgir problemas con la replicación de datos debido a la latencia o interrupciones del servicio durante la transferencia de datos. Los productos comerciales de replicación de datos intentan aliviar el riesgo comercial debido a errores o fallas de replicación. A medida que aumenta la distancia entre la fuente y el espejo, la replicación de datos puede ser más difícil.

Los principales beneficios de la replicación de datos son la recuperación ante desastres y la alta disponibilidad de aplicaciones de misión crítica. Si el origen de datos primario falla, se puede intercambiar una réplica de inmediato. También proporciona consistencia transaccional para que los datos estén actualizados y sean consistentes. Las herramientas de replicación de datos pueden reducir el trabajo de TI involucrado en la creación y administración de transacciones de replicación de datos en toda la empresa.

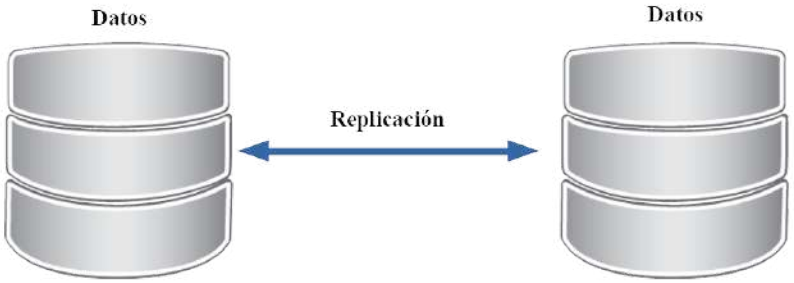


Figura 6: Replicación

2.1.13. Servidor Proxy

Un servidor proxy suele ser un sistema informático, una combinación de plataformas de hardware y aplicaciones de software, que sirve como intermediario en la comunicación de red entre las partes. (Sysel & Doležal, 2013)

Los sistemas cliente-servidor proporcionan un intermediario en la comunicación entre el cliente (generalmente enviando solicitudes) y el servidor (enviando la respuesta).

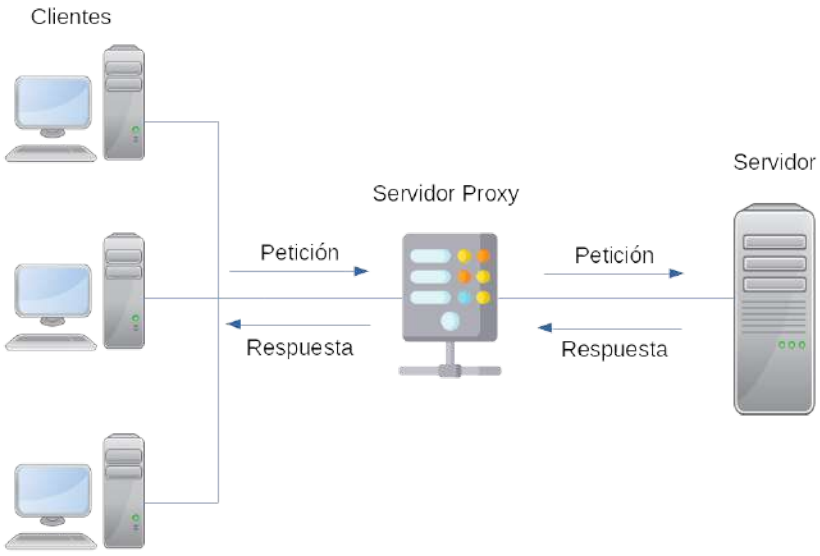


Figura 7: Servidor Proxy

Los servidores proxy ofrecen las siguientes funcionalidades básicas:

- Cortafuegos y filtrado de datos de red.
- Conexión de red compartida
- Almacenamiento en caché de datos

Los servidores proxy permiten ocultar y hacer que el ID de red sea anónima ocultando la dirección IP

Propósito de los servidores proxy

Las siguientes son las razones para usar servidores proxy:

- Monitoreo y filtrado: Se puede filtrar contenido (paquetes de datos potencialmente peligrosos) por lo que hace de cortafuegos, monitorear peticiones y el estado de red.

- Mejora del rendimiento: Recupera contenido del caché que se guardó cuando el cliente realizó una solicitud previa.
- Traducción: Ayuda a personalizar el sitio de origen para los usuarios locales al excluir el contenido de origen o sustituir el contenido de origen con contenido local original. En esto, el tráfico de los usuarios globales se enruta al sitio web de origen a través del proxy de traducción
- Acceso a servicios de forma anónima: En este caso, el servidor de destino recibe la solicitud del servidor proxy y no del usuario final.
- Seguridad: Dado que el servidor proxy oculta la identidad del usuario, por lo tanto, protege contra el correo no deseado y los ataques.

2.1.14. NAT

NAT habilita redes IP privadas que usan direcciones IP no registradas para conectarse a Internet. NAT opera en un dispositivo, generalmente conectando dos redes. Antes de que los paquetes se envíen a otra red, NAT traduce las direcciones privadas (no globalmente únicas) de la red interna en direcciones legales. ("IP Addressing: NAT Configuration Guide, Cisco IOS Release 15M&T - Configuring NAT for IP Address Conservation [Support]", 2018)

El rápido crecimiento de Internet resultó en una escasez de direcciones IPv4 disponibles. En respuesta, un subconjunto específico del espacio de direcciones IPv4 se designó como privado, para aliviar temporalmente este problema. Una dirección pública se puede enrutar en Internet. Por lo tanto, los dispositivos que deben tener acceso a Internet deben estar configurados con (o accesibles por) direcciones públicas. La asignación de direcciones públicas se rige por la Autoridad de Números Asignados de Internet (IANA). Una

dirección privada está destinada para uso interno dentro de un hogar u organización, y puede ser utilizada libremente por cualquier persona. Sin embargo, las direcciones privadas nunca se pueden enrutar en Internet. De hecho, los enrutadores de Internet están configurados para eliminar inmediatamente el tráfico con direcciones privadas. Se definieron tres rangos de direcciones privadas en RFC 1918, uno para cada clase de IPv4:

Se definieron tres rangos de direcciones privadas en RFC 1918, uno para cada clase de IPv4:

- Clase A - 10.x.x.x / 8
- Clase B - 172.16.x.x / 12
- Clase C - 192.168.x.x / 24

Es posible traducir entre direcciones privadas y públicas, utilizando Network Address Translation (NAT). NAT permite que un host configurado con una dirección privada se marque con una dirección pública, lo que permite que ese host se comunique a través de Internet. También es posible traducir varios hosts con direcciones privadas a una sola dirección pública, lo que conserva el espacio de direcciones públicas. NAT proporciona un beneficio adicional: oculta las direcciones específicas y la estructura de direccionamiento de la red interna (o privada).

Nota: NAT no está restringido a la traducción de direcciones privadas a públicas, aunque esa es la aplicación más común. NAT también puede realizar la traducción de direcciones de público a público, así como la traducción de direcciones de privado a privado.

2.1.15. Firewall o Cortafuegos

En 2003, Komar, Beekelaar y Wettern establecen que un firewall es una pieza de software o hardware que filtra todo el tráfico de red entre su computadora, red doméstica o red de la empresa e Internet.

Los firewalls también son importantes porque proporcionan un único "punto de entrada" donde se pueden imponer la seguridad y las auditorías. Un cortafuegos puede proporcionar al administrador de la red datos sobre qué tipo y cantidad de tráfico pasó a través de él, cuántos intentos se hicieron para entrar, etc. Como un sistema de televisión de seguridad de circuito cerrado, el cortafuegos no solo evita el acceso, sino que también también monitorea quién está escuchando y ayuda a identificar a quienes intentan violar la seguridad.

Propósito básico de un cortafuegos

Básicamente, un cortafuegos hace tres cosas para proteger su red:

1. Bloquea los datos entrantes que pueden contener un ataque de piratas informáticos.
2. Oculta información sobre la red al hacer que parezca que todo el tráfico saliente se origina en el cortafuegos en lugar de la red.
3. Detecta el tráfico saliente para limitar el uso de Internet y/o el acceso a sitios remotos.

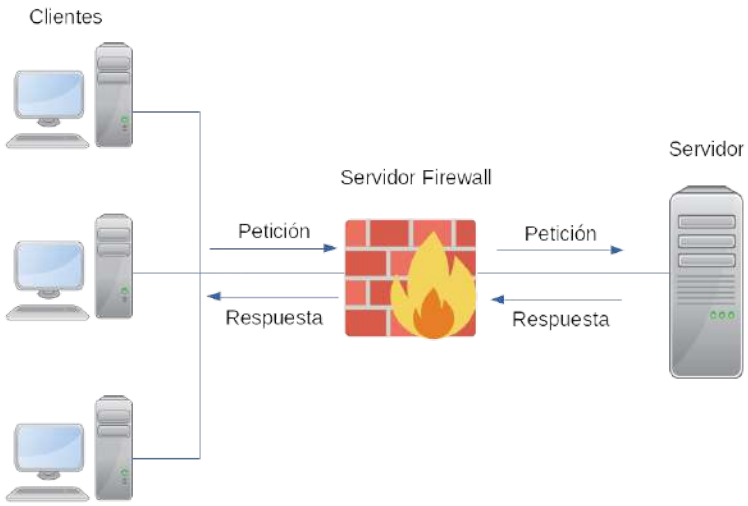


Figura 8: Servidor Firewall

2.2. Identificar los aplicaciones a ser instaladas para el Clúster.

2.2.1. HAProxy

HAProxy es una solución gratuita, muy rápida y confiable que ofrece alta disponibilidad, balanceo de carga y proxy para aplicaciones TCP y HTTP. Es especialmente adecuado para sitios web de mucho tráfico y ofrece un buen número de los más visitados del mundo. ("HAProxy - The Reliable, High Performance TCP/HTTP Load Balancer", s.f.)

Con el paso de los años, se ha convertido en el estándar de facto del balanceador de carga opensource, ahora se envía con la mayoría de las distribuciones de Linux, y a menudo se implementa de manera predeterminada en las plataformas de la nube.

Se sabe que HAProxy se ejecuta confiablemente en las siguientes plataformas:

- Linux 2.4 on x86, x86_64, Alpha, Sparc, MIPS, PARISC

- Linux 2.6 / 3.x on x86, x86_64, ARM, Sparc, PPC64
- Solaris 8/9 on UltraSPARC 2 and 3
- Solaris 10 on Opteron and UltraSPARC
- FreeBSD 4.10 - 10 on x86
- OpenBSD 3.1 to -current on i386, amd64, macppc, alpha, sparc64 and VAX (check the ports)AIX 5.1 - 5.3 on Power™ architecture

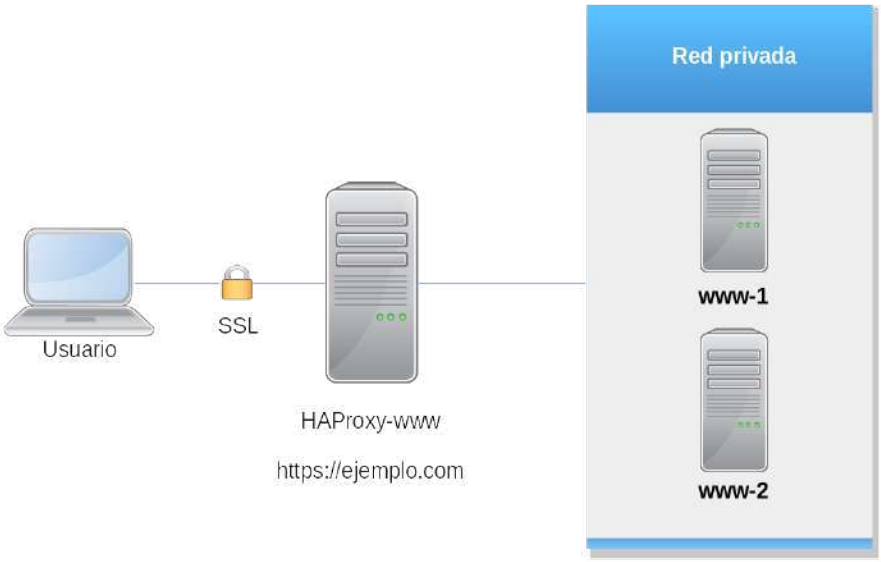


Figura 9: Servidor HAProxy

2.2.2. KeepAlived

Keepalived es un software de enrutamiento escrito en C. El objetivo principal de este proyecto es proporcionar instalaciones simples y robustas para el equilibrio de carga y alta

disponibilidad para el sistema Linux y las infraestructuras basadas en Linux. ("Keepalived for Linux", s.f.)

El marco de equilibrio de carga se basa en el módulo de kernel del servidor virtual Linux (IPVS) bien conocido y ampliamente utilizado que proporciona el equilibrio de carga capa 4 (Capa de transporte Modelo OSI). Keepalived implementa un conjunto de verificadores para mantener y administrar de manera dinámica y adaptativa el grupo de servidores con equilibrio de carga según su estado. Por otro lado, la alta disponibilidad se logra mediante el protocolo VRRP. Para ofrecer la detección de fallas de red más rápida, Keepalived implementa el protocolo BFD. La transición de estado VRRP puede tener en cuenta la sugerencia de BFD para impulsar una transición de estado rápida. Los frameworks Keepalived se pueden usar de forma independiente o todos juntos para proporcionar infraestructuras resistentes.

VRRP: En 2017, "First Hop Redundancy Protocols Configuration Guide, Cisco IOS XE Release 3S - Configuring VRRP [Cisco IOS XE 3S]" establece que el Protocolo de redundancia de enrutador virtual (VRRP) es un protocolo de elección que asigna dinámicamente la responsabilidad de uno o más enrutadores virtuales a los enrutadores VRRP en una LAN, permitiendo que varios enrutadores en un enlace de acceso múltiple utilicen la misma dirección IP virtual. Un enrutador VRRP está configurado para ejecutar el protocolo VRRP junto con uno o más enrutadores conectados a una LAN. En una configuración VRRP, se elige un enrutador como maestro del enrutador virtual, y los otros enrutadores actúan como copias de seguridad en caso de que falle el maestro del enrutador virtual.

BFD: En 2018, "IP Routing: BFD Configuration Guide, Cisco IOS Release 15M&T - Bidirectional Forwarding Detection [Cisco IOS 15.4M&T]" establece que el protocolo de detección de reenvío bidireccional (BFD) es un protocolo de detección diseñado para proporcionar tiempos de detección de fallas de ruta de reenvío rápido para todos los tipos de medios, encapsulaciones, topologías y protocolos de enrutamiento.

2.2.3. NFS (Network File System)

NFS (sistema de archivos de red: «Network File System») es un protocolo que permite acceso remoto a un sistema de archivos a través de la red. Todos los sistemas Unix pueden trabajar con este protocolo; cuando se involucran sistemas Windows, debe utilizar Samba en su lugar. ("11.4. Servidor de archivos NFS", s.f.)

NFS es una herramienta muy útil. Si bien anteriormente ha tenido muchas limitaciones, la mayoría ha desaparecido con la versión 4 del protocolo. El inconveniente es que la última versión de NFS e más difícil de configurar cuando se quieren utilizar funciones básicas de seguridad como la autenticación y el cifrado, puesto que se basa en Kerberos para estas funcionalidades.

2.2.4. Postfix

Postfix Es el servidor de correo de Wietse Venema que comenzó su vida en la investigación de IBM como una alternativa al ampliamente utilizado programa Sendmail . Ahora en Google, Wietse continúa apoyando a Postfix. ("The Postfix Home Page", s.f.)

Postfix puede ejecutarse en sistemas tipo UNIX que incluyen AIX, BSD, HP-UX, Linux, MacOS X, Solaris y más.

2.2.5. MariaDB

MariaDB Server es uno de los servidores de bases de datos más populares del mundo. Está hecho por los desarrolladores originales de MySQL y está garantizado para ser de código abierto. Los usuarios notables incluyen Wikipedia, WordPress.com y Google. (Charles, 2018)

MariaDB convierte los datos en información estructurada en una amplia gama de aplicaciones, desde bancos hasta sitios web. Es un reemplazo mejorado y de reemplazo directo para MySQL. MariaDB se usa porque es rápido, escalable y robusto, con un rico ecosistema de motores de almacenamiento, complementos y muchas otras herramientas que lo hacen muy versátil para una amplia variedad de casos de uso.

2.2.6. Apache HTTP server

Apache. El Proyecto Apache HTTP Server es un esfuerzo para desarrollar y mantener un servidor HTTP de código abierto para los sistemas operativos modernos, incluidos UNIX y Windows. El objetivo de este proyecto es proporcionar un servidor seguro, eficiente y extensible que proporcione servicios HTTP en sincronización con los estándares HTTP actuales. ("Welcome! - The Apache HTTP Server Project", s.f.)

El Servidor Apache HTTP ("httpd") se lanzó en 1995 y ha sido el servidor web más popular en Internet desde abril de 1996. En febrero de 2015, celebró su vigésimo aniversario como proyecto.

2.2.7. OpenSSH

OpenSSH es la principal herramienta de conectividad para inicio de sesión remoto con el protocolo SSH. Encripta todo el tráfico para eliminar el espionaje, el secuestro de la conexión y otros ataques. Además, OpenSSH proporciona un amplio conjunto de capacidades seguras de túnel, varios métodos de autenticación y opciones de configuración sofisticadas. ("OpenSSH", s.f.)

El conjunto de OpenSSH consta de las siguientes herramientas:

- Las operaciones remotas se realizan utilizando ssh , scp y sftp.
- Gestión de claves con ssh-add , ssh-keysign , ssh-keyscan y ssh-keygen.
- El lado del servicio consta de sshd , sftp-server y ssh-agent.

2.2.8. Rsync

Rsync es una herramienta de copia de archivos rápida y extraordinariamente versátil. Puede copiar localmente, a/desde otro host a través de cualquier shell remoto, o a/desde un demonio rsync remoto. Ofrece una gran cantidad de opciones que controlan cada aspecto de su comportamiento y permiten una especificación muy flexible del conjunto de archivos a copiar. ("rsync(1) - Linux man page", s.f.)

Es famoso por su algoritmo de transferencia delta, que reduce la cantidad de datos enviados a través de la red al enviar sólo las diferencias entre los archivos de origen y los archivos existentes en el destino. Rsync se usa ampliamente para copias de seguridad y duplicación y como un comando de copia mejorado para el uso diario.

Rsync encuentra archivos que necesitan ser transferidos usando el algoritmo lqquick checkrq (por defecto) que busca archivos que han cambiado de tamaño o en el último tiempo modificado. Cualquier cambio en los otros atributos conservados (según lo solicitado por las opciones) se realiza directamente en el archivo de destino cuando la verificación rápida indica que no es necesario actualizar los datos del archivo.

Algunas de las características adicionales de rsync son:

- Soporte para copiar enlaces, dispositivos, propietarios, grupos y permisos
- Opción de “excluir” y “excluir de” similiar a GNU tar
- Un modo de exclusión de CVS para ignorar los mismos archivos que CVS ignoraría
- Puede usar cualquier shell remoto, incluidos ssh o rsh
- No requiere privilegios de super usuario
- Canalización de transferencias de archivos para minimizar los costos de latencia
- Soporte para demonios rsync autenticados o anónimos (ideal para duplicación)

2.3. Realizar monitoreo del Clúster

En un estudio anterior (A. Bustos, 2007) dice que el tener monitorizado un sistema es muy importante en el proceso de administración del cluster ya que mediante los datos que arroje dicho monitoreo se puede prever cual es el comportamiento de un determinado dispositivo que compone el cluster, y con ello poder tomar los correctivos necesarios para evitar una caída del sistema.

