FACULTA DE INGENIERIA Y CIENCIAS APLICADAS

DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO MODULAR DE PROCESAMIENTO, RED Y ALMACENAMIENTO DE HIPERCONVERGENCIA MEDIANTE RASPBERRY PI

AUTOR

JOSE VICENTE CORREA VILLALBA

AÑO

2020

FACULTAD DE INGENIERIA Y CIENCIAS APLICADAS

DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO MODULAR DE PROCESAMIENTO, RED Y ALMACENAMIENTO DE HIPERCONVERGENCIA MEDIANTE RASPBERRY PI

Trabajo de Titulación presentado en conformidad con los requisitos establecidos para optar por el título de Ingeniero en Redes y Telecomunicaciones

Profesor Guía

MBA. José Julio Freire Cabrera

Autor:

José Vicente Correa Villalba

Año:

2020

DECLARACIÓN DEL PROFESOR GUIA

“Declaro haber dirigido este trabajo, Diseño e implementación de un prototipo modular de procesamiento, red y almacenamiento de hiperconvergencia mediante Raspberry PI, a través de reuniones periódicas con el estudiante José Vicente Correa Villalba en el semestre 202020, orientando sus conocimientos y competencias para un eficiente desarrollo del tema escogido y dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación"

José Julio Freire Cabrera

Magister en Gerencia Empresarial.

CI: 1709731457

DECLARACIÓN DEL PROFESOR CORRECTOR

““Declaro haber dirigido este trabajo, Diseño e implementación de un prototipo modular de procesamiento, red y almacenamiento de hiperconvergencia mediante Raspberry pi, a través de reuniones periódicas con el estudiante José Vicente Correa Villalba en el semestre 202020, dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación”.

Nathaly Veronica Orozco Garzón

Doctora en Ingeniería Eléctrica en el área de Telecomunicaciones y Telemática

CI: 1720938586

DECLARACION AUTORIA DEL ESTUDIANTE

“Declaro que este trabajo es original, de mi autoría, que se han citado las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales que protegen los derechos de autor vigentes.”

José Vicente Correa Villalba

CI:171975240-2

AGRADECIMIENTOS

Mi agradecimiento en primera instancia es a Dios, pues sin Él nada hubiera sido posible en mi vida, a mi familia que siempre estuvo allí apoyándome en los momentos más difíciles, a la Universidad y a cada uno de los docentes que dedicó su tiempo para compartir sus conocimientos a lo largo de toda la carrera.

DEDICATORIA

Dedico este trabajo a mi padre Nicolás Ricardo y a mi madre Gloria Alejandrina, quienes me brindaron su apoyo incondicional a lo largo de toda la carrera.

A mis hermanos y hermanas quienes de una u otra forma aportaron con un granito de arena para hacer posible este sueño.

Y finalmente a la Universidad que durante más de 5 años se convirtió en mi segundo hogar y a cada de los docentes que supo impartirme sus conocimientos y experiencias.

RESUMEN

Este trabajo presenta una propuesta de bajo costo como solución alternativa para microempresas y PYMES que requieran los servicios de procesamiento, red y almacenamiento de datos. El prototipo que se va a presentar está orientado como un servidor de bajo costo que trabaja mediante *software* libre como el Raspberry PI OS, *Docker swarm* para la creación del *cluster*, Samba para compartir archivos y dispositivos en internet, Apache o Nginx para los servicios web. Se realizará las pruebas necesarias de funcionamiento de la transmisión de información de la parte de almacenamiento, la velocidad de transmisión, el tiempo de respuesta del servicio web y la redundancia del *cluster*. Para comprobar el beneficio al final se realizará un análisis de mercado y costo para comparar el costo-beneficio del prototipo vs implementaciones existentes en el mercado a fin de determinar que a largo plazo el prototipo brindará prestaciones similares a un costo más conveniente.

Palabra clave: Raspberry PI, hiperconvergencia, almacenamiento, *cluster* Raspberry, docker *swarm*, Raspberry PI OS

ABSTRACT

This work presents a low-cost proposal as a workaround for micro-enterprises and PYMES, that require data processing, network and storage services. The prototype to be presented is oriented as a low-cost server that works with free *software* such as the Raspberry PI OS, *Docker swarm* for cluster creation, Samba for sharing files and devices on the Internet, Apache or Nginx for *web service*s. The necessary tests of storage part information transmission, transmission rate, *web service* response time and cluster redundancy will be performed. To verify the profit in the end, a market and cost analysis will be performed to compare the cost-benefit of the prototype vs existing implementations on the market to determine that in the long term the prototype will provide similar features at a more convenient cost.