Hay dos propósitos diferentes detrás de la supervisión de cualquier sistema informático, que básicamente se asignan a dos funciones de administración de sistemas diferentes. Los administradores de sistemas y los programadores de sistemas se preocupan

más por la instalación, el ajuste y la resolución de problemas de los sistemas bajo su control. Los planificadores de capacidad se preocupan más por construir una base de datos de rendimiento para analizar y predecir el consumo de recursos a lo largo del tiempo, buscando predecir el crecimiento y las actualizaciones necesarias para mantener ese crecimiento. Debido a esta diferencia de propósito, estos grupos requerirán diferentes herramientas, aunque hay áreas definidas de superposición. La administración de sistemas generalmente requiere herramientas para mostrar lo que está sucediendo en este momento, mientras que la planificación de la capacidad tiende a estar más preocupada por el uso de recursos en tendencia para reconocer el crecimiento y los cuellos de botella futuros, con el tiempo. Hay una superposición, por supuesto, ya que no puede resolver problemas de administración de sistemas sin comprender los rangos aceptables de las métricas de rendimiento de los sistemas. Entonces, cuando la función de planificación de capacidad necesita métricas históricas para predecir cuellos de botella de crecimiento futuros, la función de gestión del rendimiento necesita datos históricos similares para comprender cómo reconocer qué buscar en las métricas y qué valores son una indicación de un problema de rendimiento.

2.3.1. Métricas

Es un método para medir algo, o los resultados obtenidos de esto ("metrics | Definition of metrics by Lexico", s.f.). En otras palabras Las métricas son aquellos datos expresados numéricamente que nos sirven para **analizar el rendimiento** de un sistema informático específico.

Las métricas que nos interesan son muy similares para cualquier sistema informático. Necesitamos medir el uso de todos los recursos físicos:

- Memoria
- CPU
- Dispositivos de almacenamiento
- Red

También necesitamos medir el uso de recursos a nivel del sistema que pueden afectar el rendimiento y la capacidad:

- Swap

Puede haber más recursos a nivel de sistema dependiendo de las aplicaciones en uso

2.3.2. Utilización de CPU

Según (Derek Haynes, 2015) en Linux, no solo nos importa el porcentaje de CPU disponible utilizada, sino también el uso por diferentes estados. Hay 3 estados generales en los que puede estar su CPU:

- **Inactivo**, lo que significa que no tiene nada que hacer.
- **Ejecución de un programa en espacio de usuario**, como un shell de comandos, un servidor de correo electrónico o un compilador.
- **Ejecutando el kernel**, sirviendo interrupciones o administrando recursos.

2.3.3. Memoria RAM

RAM es el acrónimo del concepto inglés de Random Access Memory (Memoria de Acceso Aleatorio). Se trata de la memoria que, en un equipo informático, es utilizada por un procesador para recibir instrucciones y guardar los resultados ("Definición de RAM — Definicion.de", s.f.).

Linux usa tanto memoria real como archivos de intercambio, similar a la memoria virtual. Por lo general, el archivo de intercambio (SWAP) se define como el doble de la cantidad de memoria real.

Si un sistema Linux necesita más memoria real de la disponible, utilizará su archivo de intercambio para liberar algo de memoria y permitir que la memoria de otro proceso resida en real.

2.3.4. SWAP

Según ("Chapter 13. Swap Space", s.f.) el espacio de intercambio en Linux se usa cuando la cantidad de memoria física (RAM) está llena. Si el sistema necesita más recursos de memoria y la RAM está llena, las páginas inactivas en la memoria se mueven al espacio de intercambio. Si bien el espacio de intercambio puede ayudar a las máquinas con una pequeña cantidad de RAM, no debe considerarse un reemplazo para más RAM. El espacio de intercambio se encuentra en los discos duros, que tienen un tiempo de acceso más lento que la memoria física.

2.3.5 Herramientas de monitoreo

Top

Según ("top(1) - Linux manual page", s.f.) el programa **Top** proporciona una vista dinámica en tiempo real de un sistema en ejecución. Puede mostrar un resumen del sistema, así como una lista de procesos o subprocesos actualmente gestionados por el kernel de Linux.

Los campos importantes para mirar en la salida superior incluyen:

Promedio de carga: estos tres números muestran el Número de procesos ejecutables en la cola de CPU durante el último minuto, cinco minutos y quince minutos. Estos números deben compararse con el número de CPU disponibles para ese Linux.

Estados de CPU: esta línea muestra el porcentaje utilización para cada uno de los cuatro estados de CPU posibles.

Memoria y Swap: estas dos líneas muestran cuánto de la memoria real y swap están disponibles, en uso y libres. También se muestra la memoria utilizada para memorias intermedias, caché y memoria compartida.

Información de proceso:

- PID: es el identificador de proceso. Cada proceso tiene un identificador único.
- USER (USUARIO): usuario propietario del proceso.
- PR: prioridad del proceso. Si pone *RT* es que se está ejecutando en tiempo real.
- NI: asigna la prioridad. Si tiene un valor bajo (hasta -20) quiere decir que tiene más prioridad que otro con valor alto (hasta 19).
- VIRT: cantidad de memoria virtual utilizada por el proceso.
- RES: cantidad de memoria RAM física que utiliza el proceso.
- SHR: memoria compartida.
- S (ESTADO): estado del proceso.
- %CPU: porcentaje de CPU utilizado desde la última actualización.
- %MEM: porcentaje de memoria física utilizada por el proceso desde la última actualización.
- TIME+ (HORA+): tiempo total de CPU que ha usado el proceso desde su inicio.

- COMMAND (ORDEN): comando utilizado para iniciar el proceso.

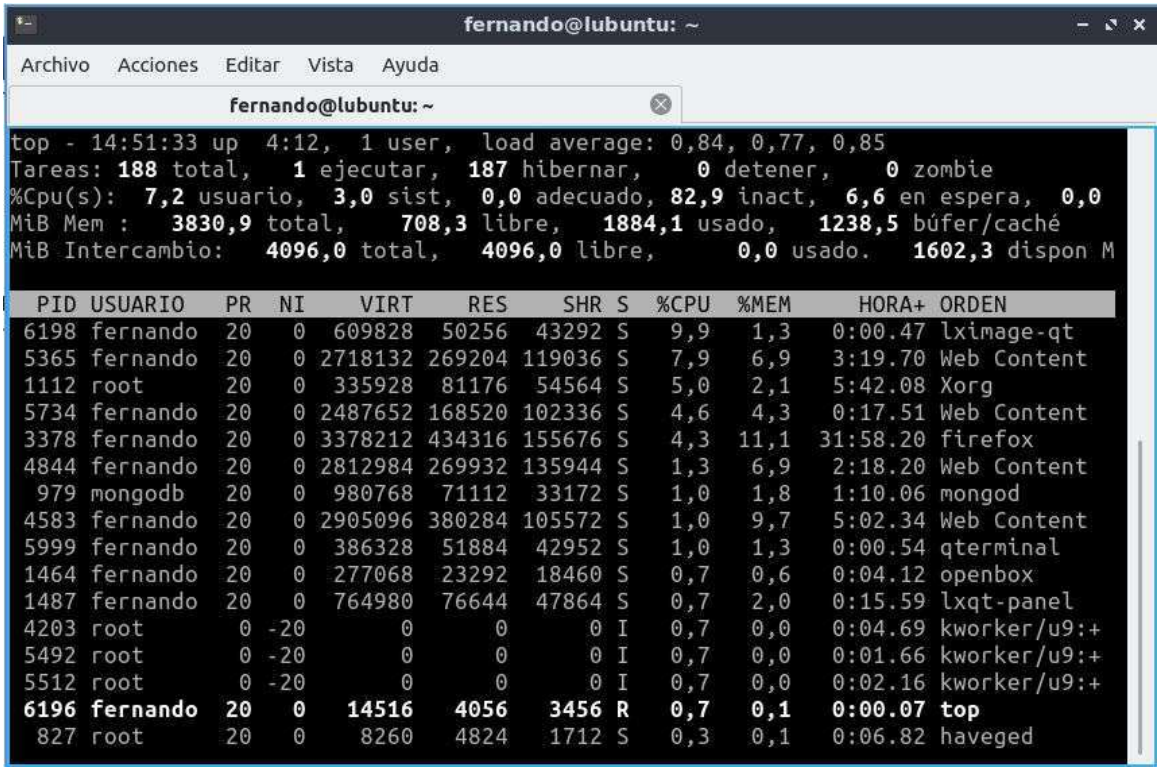


Figura 10: Comando “top”

Netstat

Según ("netstat(1) - OpenBSD manual pages", s.f.) netstat (estadísticas de red) es una herramienta de línea de comandos que muestra conexiones de red (tanto entrantes como salientes), tablas de enrutamiento y una serie de estadísticas de interfaz de red. Está disponible en los sistemas operativos Unix, Unix-like y Windows NT.

Netstat proporciona estadísticas para lo siguiente:

Proto: el nombre del protocolo (TCP o UDP).

Dirección local: la dirección IP de la computadora local y el número de puerto que se está utilizando. Se muestra el nombre de la computadora local que corresponde a la

dirección IP y el nombre del puerto a menos que se especifique el parámetro -n. Se muestra un asterisco (*) para el host si el servidor está escuchando en todas las interfaces. Si el puerto aún no está establecido, el número de puerto se muestra como un asterisco.

Dirección remota: la dirección IP y el número de puerto de la computadora remota a la que está conectado el zócalo. Los nombres que corresponden a la dirección IP y el puerto se muestran a menos que se especifique el parámetro -n. Si el puerto aún no está establecido, el número de puerto se muestra como un asterisco (*).

Estado: indica el estado de una conexión TCP. Los estados posibles son los siguientes: CLOSE_WAIT, CLOSED, ESTABLISHED, FIN_WAIT_1, FIN_WAIT_2, LAST_ACK, LISTEN, SYN_RECEIVED, SYN_SEND, and TIME_WAIT.. Para obtener más información sobre los estados de una conexión TCP, consulte RFC 793.

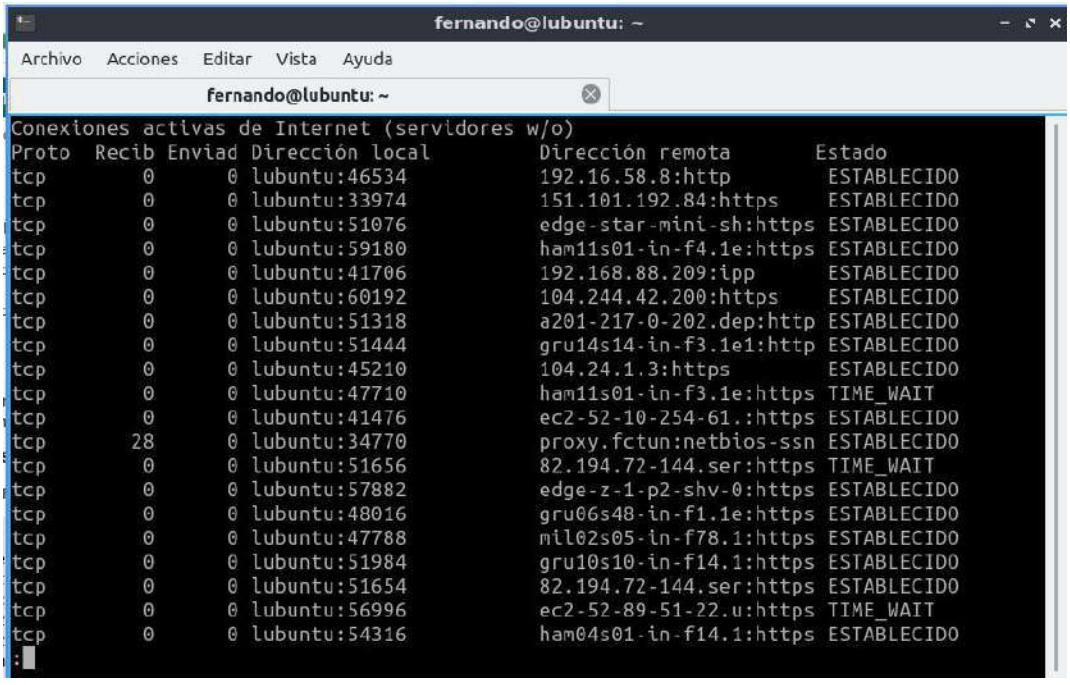


Figura 11: Comando netstat

Nagios Core

Nagios Core sirve como el planificador de eventos básico, el procesador de eventos y el administrador de alertas para los elementos que se monitorean. Cuenta con varias API que se utilizan para ampliar sus capacidades para realizar tareas adicionales, se implementa como un demonio escrito en C por razones de rendimiento y está diseñado para ejecutarse de forma nativa en sistemas Linux / * nix. ("Nagios Core. Download Nagios Core For Free Here.", s.f.)

Nagios Core es un software de código abierto con licencia de GNU GPL V2.

Actualmente proporciona:

- Monitoreo de servicios de red (SMTP, POP3, HTTP, NNTP, ICMP, SNMP, FTP, SSH)
- Monitoreo de los recursos del host (carga del procesador, uso del disco, registros del sistema) en la mayoría de los sistemas operativos de red, incluido Microsoft Windows, utilizando agentes de monitoreo.
- Y mucho más.

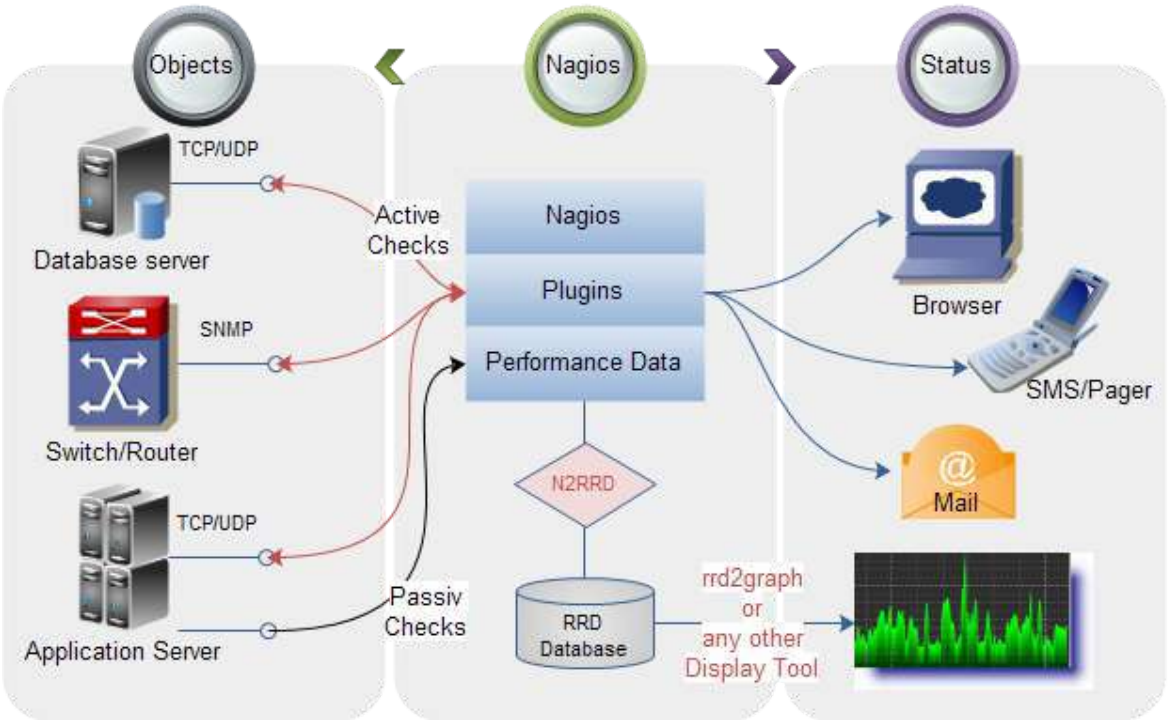


Figura 12: Funcionamiento de Nagios Core

CAPÍTULO III

MARCO METODOLÓGICO

3.1 Tipo de Estudio

Cualitativo, diseño no experimental. La presente investigación de diseño no experimental, tiene un enfoque cualitativo por lo que se efectuará una recolección de datos mediante la aplicación de herramienta como la entrevista(Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2010).

3.2 Diseño de Investigación

3.2.1. Entrevista

Se utilizó la entrevista, la cual se define como una reunión para intercambiar información entre una persona y otra, al encargado de la parte informática, obteniendo los datos necesarios para el análisis de los mismos (Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2010).

3.3 Población y Muestra

(Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2010). Expresan cuanto sigue:

> Los tipos de muestras que suelen utilizarse en las investigaciones son las no probabilísticas o dirigidas, cuya finalidad no es la generalización en términos de probabilidad. También se les conoce como "guiadas por uno o varios propósitos", pues la elección de los elementos depende de razones relacionadas con las características de la investigación (pág. 396).

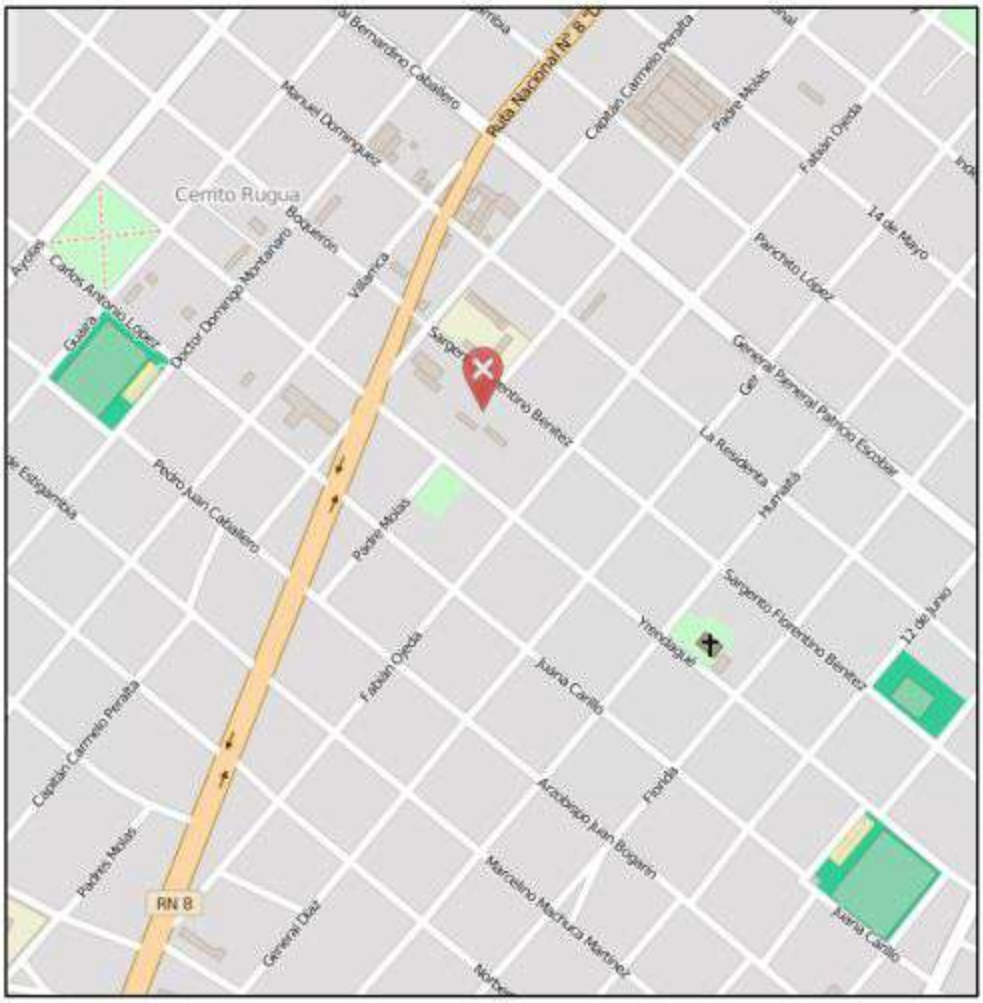
La población está compuesta por el Departamento de coordinación y gestión de soporte informático de la Facultad de Ciencias y Tecnologías de la Universidad Nacional de Caaguazú, que han participado como muestra de esta investigación. Según (Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2010).

3.4 Métodos, Técnicas y Procedimientos

Se harán entrevistas no estructuradas a los encargados de TI de la Facultad de Ciencias y Tecnologías, además se hará revisión a la documentación existente, ya sea documentación existente sobre el sistema informático actual, así como documentación existente sobre diseño e implementación de un clúster recopilado de Internet.

3.5 Localización Física

Sargento Florentino Benítez entre Capitán Carmelo Peralta y Fabián Ojeda.



Localización física de la Facultad de Ciencias y Tecnología.

CAPÍTULO IV

4. MARCO ANALÍTICO

4.1 Estudio de Factibilidad Tecnológica

Este estudio es indispensable de realizar ya que ayuda a visualizar si la Facultad de Ciencias y Tecnologías cuenta con las herramientas tecnológicas adecuadas de tal forma a que el normal funcionamiento del sistema no sea afectado.

Como parte del desarrollo del proyecto se cuenta con los equipos (hardware), y software necesario para llevar a cabo el desarrollo del presente proyecto.

4.2 Recursos necesarios para la elaboración del proyecto

4.2.1 Recursos Hardware para el prototipo (Empresa Vultr.com)

Concepto	Características	Precio por mes	Total (3 meses)
Dominio	-	14.99 USD/año	14.99 USD/año
VPS pfSense	2 GB RAM - 1 CPU 3.2 GHz - 50 GB SSD	20 USD	60 USD
VPS HAProxy * 2	1 GB RAM - 1 CPU 3.2 GHz - 25 GB SSD	10 USD	30 USD
VPS Apache * 2	1 GB RAM - 1 CPU	10 USD	30 USD

	3.2 GHz - 25 GB SSD		
VPS Tomcat * 2	1 GB RAM - 1 CPU 3.2 GHz - 25 GB SSD	10 USD	30 USD
VPS NextCloud * 2	1 GB RAM - 1 CPU 3.2 GHz - 25 GB SSD	10 USD	30 USD
VPS NFS * 2	1 GB RAM - 1 CPU 3.2 GHz - 25 GB SSD	10 USD	30 USD
VPS MariaDB * 2	1 GB RAM - 1 CPU 3.2 GHz - 25 GB SSD	10 USD	30 USD
VPS PostgreSQL * 2	1 GB RAM - 1 CPU 3.2 GHz - 25 GB SSD	10 USD	30 USD
VPS Nagios * 1	1 GB RAM - 1 CPU 3.2 GHz - 25 GB SSD	5 USD	15 USD
			TOTAL: 299.99 USD

Tabla 1: Costo del prototipo

4.2.2 Recursos Hardware para el proyecto (Promedio de costo por Servidor VPS en Paraguay)

Concepto	Características	Precio por año
Dominio	-	97.000 Gs.
VPS pfSense * 1	2 GB RAM - 1 CPU 3.2 GHz - 20 GB SSD	2.400.000 Gs.
VPS HAProxy * 2	1 GB RAM - 1 CPU 3.2	2.400.000 Gs.

	GHz - 25 GB SSD	
VPS Apache * 2	1 GB RAM - 1 CPU 3.2 GHz - 25 GB SSD	2.400.000 Gs.
VPS Tomcat * 2	1 GB RAM - 1 CPU 3.2 GHz - 25 GB SSD	2.400.000 Gs.
VPS NextCloud * 2	1 GB RAM - 1 CPU 3.2 GHz - 25 GB SSD	2.400.000 Gs.
VPS NFS * 2	1 GB RAM - 1 CPU 3.2 GHz - 25 GB SSD	2.400.000 Gs.
VPS MariaDB * 2	1 GB RAM - 1 CPU 3.2 GHz - 25 GB SSD	2.400.000 Gs.
VPS PostgreSQL * 2	1 GB RAM - 1 CPU 3.2 GHz - 25 GB SSD	2.400.000 Gs.
VPS Nagios * 1	1 GB RAM - 1 CPU 3.2 GHz - 25 GB SSD	1.200.000 Gs.
		TOTAL: 20.497.000 Gs.

Tabla 2: Costo del clúster con servidores paraguayos

4.2.3 Recursos Software

No se prevé ningún gasto en concepto de licencia por utilización de software, todo lo necesario es software libre o por lo menos software de código abierto que no tiene costo.

Descripción	**Costo**
PfSense	0 Gs.
HAProxy	0 Gs.
MariaDB	0 Gs.
Postgres	0 Gs.
PhpMyadmin	0 Gs.
Keepalived	0 Gs.

Apache	0 Gs.
Tomcat	0 Gs.
Nextcloud	0 Gs.
NFS	0 Gs.

Tabla 3: Recursos Software

4.2.4 Recursos Materiales

Descripción	**Costo**
Resma de papel tamaño carta	120.000 Gs.
Cartuchos para impresora	240.000 Gs.
TOTAL	360.000 Gs.

Tabla 4: Recursos materiales

4.2.5 Recursos Humanos

Cantidad	**Cargo**	**Costo individual/mes**	**Meses**	**Total**
2	Administrador de Servidores	3.500.000 Gs.	3	21.000.000 Gs.

Tabla 5: Recursos humanos

4.3 Análisis

4.3.1 Estructura actual de los servidores de la Facultad

Frontend pfSense

El frontend es un servidor VPS con una distribución pfSense que está basado en FreeBSD. Consta de dos tarjetas de red. Una IP pública y otra IP privada.

Sirve de Firewall y Router, bloquea todos los puertos por defecto y habilita solamente las peticiones en el puerto 80 y 443, además de puertos para conexión ssh hacia los nodos.

Todas las peticiones en el puerto 80 y 443 son redirigidos al VPS de HAProxy

Middleware HAProxy

Haproxy recibe las peticiones de pfSense en el puerto 80 y 443, configura las url y envía la petición al servidor VPS correspondiente. En HAProxy se configura los certificados SSL con Let's Encrypt.

Backend Nodos

Los nodos poseen una IP estática privada, no poseen IP Pública.

1 Nodo - 1 Servicio

Ejemplo: Se requiere la puesta en marcha de un sitio web para la consulta de notas. Se crea el VPS para esa finalidad. Se debe instalar Apache, MariaDB, PHP y otros paquetes necesarios para poner en funcionamiento la página. En caso de necesitar otro sitio, se debe repetir el proceso de instalación para el nuevo VPS

1 Nodo caído - 1 Servicio que no funciona

Si el nodo que servía una página para consulta de notas deja de funcionar, por una mala configuración o exceso de concurrencia, la página deja de funcionar.

Esquema actual

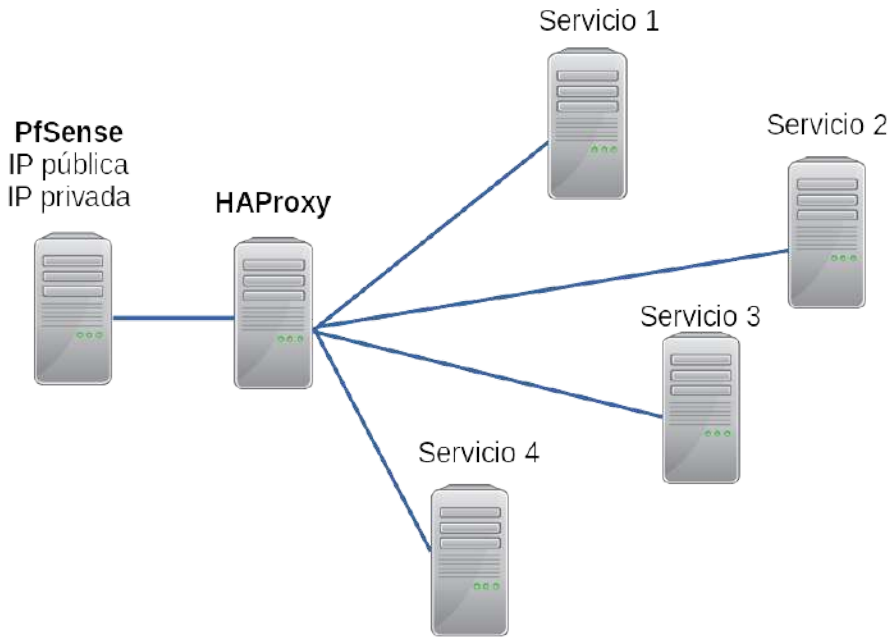


Figura 13: Esquema de servidores actual

4.3.2. Estadísticas de uso de Hardware del Servidor pfSense

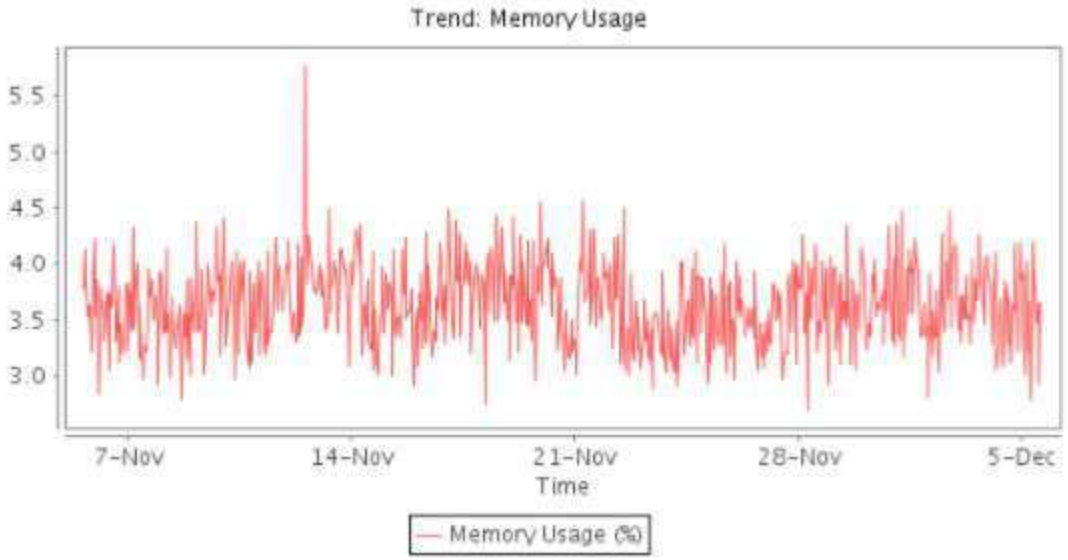


Figura 14: Uso de memoria RAM pfsense

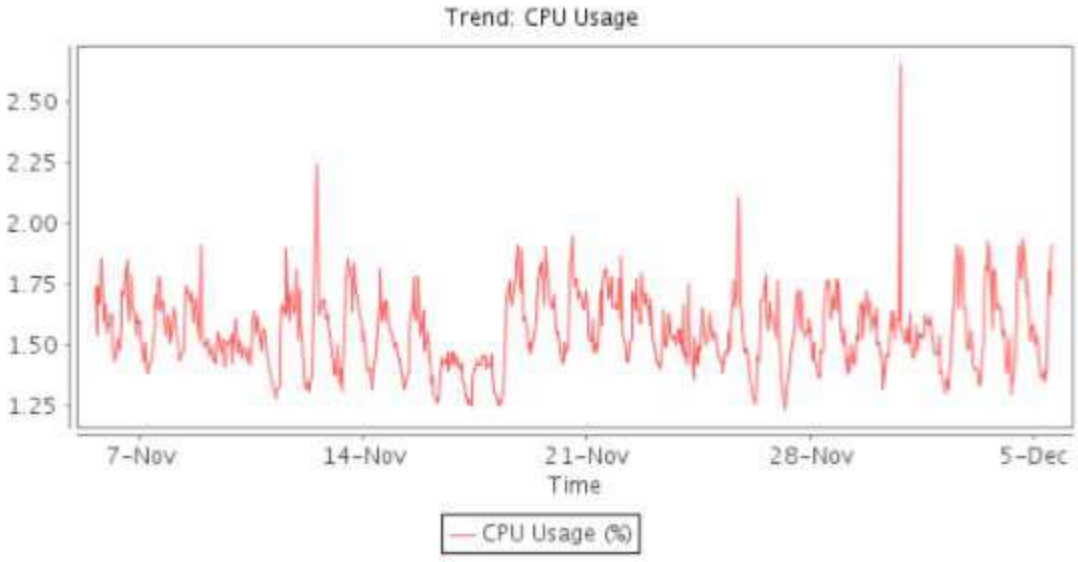


Figura 15: Uso de CPU pfsense

4.3.3. Estadísticas de uso de Hardware del Servidor Página Web Facultad de Ciencias y Tecnologías

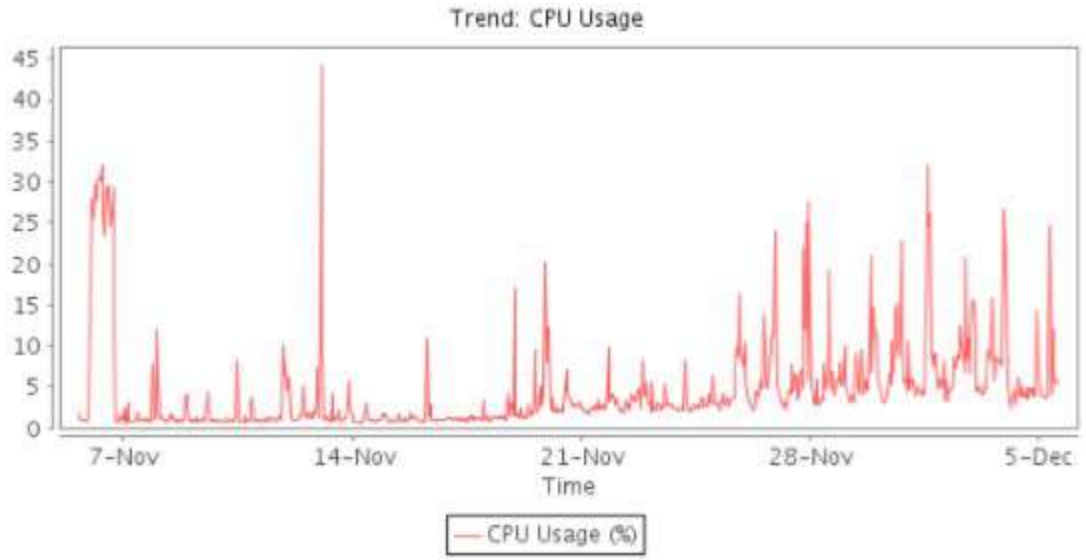


Figura 16: Uso de CPU fctunca.edu.py

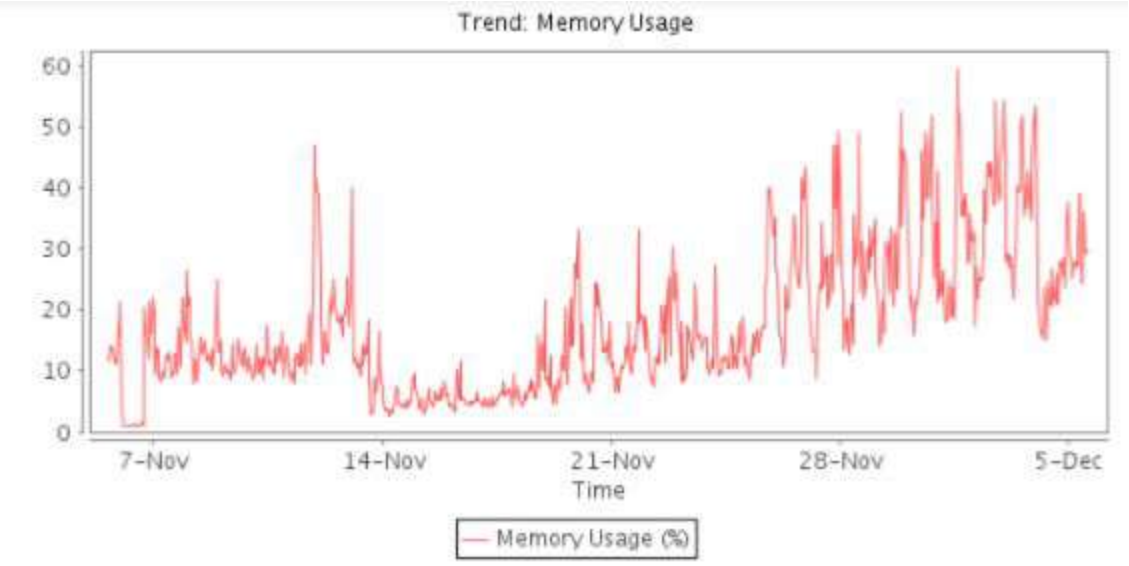


Figura 17: Uso de memoria RAM fctunca.edu.py

4.4 Diseño

El diseño propuesto para solucionar el problema de 1 Nodo caído - 1 Servicio que no funciona es duplicar los nodos y utilizar estos nodos para múltiples proyectos.

Frontend

No habrá cambios para el Frontend, seguirá funcionando de la misma manera.

El frontend es un servidor VPS con una distribución pfSense que está basado en FreeBSD. Consta de dos tarjetas de red. Una IP pública y otra IP privada.

Sirve de Firewall y Router, bloquea todos los puertos por defecto y habilita solamente las peticiones en el puerto 80 y 443, además de puertos para conexión ssh hacia los nodos.

Todas las peticiones en el puerto 80 y 443 son redirigidos al VPS de HAProxy

Middleware HAProxy

Se añade 1 Nodo HAProxy, con exactamente la misma configuración, estos dos nodos comparten una IP Virtual y será de tipo Master - Backup. El master trabaja constantemente hasta que por algún motivo deja de funcionar, entonces el Backup toma el lugar del Master.

Haproxy recibe las peticiones de pfSense en el puerto 80 y 443, configura las url y envía la petición al servidor VPS correspondiente. En HAProxy se configura los certificados SSL con Let’s Encrypt.

Backend Nodos

Habrá dos nodos por servicio, del tipo Master - Master. El HAProxy hará un balanceo de carga, es decir, el Master con menos conexiones recibirá la siguiente petición. Con esto se logra el Balanceo de Carga

2 Nodo - 1 Servicio

Ejemplo: Apache2 consta de dos nodos que hacen exactamente lo mismo, con exactamente los mismos datos. Se colocarán todos los servicios que requieran de Apache en estos nodos.

Se centraliza la base de datos en dos nodos MariaDB y en dos nodos PostgreSQL. Además el almacenamiento se centraliza en dos Nodos NFS.

La idea central de colocar los nodos en esta configuración es que siempre esté disponible un nodo brindando el servicio requerido. Con esto se logra la Alta Disponibilidad

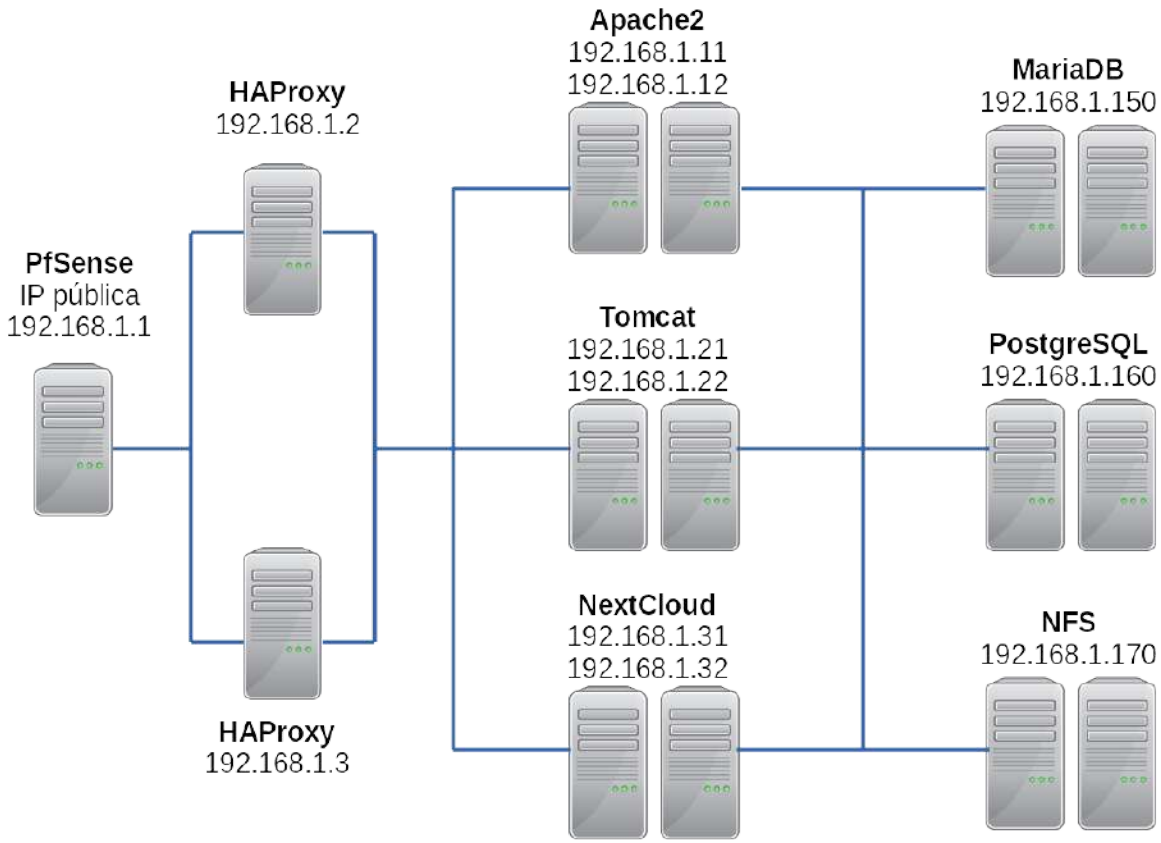


Figura 18: Estructura final del clúster

CAPÍTULO V

INSTALACIÓN DEL CLÚSTER

5.1. Instalación de pfSense

5.1.1. Requisitos mínimos de hardware

Los requisitos mínimos de hardware para pfSense® 2.4.4-RELEASE-p1 son:

- CPU de 600 MHz o más rápida
- RAM 512 MB o más
- Unidad de disco de 4 GB o más (SSD, HDD, etc.)
- Una o más tarjetas de interfaz de red compatibles
- Unidad USB de arranque o CD / DVD-ROM para la instalación inicial

5.1.2. Consideraciones de hardware

Al seleccionar hardware para una nueva compilación, considere cuidadosamente los requisitos de hardware actuales y futuros. Éstos incluyen:

- CPU Intel o AMD de 64 bits (x86-64, amd64) en pfSense 2.4 y posterior
- Debe poder arrancar desde USB o unidad óptica y ejecutar el instalador en pfSense 2.4 y posterior

Versión a instalar

- pfSense-CE-2.4.4-RELEASE-p1

5.1.3. Proceso de instalación

Nota: La preparación del medio de instalación queda a cargo del lector

Configurar un dominio para la instalación

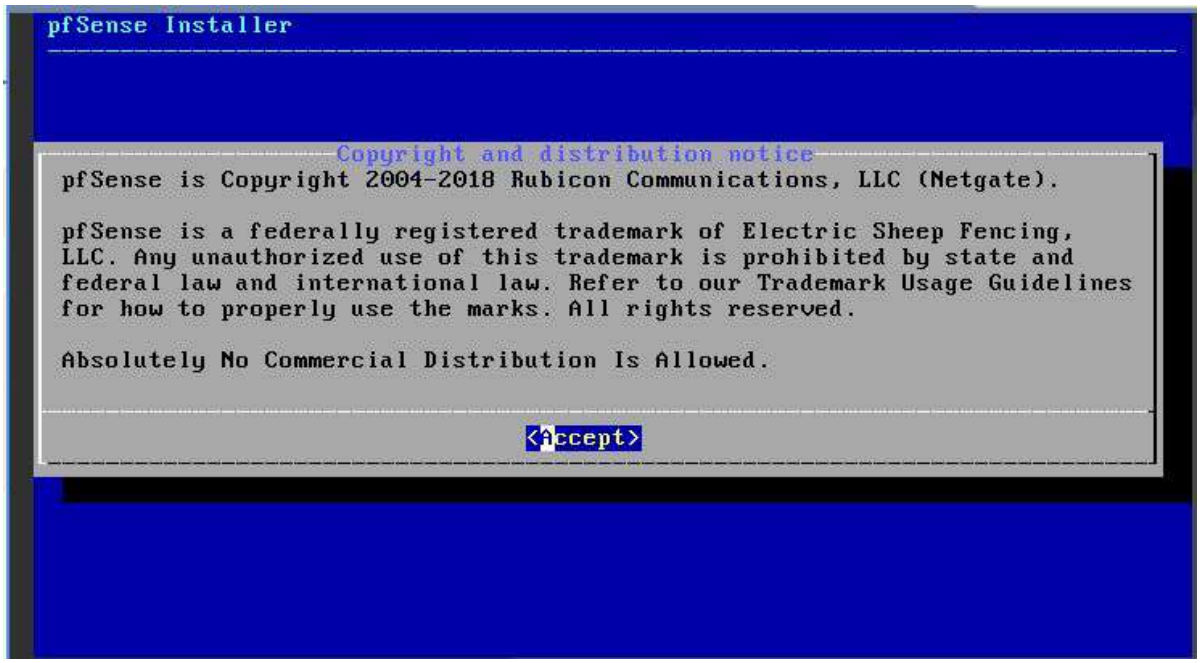
Se necesitará un dominio con una IP pública y estática. El dominio deberá estar configurado por lo menos de la siguiente manera:

	Type	Name	Data	TTL (seconds)	Priority	Actions
	A	Example: www	Example: 0.0.0.0	3600		+
	A		149.28.103.24	300		
	CNAME	*	usemoslinux.net	300		
	MX		usemoslinux.net	300	10	
	NS		ns1.vultr.com	300		
	NS		ns2.vultr.com	300		

Configuración DNS para el dominio

Como se podrá observar el dominio para esta instalación será **usemoslinux.net**

1. **Aceptamos el Copyright**



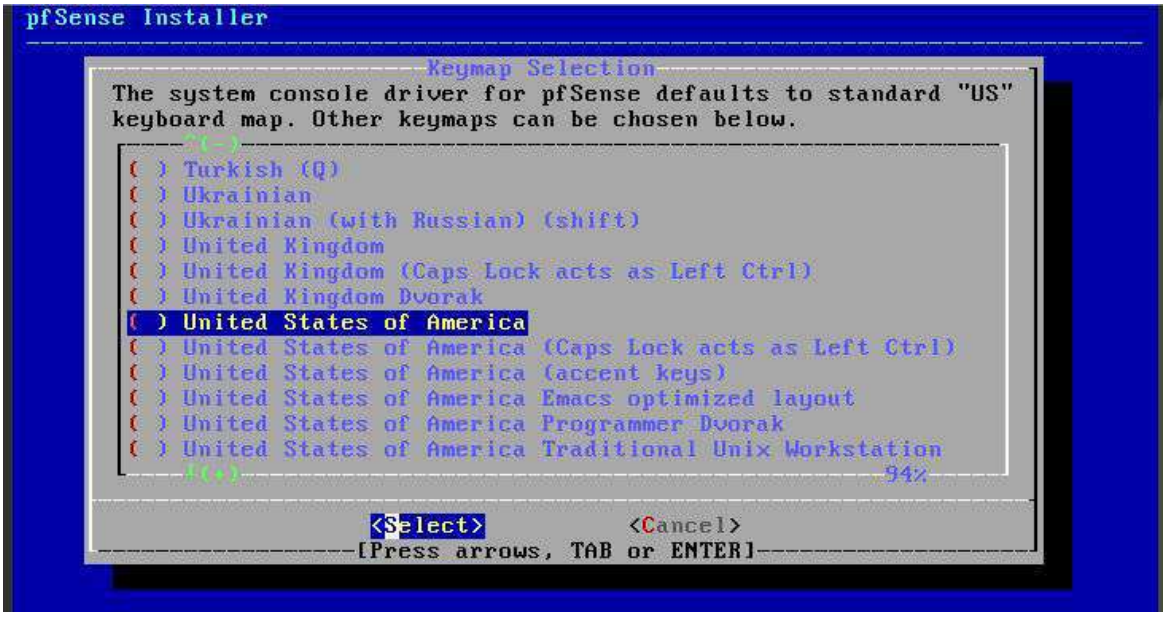
pfSense 1

2. **Elegimos la opción instalar**

pfSense 2

3. **Elegimos la distribución de teclado**

Para esta instalación elegimos la distribución “United States of America”



pfSense 3

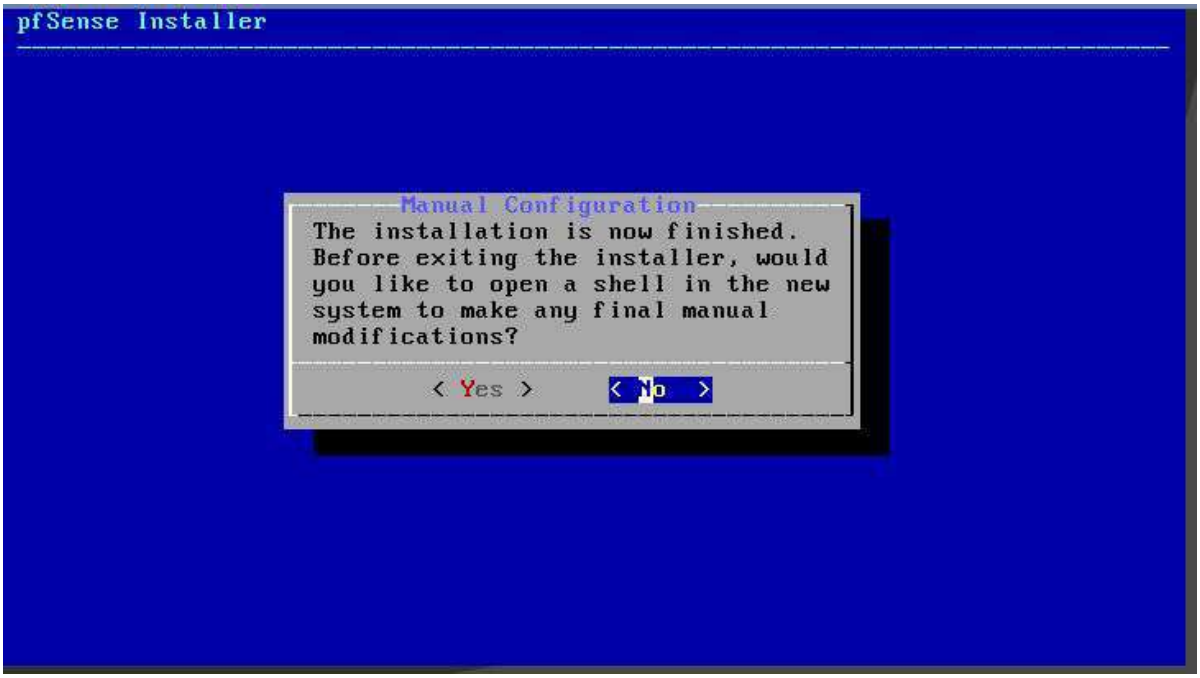
4. Particionado de disco

Elegimos la primera opción

pfSense 4

5. **Fin del proceso de instalación**

   El proceso de instalación no representa ninguna dificultad real, en unos minutos se completa y es hora de reiniciar, pero primero decimos que no queremos realizar ninguna modificación manual al sistema.



pfSense 5

6. **Quitamos el medio de instalación y reiniciamos (USB, DVD, ISO, etc)**

pfSense 6

5.1.4. Configuraciones iniciales

1. **Configuramos VLANs**

pfSense primer boot 1

2. **WAN Interface**

   Elegimos vtnet0

pfSense primer boot 2

3. **LAN Interface**

   Escribimos vtnet1

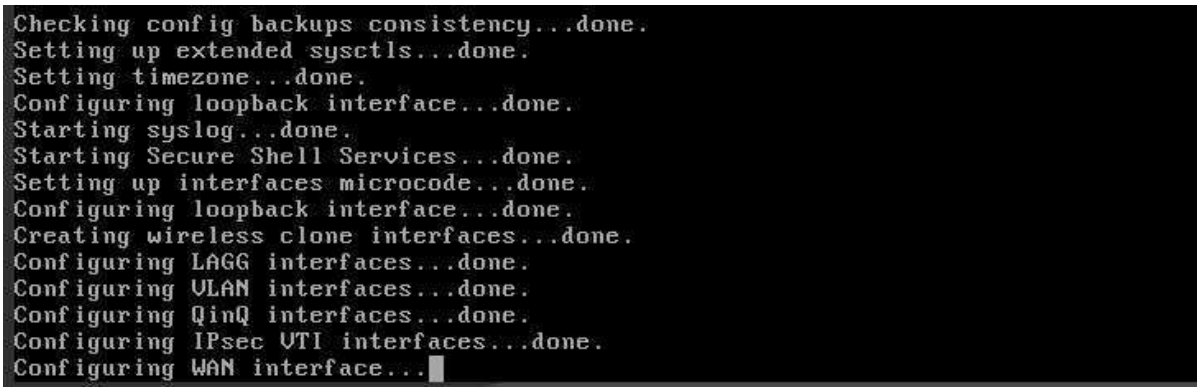
pfSense primer boot 3

4. **Pregunta si estamos seguros de nuestra configuración**

   Escribimos "y"

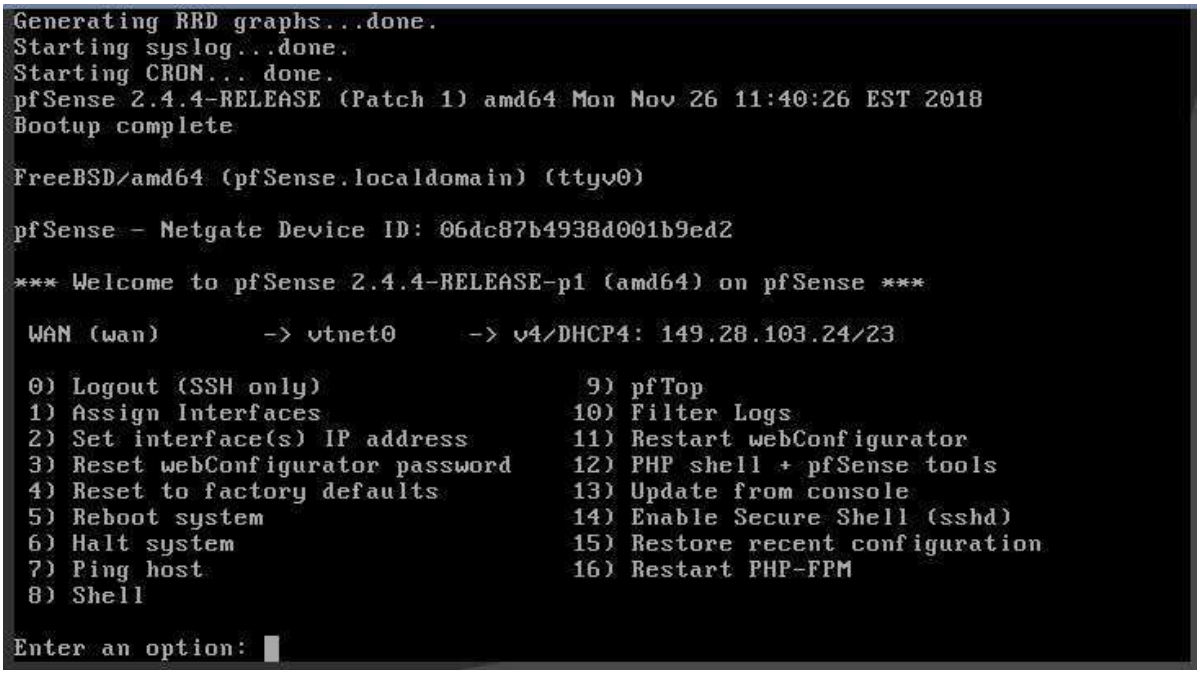
pfSense primer boot 4

5. **Esperamos a que la configuración se realice completamente**



pfSense primer boot 5

6. **Fin de configuraciones iniciales**

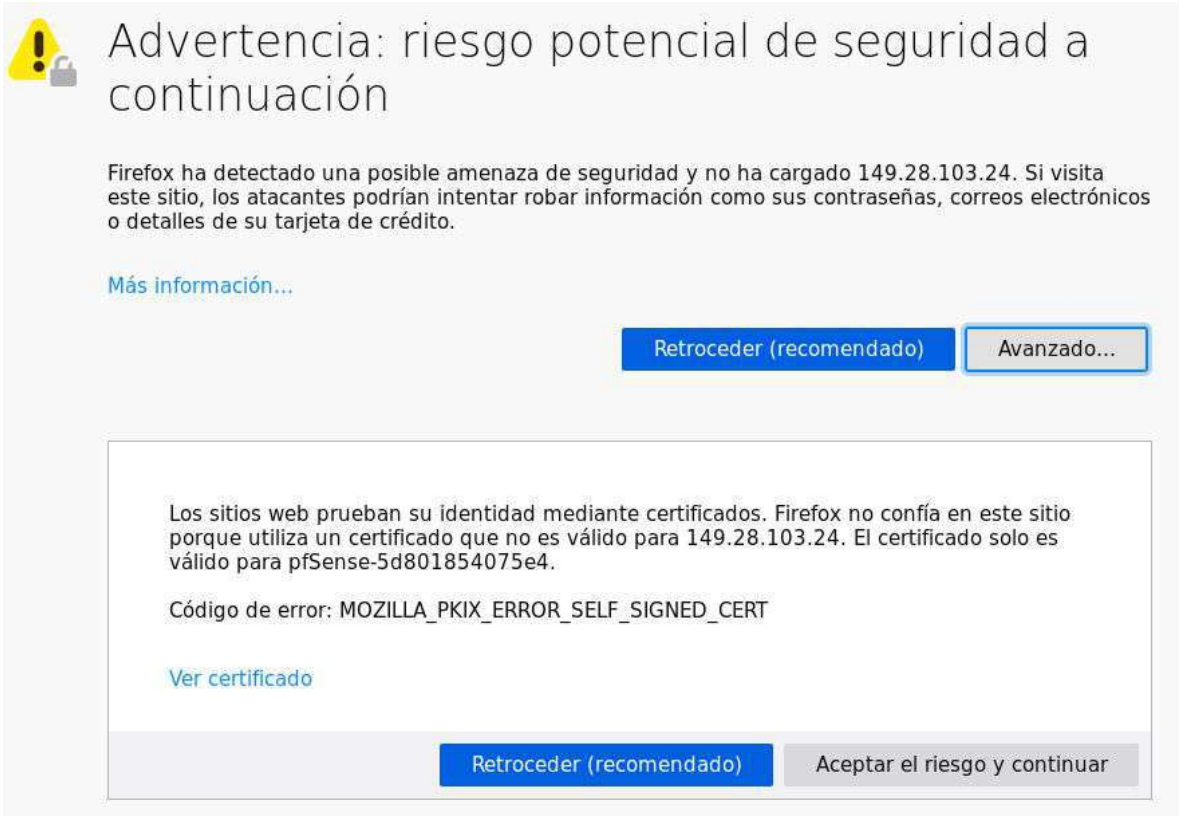


pfSense primer boot 6

5.1.5. Configuraciones desde la interfaz web

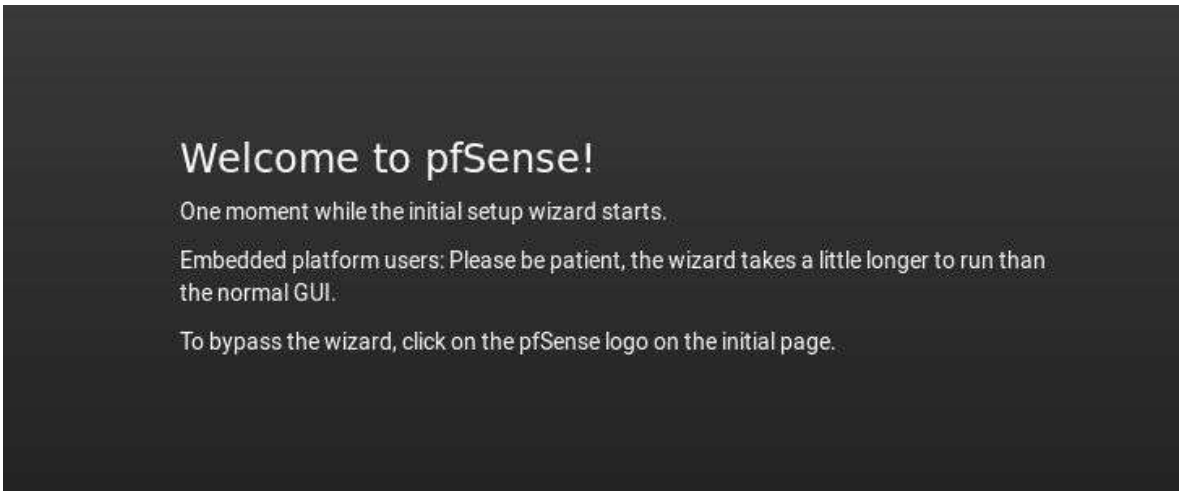
1. **La primera vez ingresamos por la IP del servidor**

   No hemos configurado ningún certificado SSL, por esa razón los navegadores nos advierten que nuestra conexión no es segura



pfSense interfaz web 1

2. **Esperamos a que se realicen las configuraciones iniciales**



pfSense interfaz web 2

3. **Entramos con el usuario y password por defecto**

   Usuario: admin

   Password: pfsense

pfSense interfaz web 3

4. **Cambiamos contraseña**

pfSense interfaz web 4

5. **Cambiamos idioma**

   System > General configuration > Save

pfSense interfaz web 5

6. **Desactivamos DNS Rebinding checks**

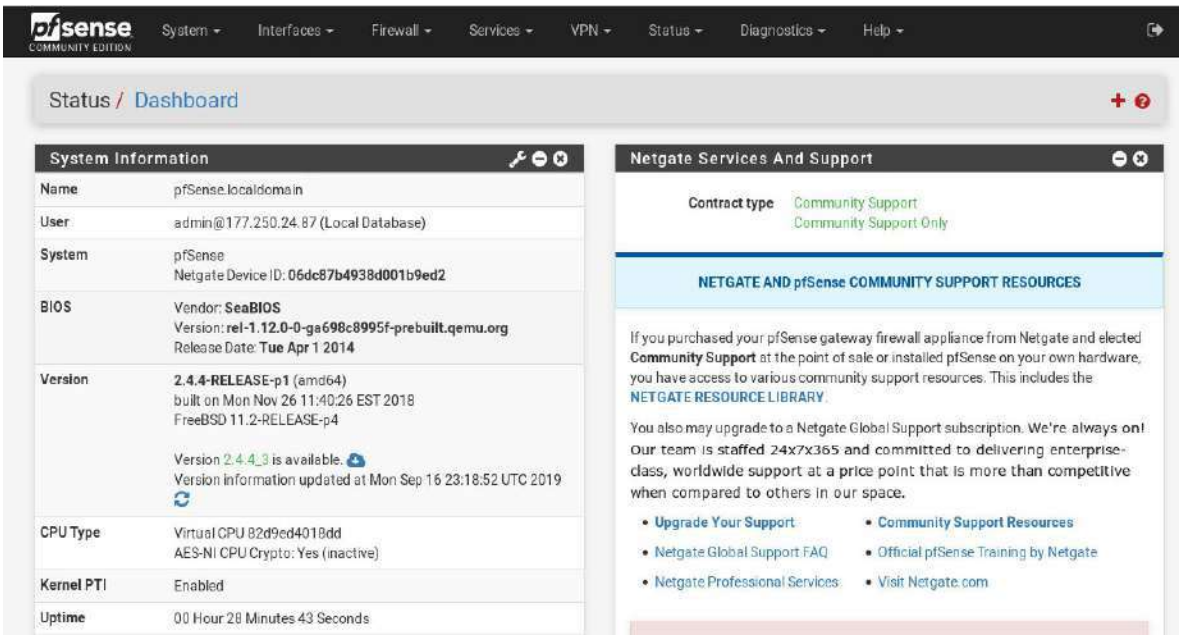
pfSense interfaz web 6

7. **Desactivamos HTTP_REFERER**

pfSense interfaz web 7

8. **Ingresar desde dominio**

   Una vez realizados los pasos anteriores podemos ingresar mediante el dominio que elegimos. Nuestro caso **usemoslinux.net**, pero elegimos usar un subdominio el cual es **pfsense.usemoslinux.net**



pfSense interfaz web 8

Usar Let's Encrypt en pfSense

1. **Instalamos el paquete ACME**
   Sistema > Gerente de empaquetación > Paquetes disponibles > Install

Let’s Encrypt 1

5.2. Instalación de HAProxy

5.2.1. Requerimientos del sistema

"Operating System and Hardware Requirements | HAProxy Enterprise 1.5r2" (s.f.) define los siguientes requerimientos del sistema para el correcto funcionamiento de HAProxy

Bajo Nivel

- Tráfico TCP o HTTP
- Hasta 1000 conex/s
- Muy poco tráfico SSL o compresión gzip

Este tipo de carga de trabajo puede lograrse mediante una máquina virtual o un servidor básico. Necesitas al menos:

- 1 Núcleo de CPU
- 1 GB de RAM

Medio Nivel

- Tráfico TCP o HTTP (incluida la manipulación HTTP)
- Hasta 4000 conex/s
- Bajo tráfico SSL o compresión gzip

Este tipo de carga de trabajo puede lograrse mediante una máquina virtual o un servidor básico. Necesitas al menos:

- 2 Núcleos de CPU
- 1 GB de RAM

Alto Nivel

- Tráfico TCP o HTTP (incluida la manipulación HTTP)
- Hasta 20000 conex/s
- 10% del tráfico cifrado (SSL) o comprimido

Este tipo de carga de trabajo solo se puede lograr con un servidor de bare metal. Necesitas al menos:

- 2 núcleos de CPU, lo más rápido posible
- 4 GB de RAM
- Potente tarjeta de red

5.2.2. Establecer IP estática a servidor HAProxy

Para configurar una dirección IP estática usando la nueva herramienta NetPlan en el servidor Ubuntu 18.04.

NODO 1: modificaremos el archivo **/etc/netplan/01-netcfg.yaml** (si no existe lo creamos), copiamos y pegamos el siguiente texto

```
#------------------------------------------------------------------------------------------------------------
----
network:
  ethernets:
    ens7:
      addresses: [192.168.1.2/24]
      gateway4: 192.168.1.1
      nameservers:
```

```
            addresses: [8.8.8.8,8.8.4.4]
          dhcp4: no
      version: 2
#-------------------------------------------------------------------------------------------------------
----
```

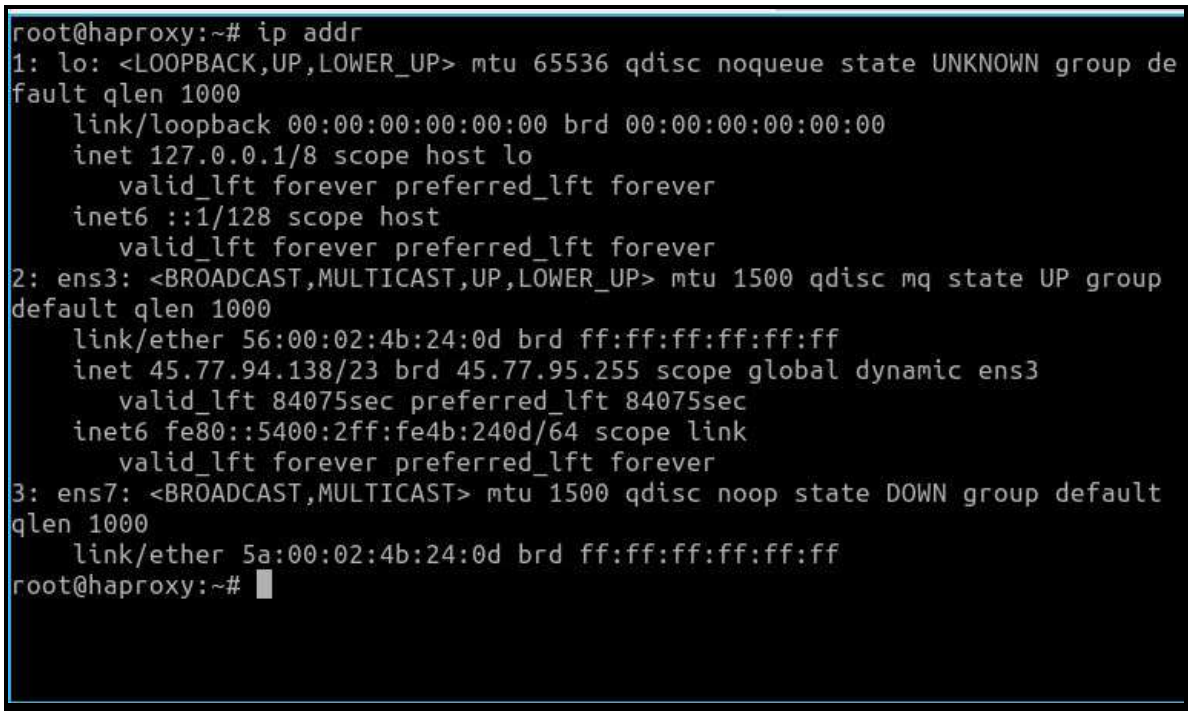
NODO 2: modificaremos el archivo **/etc/netplan/01-netcfg.yaml** (si no existe lo creamos), copiamos y pegamos el siguiente texto

```
#-------------------------------------------------------------------------------------------------------
----
    network:
      ethernets:
        ens7:
          addresses: [192.168.1.3/24]
          gateway4: 192.168.1.1
          nameservers:
            addresses: [8.8.8.8,8.8.4.4]
          dhcp4: no
      version: 2
#-------------------------------------------------------------------------------------------------------
----
```

Todo lo marcado en rojo debe ser modificado

1. **ens7:** Se refiere a la interfaz de red. Para ver las interfaces de red disponibles ejecutar el siguiente comando:

   # ip addr



HaProxy 1

   Ahora vemos que el número 3 está sin IP, usaremos esta interfaz

2. **192.168.1.x:** La IP que queremos para nuestro servidor
3. **IP pfSense:** Gateway, salida a Internet, en nuestro caso el servidor pfSense
4. **8.8.8.8, 8.8.4.4:** Servidores DNS, utilizaremos la IP de pfSense

   Por último para aplicar el nuevo archivo ejecutamos:

   # netplan apply

   Para ver los cambios

   # ip addr

5.2.3. Establecer IP Virtual para los dos nodos HAProxy

La IP Virtual de HAProxy será **192.168.1.200**

Nodo 1 y Nodo 2:

Ejecutar el siguiente comando:

```
# apt install keepalived -y
```

Nodo 1:

nano /etc/keepalived/keepalived.conf

La IP Virtual será **192.168.1.200**, la prioridad para el **master** será **200**

```
vrrp_instance VI_1 {
    state MASTER
    interface ens7
    virtual_router_id 51
    priority 200
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass password
    }
    virtual_ipaddress {
        192.168.1.200
    }
}
```

Nodo 2:

nano /etc/keepalived/keepalived.conf

La IP Virtual será **192.168.1.200**, la prioridad para el **backup** será **100**

```
vrrp_instance VI_1 {
    state BACKUP
    interface ens7
    virtual_router_id 51
    priority 100
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass password
    }
    virtual_ipaddress {
        192.168.1.200
    }
}
```

Nodo 1 y Nodo 2:

Una vez que ambos servicios Keepalived están configurados, inicia cada servicio y actívalo en el arranque.

```
# systemctl start keepalived
# systemctl enable keepalived
```

5.2.4. Instalación de HAProxy en Nodo 1 y Nodo 2

Bastará con este simple comando

```
# apt install haproxy -y
```

Agregaremos la dirección **usemoslinux.net** como primer sitio de prueba para el clúster apache. Abrimos ***/etc/haproxy/haproxy.cfg*** y agregamos al final del archivo lo siguiente:

5.2.5. Instalamos certbot en el Nodo 1 y Nodo 2 para poder utilizar Let's Encrypt

```
# add-apt-repository -y ppa:certbot/certbot
```

```
# apt-get update
```

```
# apt-get install -y certbot
```

Problemas para Let's Encrypt y HAProxy

El primer obstáculo para sortear surge porque Let's Encrypt autoriza un certificado para un servidor al solicitar un archivo a través de una solicitud HTTP (S). Sin embargo, HAProxy no es un servidor web. No servirá archivos por sí mismo, solo redirigirá una solicitud a otra ubicación. Nuestros servidores de aplicaciones no podrán manejar esta solicitud de autorización.

Dado que queremos nuestro certificado SSL en el equilibrador de carga (terminación SSL), nuestro objetivo es encontrar una manera de que HAProxy reconozca una solicitud de LetsEncrypt y la enrute a un servicio web que responderá con la respuesta que LetsEncrypt necesita para autorizar el certificado.

LetsEncrypt viene con su propio oyente de servidor web incorporado para tal caso de uso, así podemos lograr esto.

El segundo obstáculo es que HAProxy espera que un certificado SSL esté en un archivo que incluya la cadena de certificados, el certificado raíz y la clave privada. HAProxy tiene la clave privada en un archivo separado, por lo que nuestro último paso es combinar los archivos en algo que HAProxy pueda leer.

5.2.6. Archivo de configuración HAProxy Nodo 1 y Nodo 2

Ahora necesitamos editar el archivo ***/etc/haproxy/haproxy.cfg*** y agregar lo siguiente al final:

```
##########################
frontend lets-encrypt
        bind *:80
        acl letsencrypt-acl path_beg /.well-known/acme-challenge/
        use_backend letsencrypt-backend if letsencrypt-acl
backend letsencrypt-backend
   server letsencrypt 127.0.0.1:8888
##########################
##########################
frontend usemoslinux.net
        bind *:80
        mode http
        acl USEMOSLINUX hdr_dom(host) -i usemoslinux.net
        use_backend usemoslinux if USEMOSLINUX
backend usemoslinux
```

mode http

balance leastconn

cookie SERVERUSED insert indirect nocache

option httpchk HEAD /

server server1 192.168.1.11:80 cookie server1

server server2 192.168.1.12:80 cookie server2

##########################

El primer bloque corresponde a Let's Encrypt, indispensable para crear certificados ssl. Luego viene la configuración del sitio web de prueba. Todo lo marcado en rojo es importante, en especial usemoslinux.net que es el sitio web que vamos a mostrar y 192.168.1.{11:12} que es la IP privada del VPS a la que apunta.

En 2019, "The Four Essential Sections of an HAProxy Configuration" establece los siguientes parámetros:

balance: Los valores comunes de equilibrio de carga incluyen roundrobin, que solo elige el siguiente servidor y comienza de nuevo en la parte superior de la lista, y leastconn, donde HAProxy selecciona el servidor con la menor cantidad de sesiones activas.

cookie: La configuración de cookies permite la persistencia basada en cookies. Le dice a HAProxy que envíe una cookie llamada SERVERUSED al cliente y que la asocie con el nombre del servidor que dio la respuesta inicial. Esto hace que el cliente continúe hablando con ese servidor durante la duración de su sesión. Tenga en cuenta que el nombre del servidor se establece con un argumento de cookie en la línea del servidor.

option httpchk: La configuración de la opción httpchk hace que HAProxy envíe verificaciones de estado de Capa 7 (HTTP) en lugar de verificaciones de Capa 4 (TCP) a sus servidores de back-end. Los servidores que no responden no reciben más solicitudes. Las comprobaciones de TCP tienen éxito si son capaces de establecer una conexión con la IP y el puerto del servidor de fondo, las comprobaciones de estado de HTTP esperan volver a una respuesta HTTP exitosa. Las comprobaciones de estado más inteligentes son fundamentales para eliminar los servidores que no responden, incluso si no responde significa simplemente obtener una mala respuesta HTTP como 500 Server Error. De forma predeterminada, una comprobación de estado de HTTP realiza una solicitud a la ruta raíz **"/"**.

5.2.7. Configurar DNS del dominio

No podemos crear ningún certificado si el dominio no está debidamente configurado, La configuración DNS para que el dominio **usemoslinux.net:**

Type	Name	Data	TTL (seconds)	Priority	Actions
A	Example: www	Example: 0.0.0.0	3600		+
A		45.77.80.188	300		
CNAME	*	usemoslinux.net	300		
MX		usemoslinux.net	300	10	
NS		ns1.vultr.com	300		
NS		ns2.vultr.com	300		

DNS usemoslinux.net

Nota: La entra "A" debe apuntar a la IP pública y estática del **Frontend**

5.2.8. Crear Nodo 1 y Nodo 2 Apache web server

Antes de probar la configuración, obviamente debemos crear los nodos que recibirán las peticiones:

Nodo 1: modificaremos el archivo **/etc/netplan/01-netcfg.yaml** (si no existe lo creamos), copiamos y pegamos el siguiente texto

```
#------------------------------------------------------------------------------------------------------------
----
    network:
      ethernets:
        ens7:
          addresses: [192.168.1.11/24]
          gateway4: 192.168.1.1
          nameservers:
            addresses: [8.8.8.8,8.8.4.4]
          dhcp4: no
      version: 2
#------------------------------------------------------------------------------------------------------------
----
```

Instalamos Apache

```
# apt install apache2 -y
```

Nodo 2: modificaremos el archivo **/etc/netplan/01-netcfg.yaml** (si no existe lo creamos), copiamos y pegamos el siguiente texto

```
#-------------------------------------------------------------------------------------------------------------
----
        network:
          ethernets:
            ens7:
              addresses: [192.168.1.12/24]
              gateway4: 192.168.1.1
              nameservers:
                addresses: [8.8.8.8,8.8.4.4]
              dhcp4: no
          version: 2
#-------------------------------------------------------------------------------------------------------------
----
```

Instalamos Apache

```
# apt install apache2 -y
```

5.2.9. Creamos un nuevo certificado

Con la configuración hecha anteriormente, podemos ejecutar el comando de creación de nuevo certificado

```
# certbot certonly --standalone -d usemoslinux.net --http-01-port=8888
```

Se ha creado dos archivo **fullchain.pem** y **privkey.pem** en el directorio **/etc/letsencrypt/live/usemoslinux.net/**, debemos colocar ambos archivos en uno solo para hacerlo funcionar.

```
# cd /etc/letsencrypt/live/usemoslinux.net/ &&  touch usemoslinux.pem

# cat fullchain.pem privkey.pem > usemoslinux.net
```

Ahora colocamos estas dos líneas para hacer funcionar **https** en el archivo de configuración de haproxy **/etc/haproxy/haproxy.cfg**

```
bind *:443 ssl crt /etc/letsencrypt/live/usemoslinux.net/usemoslinux.pem

http-request redirect scheme https unless { ssl_fc }
```

La configuración final debería quedar como sigue:

```
##########################
frontend lets-encrypt
   bind *:80
   # Esta URL para ver si es una solicitud de letsencrypt
   acl letsencrypt-acl path_beg /.well-known/acme-challenge/
   use_backend letsencrypt-backend if letsencrypt-acl
# LE Backend
backend letsencrypt-backend
   server letsencrypt 127.0.0.1:8888
##########################
##########################
frontend usemoslinux.net
        bind *:80
        mode http
        bind *:443 ssl crt /etc/letsencrypt/live/usemoslinux.net/usemoslinux.pem
```

```
        http-request redirect scheme https unless { ssl_fc }
        acl USEMOSLINUX hdr_dom(host) -i usemoslinux.net
        use_backend usemoslinux if USEMOSLINUX
backend usemoslinux
        mode http
        balance leastconn
        cookie SERVERUSED insert indirect nocache
        option httpchk HEAD /
        server server1 192.168.1.11:80 cookie server1
        server server2 192.168.1.12:80 cookie server2
##########################
```

Ahora reiniciamos haproxy con:

```
# service haproxy restart
```

Ingresando a la dirección **usemoslinux.net** debería estar funcionando correctamente, si un nodo de apache cae, el otro debería seguir funcionando y los usuarios finales no se darían cuenta.

5.2.10. Sincronizar Nodo 1 y Nodo 2

La comunicación entre ambos nodos debe ser sin contraseñas, es decir, con un ssh key.

En ambos nodos

```
# ssh-keygen
```

Dejamos los valores por defecto

Nodo 1 > Nodo 2

Copiamos el key generado al Nodo 2

```
# ssh-copy-id -i ~/.ssh/id_rsa root@192.168.1.3
```

Nodo 2 > Nodo 1

Copiamos el key generado al Nodo 1

```
# ssh-copy-id -i ~/.ssh/id_rsa root@192.168.1.2
```

La sincronización de los archivos de configuración en ambos nodos no puede ser automática, por la sencilla razón de que el parámetro **cookie** de HAProxy dejaría de tener sentido. Crearemos un archivo llamado **sync-haproxy.sh** en ambos nodos y añadimos lo siguiente:

```
####################
#!/bin/bash

rsync -avzhie "ssh -p 22" root@192.168.1.X:/etc/haproxy/haproxy.cfg \
/etc/haproxy/haproxy.cfg

rsync -avzhie "ssh -p 22" root@192.168.1.X:/etc/letsencrypt/  /etc/letsencrypt/

service haproxy reload
####################
```

Nota: X toma el valor 3 si se trata del Nodo 1 y el valor 2 si se trata del Nodo 2

Cada vez que añadimos un nuevo dominio a HAProxy debemos ejecutar este archivo, por ejemplo: Si modificamos el archivo **haproxy.cfg** en el Nodo 1, debemos ejecutar el archivo **sync-haproxy.sh** en el Nodo 2

```
# sh sync-haproxy.sh
```

5.3. Instalar NFS

NFS se utilizará para Apache2, Tomcat y Nextcloud. Es recomendable hacer un par de nodos NFS para cada servicio, pero por cuestiones prácticas y de costo se utilizará el mismo par de nodos para estos tres servicios.

5.3.1. Pre-requisitos

- Nodo 1: IP estática 192.168.1.51
- Nodo 2: IP estática 192.168.1.52
- IP Virtual: 192.168.1.50

5.3.2. Instalación de NFS

```
# apt install nfs-kernel-server
# mkdir /var/nfs/html -p
# chown nobody:nogroup /var/nfs/html
```

5.3.3. Exportar los directorios

```
# nano /etc/exports
```

```
/var/nfs/html    192.168.1.11(rw,sync,no_subtree_check)
/var/nfs/html    192.168.1.12(rw,sync,no_subtree_check)
```

5.3.4. Montar directorio en los clientes

```
# mount 192.168.1.50:/var/nfs/html /var/www/html
```

5.3.5. Montar automáticamente en cada reinicio

```
# nano /etc/fstab
```

Debemos agregar lo siguiente al archivo

192.168.1.50:/var/nfs/html /nfs/html nfs auto,nofail,noatime,nolock,intr,tcp,actimeo=1800 0 0

5.3.6. Crear SSH Key en ambos nodos

El protocolo SSH utiliza criptografía de clave pública para autenticar hosts y usuarios. Las claves de autenticación, llamadas claves SSH , se crean utilizando el programa keygen. ("How to use ssh-keygen to generate a new SSH key | SSH.COM", s.f.)

Ejecutaremos el siguiente comando sin argumentos y dejaremos todo por defecto.

```
# ssh-keygen
```

Copiamos la clave rsa generada en el nodo 1 al nodo 2

```
# ssh-copy-id root@192.168.1.52
```

Copiamos la clave rsa generada en el nodo 2 al nodo 1

```
# ssh-copy-id root@192.168.1.51
```

5.3.7. Sincronizar Nodo 2 con Nodo 1

Sincronizamos ambos nodos mediante un archivo llamado **sync-html.sh** que contiene lo siguiente:

```
-------------------------------------------------------------------------------------------------------------
----
#!/bin/bash
```

rsync -avzhie "ssh -p 22" root@192.168.1.51:/var/nfs/html/ \

/var/nfs/html/ --delete

5.3.8. Crear una cron job Nodo 2

El Nodo 2 es solo de respaldo. Lo ideal es que el Nodo 1 sea el único que siempre esté trabajando. En caso de que el Nodo 1 falle, automáticamente el Nodo 2 toma la IP Virtual **192.168.1.50** y hace el trabajo del Nodo 1, pero obviamente debe tener los mismos archivos del Nodo 1.

Necesitamos que el archivo anterior se ejecute cada 1 minuto por lo menos para mantener sincronizados los dos nodos lo más que se pueda.

```
# crontab -e
```

Añadimos al final:

*/1 * * * * sh ~/sync-html.sh

5.3.9. Nodo 1 deja de funcionar

El Nodo 2 que está sincronizado toma su lugar, sin embargo, se deben seguir los siguientes pasos para poner en marcha al Nodo 1 nuevamente:

Comentar el cron job del Nodo 2 para impedir que se siga sincronizando con el Nodo 1

Reiniciar Nodo 1

Ejecutar el siguiente comando (Se puede guardar como script):

```
rsync -avzhie "ssh -p 22" root@192.168.1.52:/var/nfs/html/ \
/var/nfs/html/ --delete
```

Una vez sincronizado Nodo 1 con Nodo 2, volver a iniciar KeepAlived.

5.4. Instalar Clúster Apache2

Previamente instalamos apache2 en dos nodos y modificamos sus respectivas IP estáticas privadas, con el objetivo de probar HAProxy. Ahora nos ocuparemos de montar el directorio exportado de NFS en nuestro directorio html. Esto hará que cada cambio que se haga en el Nodo 1, se vea reflejado instantáneamente en el Nodo 2 y viceversa.

5.4.1. Pre-requisitos

- Nodo 1: IP estática 192.168.1.11
- Nodo 2: IP estática 192.168.1.12
- Tener instalado nfs-common
- Montar el directorio /var/nfs/html en /var/www/html para ambos nodos
- Apache2 instalado

5.4.2. Probar NFS y Crear Virtualhost

Probamos crear directorios, archivos, subir imágenes, etc en ambos nodos y verificamos que cualquier cambio se vea reflejado instantáneamente en el otro nodo.

Para crear el Virtualhost, lo que debemos hacer es crear un archivo dominio.conf en el directorio /etc/apache2/sites-available con el siguiente contenido:

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    ServerName your_domain
    ServerAlias www.your_domain
    DocumentRoot /var/www/html/your_domain
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

5.5. Instalar Clúster Apache Tomcat

Ahora instalamos Apache Tomcat en ambos nodos

5.5.1. Pre-requisitos

- Nodo 1: IP estática 192.168.1.71
- Nodo 2: IP estática 192.168.1.72
- Tener instalado nfs-common
- Previamente exportar el directorio /var/nfs/tomcat en los nodos NFS
- Montar el directorio /var/nfs/tomcat en /opt/tomcat para ambos nodos

5.5.2. Instalar Java

```
# apt update
```

```
# apt install default-jdk
```

5.5.3. Creamos el grupo y el usuario tomcat

```
# groupadd tomcat
```

```
# useradd -s /bin/false -g tomcat -d /opt/tomcat tomcat
```

5.5.4. Descargamos tomcat en alguno de los dos nodos

```
# cd /tmp
```

```
# curl -O
http://mirror.cc.columbia.edu/pub/software/apache/tomcat/tomcat-9/v9.0.10/bin/apache-tomcat-9.0.10.tar.gz
```

```
# tar xzvf apache-tomcat-9*tar.gz -C /opt/tomcat --strip-components=1
```

5.5.5. Actualizamos los permisos en ambos nodos

```
# cd /opt/tomcat
```

```
# chgrp -R tomcat /opt/tomcat
```

```
# chmod -R g+r conf
```

```
# chmod g+x conf
```

```
# chown -R tomcat webapps/ work/ temp/ logs/
```

5.5.6. Crear un archivo de servicio systemd

```
# nano /etc/systemd/system/tomcat.service
```

Agregar lo siguiente:

```
[Unit]
Description=Apache Tomcat Web Application Container
After=network.target

[Service]
Type=forking

Environment=JAVA_HOME=/usr/lib/jvm/java-1.11.0-openjdk-amd64
Environment=CATALINA_PID=/opt/tomcat/temp/tomcat.pid
Environment=CATALINA_HOME=/opt/tomcat
Environment=CATALINA_BASE=/opt/tomcat
Environment='CATALINA_OPTS=-Xms512M -Xmx1024M -server
-XX:+UseParallelGC'
Environment='JAVA_OPTS=-Djava.awt.headless=true
-Djava.security.egd=file:/dev/./urandom'

ExecStart=/opt/tomcat/bin/startup.sh
ExecStop=/opt/tomcat/bin/shutdown.sh
```

User=tomcat

Group=tomcat

UMask=0007

RestartSec=10

Restart=always

[Install]

WantedBy=multi-user.target

---

systemctl daemon-reload

Verificar si funciona el nuevo servicio

systemctl start tomcat

systemctl status tomcat

5.5.7. Configurar Tomcat administrador web en un nodo

Debemos agregar un usuario administrador

nano /opt/tomcat/conf/tomcat-users.xml

```
<tomcat-users . . .>
    <user username="admin" password="password" roles="manager-gui,admin-gui"/>
</tomcat-users>
```

Ahora debemos permitir el acceso remoto:

En ambos archivos comentar

```
# nano /opt/tomcat/webapps/manager/META-INF/context.xml

# nano /opt/tomcat/webapps/host-manager/META-INF/context.xml

<Context antiResourceLocking="false" privileged="true" >
  <!--<Valve className="org.apache.catalina.valves.RemoteAddrValve"
         allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />-->
</Context>
```

Después de configurar HAProxy, deberíamos ver lo siguiente:

Apache Tomcat

5.5.8. Probar NFS y Crear Virtualhost

Probamos crear directorios, archivos, subir imágenes, etc en ambos nodos y verificamos que cualquier cambio se vea reflejado instantáneamente en el otro nodo.

Para crear el Virtualhost, lo que debemos hacer es crear editar el archivo server.xml en el directorio /opt/tomcat/config/ con el siguiente contenido:

```
<Host name="example.com"  appBase="webapps" unpackWARs="true"
autoDeploy="true">
    <Alias>www.example.com</Alias>

    <Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"
        prefix="example_access_log" suffix=".txt"
        pattern="%h %l %u %t %r %s %b" />

    <Context path="" docBase="/opt/tomcat/webapps/myapp1"
            debug="0" reloadable="true"/>
</Host>
```

5.6. Instalar Clúster MariaDB

El clúster agrega alta disponibilidad a la base de datos mediante la distribución de cambios a diferentes servidores. En el caso de que una de las instancias falle, otra estará rápidamente disponible para continuar sirviendo.

Los clústeres vienen en dos configuraciones generales, activo-pasivo y activo-activo. En los clústeres activo-pasivos, todas las escrituras se realizan en un único servidor activo y luego se copian en uno o más servidores pasivos que están listos para asumir el control solo en caso de una falla del servidor activo. Algunos clústeres activo-pasivos también permiten operaciones SELECT en nodos pasivos.

En un clúster activo-activo, cada nodo es de lectura-escritura y un cambio realizado en uno se replica a todos.

5.6.1. Pre-requisitos

- Nodo 1: IP estática 192.168.1.21
- Nodo 2: IP estática 192.168.1.22
- IP Virtual: 192.168.1.150
- Un usuario no root con privilegios de administrador

5.6.2. Agregar los repositorios MariaDB a cada nodo

En este paso, agregamos los repositorios de paquetes de MariaDB relevantes a cada uno de los dos nodos para que pueda instalar la versión correcta de MariaDB utilizada en este clúster. Una vez que los repositorios se actualicen en los dos nodos, instalamos MariaDB.

Primero, agregamos la clave del repositorio MariaDB con el comando apt-key, que el administrador de paquetes APT usará para verificar que el paquete sea auténtico:

```
$ sudo apt-key adv --recv-keys --keyserver hkp://keyserver.ubuntu.com:80
0xF1656F24C74CD1D8
```

Una vez que tengamos la clave de confianza en la base de datos, podemos agregar el repositorio con el siguiente comando:

```
$  sudo add-apt-repository 'deb [arch=amd64]
http://nyc2.mirrors.digitalocean.com/mariadb/repo/10.4/ubuntu bionic main'
```

Después de agregar el repositorio, ejecutamos apt update para incluir manifiestos de paquete del nuevo repositorio:

```
$ sudo apt update
```

5.6.3. Instalar MariaDB en cada nodo

En este paso, instalaremos los paquetes reales de MariaDB en los dos nodos.

A partir de la versión 10.1, los paquetes MariaDB Server y MariaDB Galera Server se combinan, por lo que la instalación de mariadb-server instalará automáticamente Galera y varias dependencias:

```
$ sudo apt install mariadb-server -y
```

Desde MariaDB versión 10.4 en adelante, el usuario root de MariaDB no tiene una contraseña por defecto. Para establecer una contraseña para el usuario root, comenzamos iniciando sesión en MariaDB:

```
$ sudo mysql -uroot
set password = password("contraseña");
```

5.6.4. Configuramos el primer Nodo

En este paso configuraremos el primer nodo. Cada nodo en el clúster debe tener una configuración casi idéntica. Debido a esto, haremos toda la configuración en la primera máquina y luego lo copiaremos a los otros nodos.

De manera predeterminada, MariaDB está configurado para verificar el directorio /etc/mysql/conf.d para obtener configuraciones adicionales de los archivos que terminan en .cnf. Creamos un archivo en este directorio con todas sus directivas específicas de clúster:

```
sudo nano /etc/mysql/conf.d/galera.cnf
[mysqld]
binlog_format=ROW
```

```
default-storage-engine=innodb
innodb_autoinc_lock_mode=2
bind-address=0.0.0.0
# Galera Provider Configuration
wsrep_on=ON
wsrep_provider=/usr/lib/galera/libgalera_smm.so
# Galera Cluster Configuration
wsrep_cluster_name="test_cluster"
wsrep_cluster_address="gcomm://192.168.1.21,192.168.1.22"

# Galera Synchronization Configuration
wsrep_sst_method=rsync
# Galera Node Configuration
wsrep_node_address="192.168.1.21"
wsrep_node_name="mariadb-01"
```

La primera sección modifica o reafirma la configuración de MariaDB / MySQL que permitirá que el clúster funcione correctamente. Por ejemplo, Galera no funcionará con MyISAM o motores de almacenamiento no transaccionales similares, y mysqld no debe estar vinculado a la dirección IP de localhost.

La sección "Galera Provider Configuration" configura los componentes de MariaDB que proporcionan una API de replicación WriteSet. Esto significa Galera en nuestro

caso, ya que Galera es un proveedor de wsrep (WriteSet Replication). Especificamos los parámetros generales para configurar el entorno de replicación inicial.

La sección "Galera Cluster Configuration" define el clúster, identifica los miembros del clúster por dirección IP o nombre de dominio resoluble y crea un nombre para el clúster para garantizar que los miembros se unan al grupo correcto. Podemos cambiar wsrep_cluster_name por algo más significativo que test_cluster o dejarlo como está, pero debemos actualizar wsrep_cluster_address con las direcciones IP privadas de nuestros nodos.

La sección "Galera Synchronization Configuration" define cómo el clúster se comunicará y sincronizará los datos entre los miembros. Esto se usa solo para la transferencia de estado que ocurre cuando un nodo se conecta. Para la configuración inicial, estamos utilizando rsync, porque está comúnmente disponible.

La sección "Galera Node Configuration" aclara la dirección IP y el nombre del servidor actual. Esto es útil cuando se intenta diagnosticar problemas en los registros y para hacer referencia a cada servidor de varias maneras. Wsrep_node_address debe coincidir con la dirección de la máquina en la que se encuentra, pero podemos elegir el nombre que deseemos para identificar el nodo en los archivos de registro.

5.6.5. Configuramos el segundo Nodo.

```
# sudo nano /etc/mysql/conf.d/galera.cnf
```

Modificamos la siguiente sección:

```
. . .
# Galera Node Configuration
wsrep_node_address="192.168.1.22"
```

wsrep_node_name="mariadb-02"

. . .

5.6.6. Iniciamos el Clúster

Detener MariaDB en los Nodos.

```
$ sudo systemctl stop mysql
```

Iniciamos el primer nodo

```
$ sudo galera_new_cluster
```

Verificamos el cluster en el primer nodo:

```
mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_cluster_size'"
```

La salida debe ser exactamente a lo siguiente

```
+----------------------+-------+
| Variable_name        | Value |
+----------------------+-------+
| wsrep_cluster_size |    1     |
+----------------------+-------+
```

Iniciamos el segundo nodo

```
$ sudo systemctl start mysql
```

No se mostrará ningún resultado en la ejecución exitosa. Veremos que el tamaño del clúster aumenta a medida que cada nodo se conecta:

```
mysql -u root -p -e "SHOW STATUS LIKE 'wsrep_cluster_size'"
+----------------------+-------+
```

| Variable_name | Value |

+----------------------+-------+

| wsrep_cluster_size | 2 |

+----------------------+-------+

5.6.7. Permitir que otros nodos de la red accedan al clúster

Repetir el proceso en cada nodo

Editamos el archivo **my.cnf** y comentamos la línea bind-address

bind-address =

Reiniciamos el servicio en ambos nodos

```
$ sudo systemctl mariadb restart
```

Nos logueamos a MariaDB y creamos un usuario con acceso total que puede hacer peticiones desde cualquier host

```
mysql -u root -p
GRANT ALL PRIVILEGES ON *.* TO 'admin'@'%' IDENTIFIED BY 'tesis2019'
WITH GRANT OPTION;
```

Ahora que el clúster MariaDB está configurado procedemos con la instalación de PHPMyAdmin

5.6.8. Instalación de PHPMyAdmin en el nodo 1 y nodo 2

Instalamos Apache2

```
$ sudo apt update
```

```
$ sudo apt install apache2
```

Instalamos PHP

```
$ sudo apt install php libapache2-mod-php php-mysql
```

Instalamos PHPMyadmin

```
$ sudo apt install phpmyadmin php-mbstring php-gettext
```

Habilitamos mbstring

```
$ sudo phpenmod mbstring
```

Editamos la configuración de Apache para PHPMyAdmin y agregamos AllowOverride All

```
$ sudo nano /etc/apache2/conf-available/phpmyadmin.conf
```

```
<Directory /usr/share/phpmyadmin>
    Options FollowSymLinks
    DirectoryIndex index.php
    AllowOverride All
    . . .
```

```
$ sudo systemctl restart apache2
```

Ahora creamos un archivo .htaccess para PHPMyAdmin, con el objetivo de agregar más seguridad añadimos un usuario y contraseña de Apache2

```
$ sudo nano /usr/share/phpmyadmin/.htaccess
```

```
####################
AuthType Basic
AuthName "Restricted Files"
```

AuthUserFile /etc/phpmyadmin/.htpasswd

Require valid-user

###################

Ahora creamos el usuario habilitado para acceder a la interfaz de PHPMyAdmin

```
sudo htpasswd -c /etc/phpmyadmin/.htpasswd usuario
```

5.7. Instalar Clúster Postgres

5.7.1. Pre-requisitos

- Nodo 1: IP estática 192.168.1.31
- Nodo 2: IP estática 192.168.1.32
- IP Virtual 192.168.1.160

5.7.2. Instalamos dependencias para postgresql en los dos nodos:

```
sudo su
cd ~
sudo sh -c 'echo "deb-src http://apt.postgresql.org/pub/repos/apt/ $(lsb_release -cs)-pgdg main 9.4" > /etc/apt/sources.list.d/pgdg.list'
sudo apt-get install wget ca-certificates unzip
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
sudo apt-get update
sudo apt-get build-dep postgresql-9.4
```

5.7.3. Construimos Postgresql en los dos nodos:

```
wget https://github.com/2ndQuadrant/bdr/archive/bdr-pg/REL9_4_12-1.tar.gz

tar -xzvf REL9_4_12-1.tar.gz

cd ~/bdr-bdr-pg-REL9_4_12-1

./configure --prefix=/usr/lib/postgresql/9.4 --enable-debug --with-openssl

make -j4 -s install-world

cd ..
```

5.7.4. Creamos BDR para la agrupación en clúster en los dos nodos:

```
wget https://github.com/2ndQuadrant/bdr/archive/bdr-plugin/1.0.6.zip

unzip 1.0.6.zip

cd ~/bdr-bdr-plugin-1.0.6

PATH=/usr/lib/postgresql/9.4/bin:"$PATH" ./configure

make -j4 -s all

make -s install

cd ..
```

Creamos usuario de postgres en los dos nodos:

```
cd ~

useradd postgres

passwd postgres

mkdir -p /var/lib/postgresql

chown postgres:postgres /var/lib/postgresql
```

```
sudo usermod -d /var/lib/postgresql postgres
```

Inicialización de Postgres en los nodos:

```
su -l postgres

export PATH=/usr/lib/postgresql/9.4/bin:$PATH

mkdir ~/9.4-bdr

initdb -D ~/9.4-bdr -A trust
```

5.7.5. Editamos la configuración de postgres:

```
nano ~/9.4-bdr/postgresql.conf

listen_addresses = '*'

shared_preload_libraries = 'bdr'

wal_level = 'logical'

track_commit_timestamp = on

max_connections = 100

max_wal_senders = 10

max_replication_slots = 10

max_worker_processes = 10
```

Editamos el acceso postgres en los nodos:

```
nano ~/9.4-bdr/pg_hba.conf

##########

local   replication     postgres                        trust

host    replication     postgres        127.0.0.1/32    trust

host    replication     postgres        ::1/128         trust
```

host all all 0.0.0.0/0 password

host replication postgres 10.200.200.100/32 trust

host replication postgres 10.200.200.110/32 trust

host replication bdrsync 10.200.200.100/32 password

host replication bdrsync 10.200.200.110/32 password

##########

```
pg_ctl -l ~/log -D ~/9.4-bdr start
```

5.7.6. Creamos cuentas para sincronizar en los dos nodos:

```
psql -c "CREATE USER bdrsync superuser;"

psql -c "ALTER USER bdrsync WITH PASSWORD '12345#';"
```

Creamos cuentas ficticias y bases de datos para probar en los nodos:

```
createuser test_user

createdb -O test_user test_db

psql

alter user test_user with encrypted password 'test_pass';

grant all privileges on database test_db to test_user;

\c test_db

GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO test_user;

GRANT ALL PRIVILEGES ON ALL SEQUENCES IN SCHEMA public TO
test_user;

\q
```

5.7.7. Agregamos la extensión BDR a la base de datos en todos los servidores:

```
psql test_db -c 'CREATE EXTENSION btree_gist;'

psql test_db -c 'CREATE EXTENSION bdr;'
```

En el nodo DB 1:

```
psql

\c test_db

SELECT bdr.bdr_group_create(

    local_node_name := 'nodo1',

    node_external_dsn := 'host=10.200.200.100 user=bdrsync dbname=test_db
password=12345#' );
```

En el nodo DB 2:

```
psql

\c test_db

SELECT bdr.bdr_group_join(

    local_node_name := 'nodo2',

    node_external_dsn := 'host=10.200.200.110 user=bdrsync dbname=test_db
password=12345#',

    join_using_dsn := 'host=10.200.200.100 user=bdrsync dbname=test_db
password=12345#' );
```

Veremos conexiones de clúster:

```
select * from bdr.bdr_nodes;

select * from bdr.bdr_connections;
```

5.7.8. Instalación de PgAdmin4

Agregar repositorio PostgreSQL

```
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
```

```
sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt/ `lsb_release -cs`-pgdg main" >> /etc/apt/sources.list.d/pgdg.list'
```

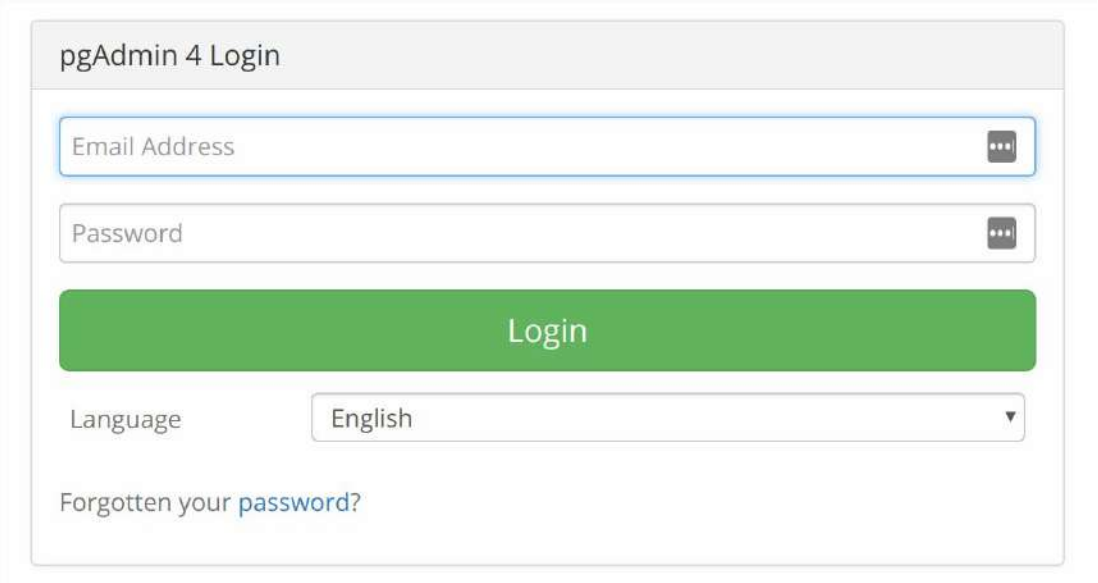
```
sudo apt update
```

```
sudo apt install pgadmin4 pgadmin4-apache2 -y
```

Durante la instalación, nos solicitará la configuración del nombre de usuario del correo electrónico y la configuración de la contraseña.

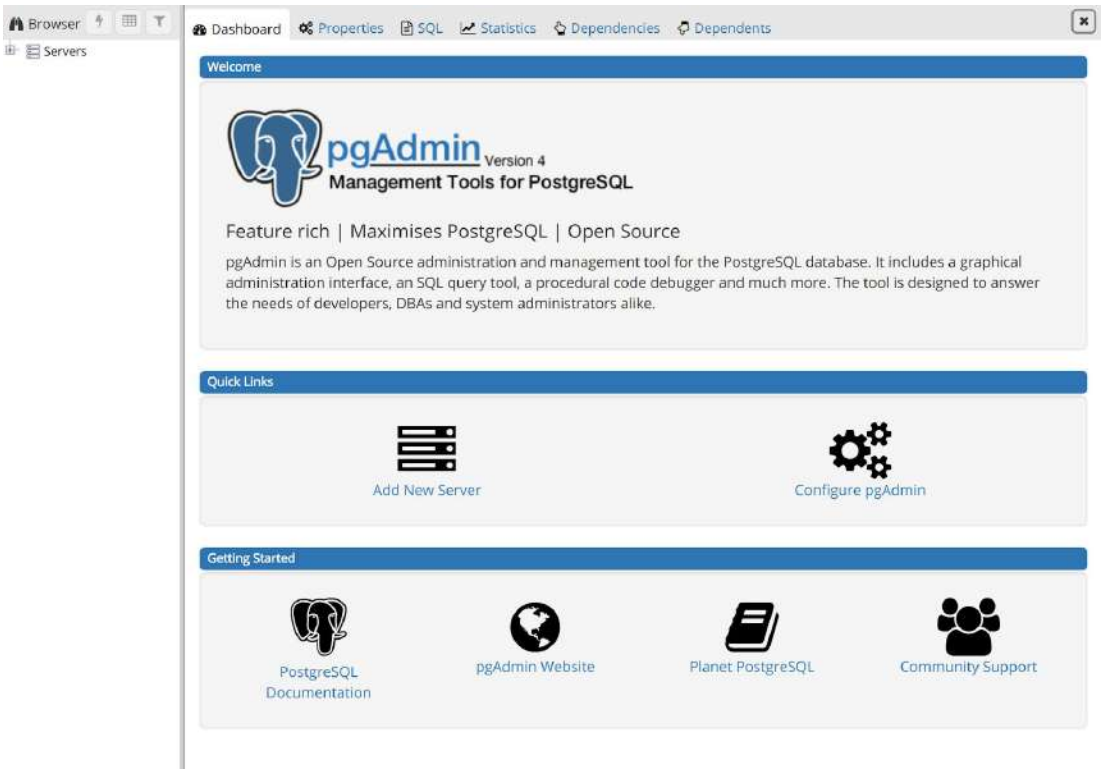
Una vez completada la instalación, abrimos nuestro navegador web y escribimos la dirección IP del servidor.

Iniciamos sesión



pgAdmin 1

Y obtenemos el panel de control pgAdmin.



pgAdmin 2

5.8. Instalamos Nextcloud 16

5.8.1. Pre-requisitos

- Nodo 1: IP estática 192.168.1.31
- Nodo 2: IP estática 192.168.1.32
- Exportar un directorio llamado /var/nfs/nextcloud en los nodos NFS
- Tener instalado nfs-common en ambos nodos
- Montar /var/nfs/nextcloud en /var/www/html/nextcloud/data

5.8.2. Creamos la Base de Datos

Nos dirigimos a nuestro cluster MariaDB y creamos una nueva base de datos

5.8.3. Descargamos y descomprimimos Nextcloud

```
cd /var/www/html
sudo wget https://download.nextcloud.com/server/releases/nextcloud-16.0.0.tar.bz2 -O nextcloud-16-latest.tar.bz2
tar -xvjf nextcloud-16-latest.tar.bz2
sudo chown -R www-data:www-data nextcloud
```

5.8.4. Configuramos Apache

```
sudo nano /etc/apache2/sites-available/nextcloud.conf
################
Alias /nextcloud "/var/www/html/nextcloud/"
<Directory /var/www/html/nextcloud/>
```

```
    Options +FollowSymlinks
    AllowOverride All
     <IfModule mod_dav.c>
       Dav off
     </IfModule>
    SetEnv HOME /var/www/html/nextcloud
    SetEnv HTTP_HOME /var/www/html/nextcloud
</Directory>
################
sudo a2ensite nextcloud
sudo a2enmod rewrite headers env dir mime
sudo sed -i '/^memory_limit =/s/=.*/= 512M/' /etc/php/7.2/apache2/php.ini
sudo systemctl restart apache2
```

5.8.5. Configuración desde la interfaz web

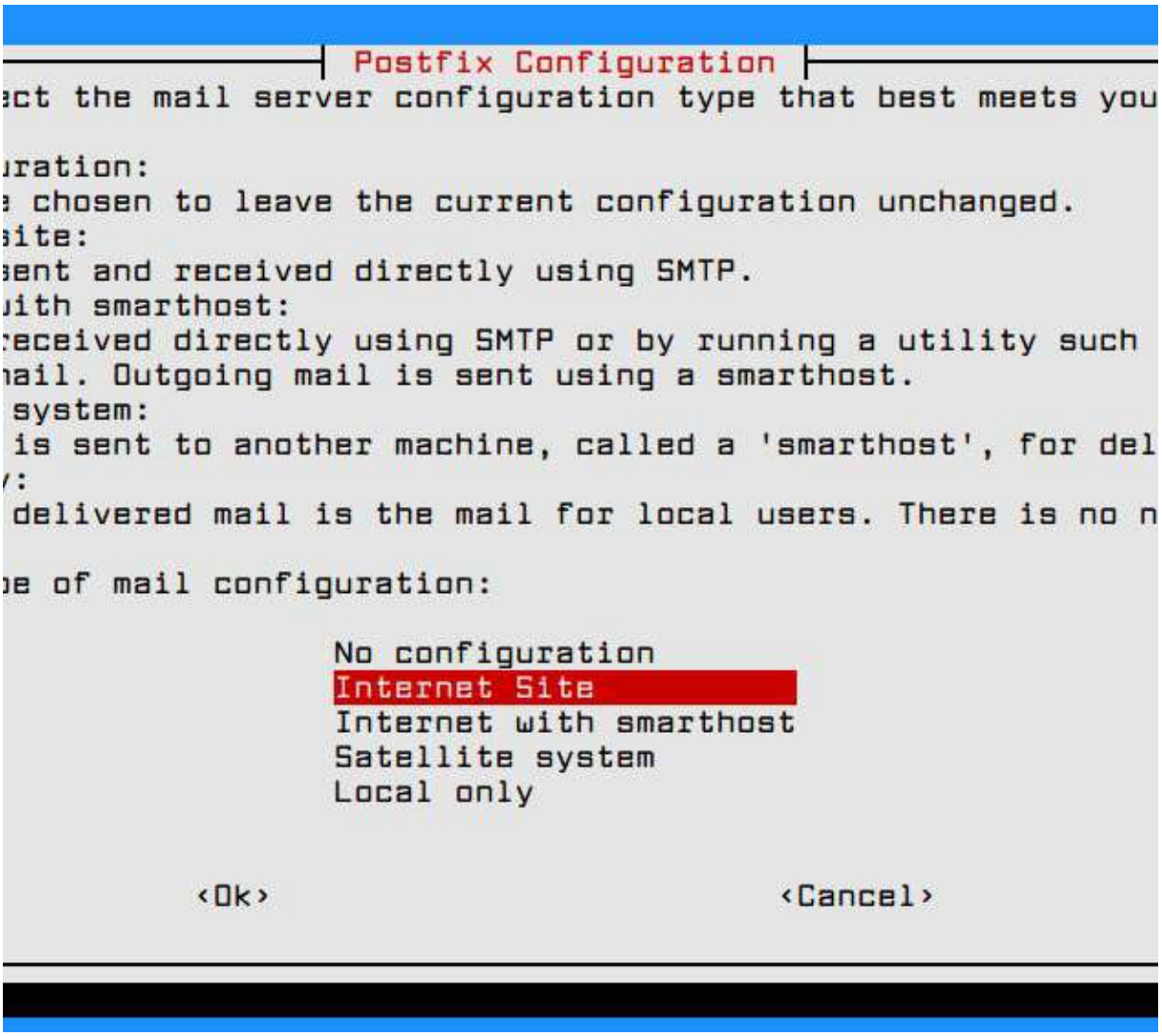
Ahora podemos apuntar nuestro navegador a https: //nextcloud.usemoslinux.net y finalizamos la instalación. Debemos crear un usuario/contraseña de administrador, así como completar la información necesaria de la base de datos.

Aplicamos los mismos pasos al nodo 2.

5.9. Instalamos Postfix para usar Gmail SMTP

apt install mailutils

Durante la instalación, proporcionamos cierta información necesaria para configurar Postfix. Seleccionamos Internet Site para habilitar Postfix en los correos enviados y recibidos y presionamos Entrar para continuar.



Instalación de Postfix.

Establecemos el nombre del correo.

5.9.1. Configuramos Postfix para usar Gmail SMTP

Abrimos el archivo de configuración de Postfix /etc/postfix/main.cf y configúrelo de la siguiente manera:

```
vim /etc/postfix/main.cf
```

5.9.2. Configurar el servidor de retransmisión Postfix

Buscamos la línea relayhost =y establecemos su valor en SMTPS de Gmail para que parezca.

```
relayhost = [smtp.gmail.com]:587
```

5.9.3. Habilitar autenticación SMTP

Habilitamos el soporte de Cyrus-SASL para la autenticación estableciendo el valor de smtp_sasl_auth_enablea yes.

```
smtp_sasl_auth_enable = yes
```

Configuramos Postfix para usar el archivo con las credenciales SASL. Esto se puede hacer definiendo la ruta de la sasl_passwd siguiente manera.

```
smtp_sasl_password_maps = hash: / etc / postfix / sasl_passwd
```

Configuramos las opciones de seguridad de SASL para deshabilitar las opciones que permiten la autenticación anónima.

```
smtp_sasl_security_options = noanonymous
```

5.9.4. Habilitar cifrado STARTTLS

Hacemos cumplir el cifrado STARTTLS para SMTP saliente con Postfix agregando la siguiente línea. Cuando se especifica un valor que no está vacía, esto anula los parámetros obsoletos smtp_use_tls, smtp_enforce_tls y smtp_tls_enforce_peername.

```
smtp_tls_security_level = encrypt
```

Definimos la ruta a los certificados de CA. Los certificados raíz públicos se encuentran generalmente /etc/ssl/certs/ca-certificates.crt en los sistemas Debian / Ubuntu.

```
smtp_tls_CAfile = /etc/ssl/certs/ca-certificates.crt
#########
...
relayhost = [smtp.gmail.com]: 587
...
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = picadillo: / etc / postfix / sasl_passwd
smtp_sasl_security_options = noanonymous
smtp_tls_security_level = cifrar
```

```
smtp_tls_CAfile = /etc/ssl/certs/ca-certificates.crt
########
```

5.9.5. Agregamos credenciales a sasl_passwd

Dado que Postfix actúa como cliente de correo, debemos saber cuándo proporcionar un nombre de usuario y contraseña. Por lo tanto, creamos el archivo sasl_passwd definido anteriormente /etc/postfix/sasl_passwd y configuramos las credenciales del servidor de retransmisión de correo como se muestra a continuación.

```
vim /etc/postfix/sasl_passwd
[smtp.gmail.com]:587 userid@gmail.com:password
```

Las credenciales se establecen en texto sin formato. Por lo tanto, para que sea un poco seguro, cambie la propiedad y el permiso a rooty read-write solo respectivamente.

```
chown root:root /etc/postfix/sasl_passwd
chmod 600 /etc/postfix/sasl_passwd
```

5.9.6. Crear archivo DB de sasl_passwd

```
postmap /etc/postfix/sasl_passwd
```

Esto asignará la misma propiedad y permisos al archivo de base de datos que el establecido para el archivo sasl_passwd anterior.

```
ls -l /etc/postfix/sasl_passwd*
-rw------- 1 root root    51 Jan  6 21:57 /etc/postfix/sasl_passwd
-rw------- 1 root root 12288 Jan  6 22:04 /etc/postfix/sasl_passwd.db
```

```
sudo systemctl restart postfix
```

5.10. Instalación Nagios 4

Nagios es un popular sistema de monitoreo de código abierto. Mantiene un inventario de sus servidores y los monitorea para que sepa que sus servicios críticos están en funcionamiento. El uso de un sistema de monitoreo como Nagios es una herramienta esencial para cualquier entorno de producción, ya que al monitorear el tiempo de actividad, el uso de la CPU o el espacio en disco, puede evitar problemas antes de que ocurran.

5.10.1. Pre requisitos

- Nodo 1: IP estática 192.168.1.
- Nodo 2: IP estática 192.168.1.

Instalamos los paquetes requeridos

```
$ sudo apt install autoconf gcc make unzip libgd-dev libmcrypt-dev libssl-dev dc
snmp libnet-snmp-perl gettext
$ cd ~
$ curl -L -O
https://github.com/NagiosEnterprises/nagioscore/archive/nagios-4.4.4.tar.gz
tar zxf nagios-4.4.4.tar.gz
cd nagioscore-nagios-4.4.4
```

Antes de construir Nagios, ejecutamos el configure script y especificamos el directorio de configuración de Apache:

```
$ ./configure --with-httpd-conf=/etc/apache2/sites-enabled
--with-mail=/usr/sbin/sendmail
```

```
########
*** Configuration summary for nagios 4.4.4 2019-07-29 ***:
 General Options:
 -------------------------
        Nagios executable:  nagios
        Nagios user/group:  nagios,nagios
       Command user/group:  nagios,nagios
            Event Broker:  yes
        Install ${prefix}:  /usr/local/nagios
    Install ${includedir}:  /usr/local/nagios/include/nagios
                Lock file:  /run/nagios.lock
  Check result directory:  /usr/local/nagios/var/spool/checkresults
          Init directory:  /lib/systemd/system
 Apache conf.d directory:  /etc/apache2/sites-enabled
             Mail program:  /bin/mail
                  Host OS:  linux-gnu
         IOBroker Method:  epoll
 Web Interface Options:
 ------------------------
```

HTML URL: http://localhost/nagios/

CGI URL: http://localhost/nagios/cgi-bin/

Traceroute (used by WAP):

Review the options above for accuracy. If they look okay,

type 'make all' to compile the main program and CGIs.

########

```
$ make all
```

Creamos una nagios de usuario y nagios grupo. Usamos para ejecutar el proceso de Nagios:

```
$ sudo make install-groups-users
```

Ahora ejecutamos estos make comandos para instalar archivos binarios de Nagios, archivos de servicio y sus archivos de configuración de muestra:

```
$ sudo make install
```

```
$ sudo make install-daemoninit
```

```
$ sudo make install-commandmode
```

```
$ sudo make install-config
```

Usamos Apache para servir la interfaz web de Nagios, así que ejecutamos lo siguiente para instalar los archivos de configuración de Apache y configurar sus ajustes:

```
$ sudo make install-webconf
```

Habilitamos Apache rewrite y los módulos cgi con el comando a2enmod:

```
$ sudo a2enmod rewrite
```

```
$ sudo a2enmod cgi
```

Para emitir comandos externos a través de la interfaz web a Nagios, agregamos el usuario del servidor web, www-data , al grupo nagios:

```
$ sudo usermod -a -G nagios www-data
```

Usamos el comando htpasswd para crear un usuario administrador llamado nagiosadmin que pueda acceder a la interfaz web de Nagios:

```
$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
$ sudo systemctl restart apache2
```

5.10.2. Instalación de los complementos de Nagios

```
$ cd ~
$ curl -L -O https://nagios-plugins.org/download/nagios-plugins-2.2.1.tar.gz
```

Extraemos el archivo NRPE y navegamos al directorio extraído:

```
$ tar zxf nagios-plugins-<^>2.2.1<^.tar.gz
$ cd nagios-plugins-2.2.1
$ ./configure
$ make
$ sudo make install
```

5.10.3. Instalación del complemento check_nrpe

Nagios monitorea los hosts remotos utilizando Nagios Remote Plugin Executor, o NRPE. Se compone de dos piezas:

- El check_nrpecomplemento que utiliza el servidor Nagios.

- El daemon NRPE, que se ejecuta en los hosts remotos y envía datos al servidor Nagios.

```
$ cd ~

$ curl -L -O
https://github.com/NagiosEnterprises/nrpe/releases/download/nrpe-3.2.1/nrpe-3.2.1.tar
.gz

$ tar zxf nrpe-3.2.1.tar.gz
$ cd nrpe-3.2.1
$ ./configure
$ make check_nrpe
$ sudo make install-plugin
```

5.10.4. Configuración de Nagios

Ahora realizamos la configuración inicial de Nagios, que implica editar algunos archivos de configuración. Solo necesitamos realizar esta sección una vez en su servidor Nagios.

```
$ sudo nano /usr/local/nagios/etc/nagios.cfg
```

Descomentamos la siguiente línea

```
#######
#cfg_dir=/usr/local/nagios/etc/servers
#######
```

Ahora creamos el directorio que almacenará el archivo de configuración para cada servidor que supervisará:

```
$ sudo mkdir /usr/local/nagios/etc/servers
```

```
$ sudo nano /usr/local/nagios/etc/objects/contacts.cfg
```

Configuramos el correo electrónico

```
########
define contact{
        contact_name    nagiosadmin             ; Short name of user
        use             generic-contact         ; Inherit default values from generic-contact
template (defined above)
        alias           Nagios Admin            ; Full name of user
        email           your_email@your_domain.com ;
########
```

A continuación, agregamos un nuevo comando a nuestra configuración de Nagios que nos permita usar el comando check_nrpe en las definiciones de servicio de Nagios. Abrimos el archivo /usr/local/nagios/etc/objects/commands.cfg en tu editor:

```
$ sudo nano /usr/local/nagios/etc/objects/commands.cfg
```

Agregamos los siguiente al final del archivo

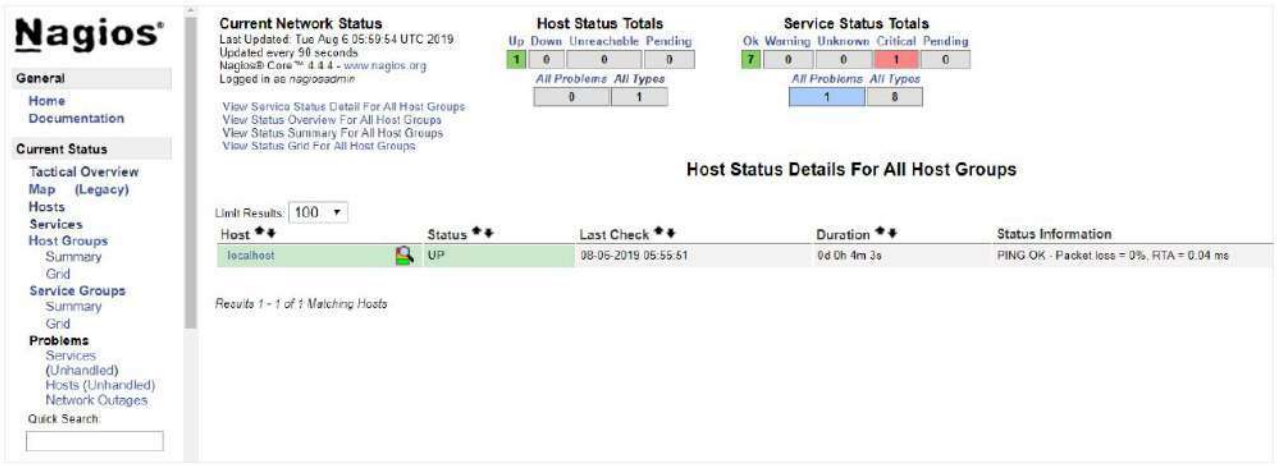
```
########
define command{
        command_name check_nrpe
        command_line $USER1$/check_nrpe -H $HOSTADDRESS$ -c $ARG1$
}
```

########

```
$ sudo systemctl start nagios
```

5.10.5. Acceder a la interfaz web de Nagios

Abrimos su navegador web favorito y vaya a su servidor Nagios visitando http://nagios_server_public_ip/nagios.



Interfaz web Nagios.

5.10.6. Instalación de complementos Nagios y NRPE Daemon en cada servidor a monitorear

Primero creamos un usuario nagios que ejecutará el agente NRPE:

```
$ sudo useradd nagios
```

Instalamos los requisitos previos de NRPE:

```
$ sudo apt install autoconf gcc libmcrypt-dev make libssl-dev wget dc build-essential gettext
```

Descargamos Nagios Plugins en su directorio de inicio con curl

```
cd ~
```

```
$ curl -L -O https://nagios-plugins.org/download/nagios-plugins-2.2.1.tar.gz
$ tar zxf nagios-plugins-2.2.1.tar.gz
$ cd nagios-plugins-2.2.1
$ ./configure
$ make
$ sudo make install
```

Instalamos y configuramos el daemon NRPE.

```
$ cd ~
$ curl -L -O
https://github.com/NagiosEnterprises/nrpe/releases/download/nrpe-3.2.1/nrpe-3.2.1.tar.gz
$ tar zxf nrpe-3.2.1.tar.gz
$ cd nrpe-3.2.1
$ ./configure
$ make nrpe
$ sudo make install-daemon
$ sudo make install-config
$ sudo make install-init
```

Ahora, actualizamos el archivo de configuración NRPE y agregamos algunas comprobaciones básicas que Nagios puede monitorear.

Primero, supervisamos el uso del disco de este servidor. Usamos el comando df -h para buscar el sistema de archivos raíz. Utilizamos este nombre de sistema de archivos en la configuración NRPE:

```
$ df -h /
```

Abrimos el archivo /usr/local/nagios/etc/nrpe.cfg en su editor:

```
$ sudo nano /usr/local/nagios/etc/nrpe.cfg
```

Localizamos estas configuraciones y modificamos:

```
#########
server_address=second_ubuntu_server_private_ip
allowed_hosts=127.0.0.1,::1,your_nagios_server_private_ip
command[check_vda1]=/usr/local/nagios/libexec/check_disk -w 20% -c 10% -p /dev/vda1
#########
```

Iniciamos NRE

```
$ sudo systemctl start nrpe.service
```

Nos aseguramos de que el servicio se esté ejecutando verificando su estado:

```
$ sudo systemctl status nrpe.service
```

A continuación, permitimos el acceso al puerto a 5666 través del firewall. Configuramos para permitir conexiones TCP al puerto 5666 con el siguiente comando:

```
$ sudo ufw allow 5666/tcp
```

Ahora podemos verificar la comunicación con el servidor remoto NRPE. Ejecutamos el siguiente comando en el servidor Nagios:

```
$ /usr/local/nagios/libexec/check_nrpe -H second_ubuntu_server_ip
```

5.10.7. Monitoreo de hosts con Nagios

Agregamos archivos de configuración para cada servidor especificando lo que deseamos monitorear. Luego podemos ver esos servidores en la interfaz web de Nagios.

En nuestro servidor Nagios, creamos un nuevo archivo de configuración para cada uno de los hosts remotos que deseamos monitorear /usr/local/nagios/etc/servers/.

```
$ sudo nano /usr/local/nagios/etc/servers/your_monitored_server_host_name.cfg
```

Agregamos la siguiente definición de host:

```
########
define host {
        use                     linux-server
        host_name               your_monitored_server_host_name
        alias                   My client server
        address                 your_monitored_server_private_ip
        max_check_attempts              5
        check_period                24x7
        notification_interval           30
        notification_period             24x7
}
```

########

Agregamos este bloque para monitorear el promedio de carga:

########

```
define service {
        use                     generic-service
        host_name               your_monitored_server_host_name
        service_description     Load average
        check_command           check_nrpe!check_load
}
```

########

La directiva use generic-service le dice a Nagios que herede los valores de una plantilla de servicio llamada servicio genérico, que está predefinida por Nagios.

Agregamos este bloque para monitorear el uso del disco.

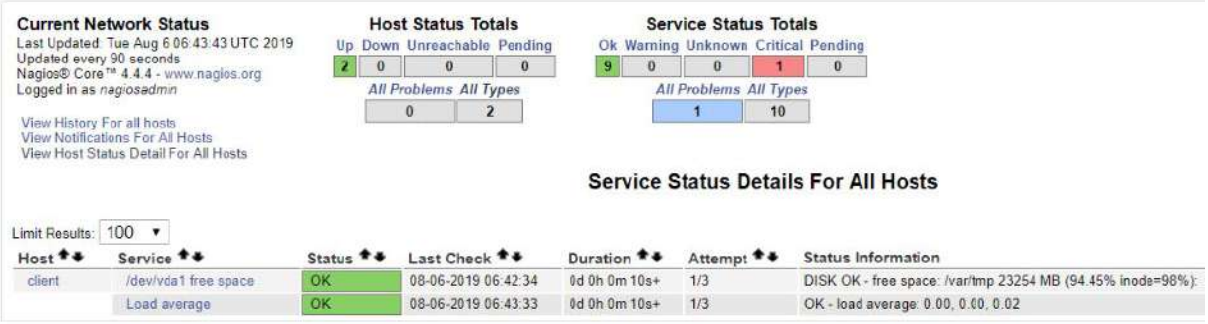
########

```
define service {
        use                     generic-service
        host_name               your_monitored_server_host_name
        service_description     /dev/vda1 free space
        check_command           check_nrpe!check_vda1
}
```

########

```
$ sudo systemctl restart nagios
```

Después de varios minutos, Nagios verificará los nuevos servidores y veremos en la interfaz web de Nagios. Hacemos clic en el enlace Servicios en la barra de navegación izquierda para ver todos nuestros servidores y servicios monitoreados.



Monitoreo de Servidores

CAPÍTULO VI

CONCLUSIÓN Y RECOMENDACIONES

En conclusión, un clúster trae enormes beneficios como también dificultades. Podemos nombrar como beneficios la alta disponibilidad de servicios y alta escalabilidad así como un balanceo de carga entre nodos para no saturar los nodos con demasiada concurrencia. También beneficia a la fácil administración, concentrando todas las aplicaciones con requerimientos de software similares en un mismo clúster, así se evita instalar los requerimientos de cada aplicación cada vez en cada nodo.

Como dificultad principal podemos mencionar el conocimiento avanzado en servidores Linux que un informático debe poseer para poner en marcha este clúster, también se requiere de mucho presupuesto para el ensayo/error antes de utilizar el clúster en producción.

Como recomendación final, este trabajo se puede implementar en la red de servidores de la Facultad que actualmente están alojados en la Senatics (Nube Py), no se presentan mayores dificultades, hay personas altamente calificadas para la puesta en marcha de este proyecto de Clúster, además parte del esquema que utiliza este clúster ya está en funcionamiento (pfSense, HAProxy)

Bibliografía

Buyya, R. (1999). High Performance Cluster Computing: Architectures and Systems. NJ.

S. Coréz y M. Galarza (2005), Cluster de Alta Disponibilidad para Servidores Web en LINUX (pg. 5).

Oñate, A., Ortega, V. (2010). Clúster de alta disponibilidad con balanceo de carga para servicios corporativos sobre Debian GNU/Linux.

A. Khare, Y. Huang, H. Doan y M. Sing (2015), A Fresh Graduate's Guide to Software Development Tools and Technologies

Bourke, T. (2001). Server Load Balancing. Sebastopol, CA, United States of America: O'Reilly Media, Incorporated.

Kent Roberts, K. R. (2018b, 27 noviembre). What is Server Load Balancing? The Function of a Load Balancer. Recuperado 12 agosto, 2019, de https://www.atlantic.net/hipaa-data-centers/server-load-balancing/

IBM Knowledge Center. (s.f.). Recuperado 12 agosto, 2019, de https://www.ibm.com/support/knowledgecenter/en/ssw_ibm_i_71/rzajw/rzajwviproute.html

Sklaroff, J. R. (1976). Redundancy Management Technique for Space Shuttle Computers. IBM Journal of Research and Development.

What is Data Replication: Definition | Informatica India. (s.f.). Recuperado 13 agosto, 2019, de

https://www.informatica.com/in/services-and-training/glossary-of-terms/data-replication-definition.html

Sysel, M., & Doležal, O. (2013). An Educational HTTP Proxy Server. Documento presentado en 24th DAAAM International Symposium on Intelligent Manufacturing and Automation, Zadar, Croatia. Recuperado de https://core.ac.uk/download/pdf/81219099.pdf

IP Addressing: NAT Configuration Guide, Cisco IOS Release 15M&T - Configuring NAT for IP Address Conservation [Support]. (2018, 16 octubre). Recuperado 20 agosto, 2019, de https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/ipaddr_nat/configuration/15-mt/nat-15-mt-book/iadnat-addr-consv.html

Komar, B., Beekelaar, R., & Wettern, J. (2003). Firewalls For Dummies. New York, United States of America: Wiley.

HAProxy - The Reliable, High Performance TCP/HTTP Load Balancer. (s.f.). Recuperado 20 agosto, 2019, de http://www.haproxy.org/

Keepalived for Linux. (s.f.). Recuperado 20 agosto, 2019, de https://www.keepalived.org/

First Hop Redundancy Protocols Configuration Guide, Cisco IOS XE Release 3S - Configuring VRRP [Cisco IOS XE 3S]. (2017, 6 septiembre). Recuperado 20 agosto, 2019, de https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/ipapp_fhrp/configuration/xe-3s/fhp-xe-3s-book/fhp-vrrp.html

IP Routing: BFD Configuration Guide, Cisco IOS Release 15M&T - Bidirectional Forwarding Detection [Cisco IOS 15.4M&T]. (2018, 19 diciembre). Recuperado 20 agosto, 2019, de https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/iproute_bfd/configuration/15-mt/irb-15-mt-book/irb-bi-fwd-det.html

11.4. Servidor de archivos NFS. (s.f.). Recuperado 21 agosto, 2019, de https://debian-handbook.info/browse/es-ES/stable/sect.nfs-file-server.html

The Postfix Home Page. (s.f.). Recuperado 21 agosto, 2019, de http://www.postfix.org

Charles, C. (2018, 23 mayo). About MariaDB - MariaDB.org. Recuperado 21 agosto, 2019, de https://mariadb.org/about/

Welcome! - The Apache HTTP Server Project. (s.f.). Recuperado 21 agosto, 2019, de https://httpd.apache.org/

OpenSSH. (s.f.). Recuperado 21 agosto, 2019, de https://www.openssh.com

Rsync(1) - Linux man page. (s.f.). Recuperado 22 agosto, 2019, de https://linux.die.net/man/1/rsync

Metrics | Definition of metrics by Lexico. (s.f.). Recuperado 29 agosto, 2019, de https://www.lexico.com/en/definition/metrics

Derek Haynes, D. (2015, 24 febrero). Understanding Linux CPU stats. Recuperado 27 agosto, 2019, de https://scoutapm.com/blog/understanding-linuxs-cpu-stats

Definición de RAM — Definicion.de. (s.f.). Recuperado 29 agosto, 2019, de https://definicion.de/ram/

Chapter 13. Swap Space. (s.f.). Recuperado 27 agosto, 2019, de https://docs.fedoraproject.org/en-US/Fedora/14/html/Storage_Administration_Guide/ch-swapspace.html

Netstat(1) - OpenBSD manual pages. (s.f.). Recuperado 29 agosto, 2019, de https://man.openbsd.org/netstat

Nagios Core. Download Nagios Core For Free Here.. (s.f.). Recuperado 24 septiembre, 2019, de https://www.nagios.org/projects/nagios-core/

Hernández Sampieri, R., Fernández Collado, C. and Baptista Lucio, P. (2010). Fundamentos de metodología de la investigación. Madrid: McGraw-Hill.

Operating System and Hardware Requirements | HAProxy Enterprise 1.5r2. (s.f.). Recuperado 21 septiembre, 2019, de https://www.haproxy.com/documentation/hapee/1-5r2/getting-started/os-hardware/

The Four Essential Sections of an HAProxy Configuration. (2019, 11 mayo). Recuperado 8 noviembre, 2019, de https://www.haproxy.com/blog/the-four-essential-sections-of-an-haproxy-configuration/

How to use ssh-keygen to generate a new SSH key | SSH.COM, (s.f.). Recuperado 15 de noviembre, 2019, de https://www.ssh.com/ssh/keygen/

Anexo A

Entrevista

ENTREVISTA SOBRE LA ESTRUCTURA DE SERVIDORES DE LA FACULTAD DE CIENCIAS Y TECNOLOGÍAS.

Fecha: 01/07/2019 **Hora:** 09:00

Lugar de la entrevista: Facultad de Ciencias y Tecnología, con sitio en Sargento Florentín Benítez, Coronel Oviedo, Paraguay.

Entrevistadores: Fernando Rojas y Juan Silvero

Entrevistado: Lic. Cristhian Mendieta

Área: Coordinación y gestión de soporte informático

Cargo que ocupa: Encargado de Coordinación y gestión de soporte informático

1. ¿Qué entidad es la provee a la facultad de servidores?

   La Senatics con Nube PY

2. ¿Qué tipo de infraestructura provee?

   VPS en la nube

3. ¿Las máquinas virtuales (VPS) qué tipo de IP tienen?

   Tienen IP privada

4. ¿Las máquinas virtuales (VPS) están conectados en una red LAN?

   Todas las máquinas están conectadas en una red LAN

5. ¿Cómo tienen acceso a la red Internet las máquinas virtuales?

Mediante una IP pública brindada por la entidad

6. ¿La IP pública está asociada a un servidor VPS?

   Sí, la IP pública está asociada a un servidor VPS

7. ¿Qúe tipo de software utilizan en la máquina que posee la IP pública?

   Tiene instalado pfSense, que es una distribución de FreeBSD. Sirve por Router NAT y como Firewall

8. ¿Hay algún middleware entre pfsense y las máquinas virtuales?

   El middleware que se utiliza es HAProxy, se encarga de dirigir las peticiones que provienen de pfSense a las máquinas virtuales

9. ¿Cómo se pone en marcha un nuevo servicio para la Facultad?

   Se crea una nueva instancia VPS, se instalan los requerimientos de software y se instala el servicio

10. ¿Hay alguna limitación en la cantidad de máquinas virtuales?

    Existe una limitación en la cantidad de máquinas que se puede crear, pero es cuestión de solicitar el aumento de este límite.

11. ¿Cuáles son los límites que impone Senatics para sus servidores?

    Nos permite crear 40 servidores VPS. El espacio en disco que nos provee para la totalidad de Servidores es de 1 TB. No tiene restricciones en memoria Ram ni Procesadores.

12. ¿Alguna vez se saturó un servidor?

    Si. Varias veces y tuve que aumentar las capacidades de Ram y Procesador.