Keywords: *Raspberry PI, hyperconvergence, storage, Raspberry cluster, docker swarm, Raspberry PI OS*

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Antecedentes

Dia a día los usuarios y empresas generan una gran cantidad de información que para ellos es considerada información confidencial y de delicado tratamiento, esto conlleva a los ingenieros en Telecomunicaciones y otras ramas similares a buscar permanentemente soluciones y alternativas para salvaguardar la información, sobre todo de las empresas.

En el mercado actual existen diferentes tipos de soluciones para el albergamiento de toda esta información, sea esta en servidores físicos o en los modernos y demandados servidores de almacenamiento de datos en la nube, los cuales proveen y cubren las necesidades actuales de una gran empresa como Amazon *Web service* o Google Cloud.

Sin embargo, una de las principales desventajas que poseen estas soluciones, sea esta de la creación de un centro de datos físico o el alquiler mensual de un servidor en la nube, es su alto costo de arrendamiento y mantenimientos, por lo que muchas de estas soluciones se encuentran fuera del alcance de las pequeñas empresas, PYMES y microempresas familiares.

Este trabajo de tesis propone experimentar en el centro de datos de la Universidad de las Américas la posibilidad de crear un prototipo económico, que pueda ser una alternativa y solución en hiperconvergencia de datos, que proporcione servicios de almacenamiento y consulta de datos a las PYMES y microempresas que a su vez les permitiría tener todos los servicios para hacer crecer sus negocios.

El objetivo es demostrar que mediante la creación de un *cluster* prototipo, utilizando placas Raspberry PI, se podría crear un equipo de procesamiento que pueda alojar un servidor web, servicios de almacenamiento, redundancia y red, el cual podría ser una opción como solución para el funcionamiento de varias PYMES y micro empresas.

Alcance

Se creará un prototipo de *cluster* con placas Raspberry PI, dentro mediante el uso del S.O. Raspberry PI OS para el procesamiento de datos mismo que tendrá un diseño de almacenamiento y red y además mediante la virtualización se podrá proveer servicios web, base de datos, etc.

Se realizará un análisis de mercado sobre los costos de los servidores de datos y servidores en la nube y se realizará una comparación de costos para determinar el beneficio económico que podría ofrecer este prototipo.

El prototipo tendrá funcionalidad inicial a nivel local, para el entorno de la intranet de la PYME o microempresa.

Justificación

La revolución tecnológica en la actualidad ha exigido a las PYMES y microempresas dar un salto enorme a la digitalización para poder mantenerse competitivos en el mercado y exponer sus productos de la mejor manera.

Estos cambios de realidad social son a nivel mundial y el objetivo de los estudiantes y las universidades es brindar a la sociedad servicios y laboratorios que permitan desarrollar e implementar más opciones tecnológicas para proveer soluciones rentables, por lo que el prototipo servirá como un piloto con fines educativos y experimentales para futuras aplicaciones y promociones.

Además, este prototipo en caso de tener los mejores resultados y poder comercializarlo se podría convertir en una solución alternativa para las PYMES y micronegocios que requieren un servidor de hiperconvergencia o centro de datos, ya que, en la mayoría de los casos, la adquisición de estos servidores supone un costo demasiado alto, así mismo el arrendamiento de servicios similares demandan de altos recursos económicos, muchas veces inalcanzables para este sector empresarial. De esta manera se podría realizar un aporte valioso a la sociedad.

Objetivo General

Diseñar un prototipo de hiperconvergencia mediante un *cluster* con el uso de tarjetas Raspberry PI.

Objetivos Específicos

a) Realizar un análisis de la situación actual sobre la demanda de este tipo de servicios

b) Analizar todas las alternativas del mercado referentes a hiperconvergencia con el fin de obtener una guía y referencias actuales sobre el tema.

c) Plantear una alternativa económica mediante un prototipo mismo que será el entregable para el centro de datos Experimental de la Universidad de Las Américas.

d) Diseñar un prototipo de *cluster* sobre elementos Raspberry, empleando *software* libre para este fin y probar su funcionamiento.

1. CAPÍTULO I. MARCO TEÓRICO

En este capítulo se revisará los términos básicos de las redes de datos, tipos de redes, los centros de datos, topologías, sistemas embebidos y demás componentes que conforman la parte del *hardware* que se utilizará en el diseño e implementación del proyecto y también se revisará los conceptos de *software,* tales como los sistemas operativos, hipervisores, contenedores y programas ya que, sin este componente el *hardware* por sí solo no podría operar.

1.1. Las redes de comunicaciones

Bajo la denominación de redes de comunicaciones se agrupa a la totalidad de los elementos de *hardware* y *software* que conforman los distintos tipos de redes de comunicaciones con el objetivo de intercambiar información entre un emisor y un receptor, mismos que pueden estar en dos puntos geográficos remotos utilizando diversas tecnologías (Castro, 2015).

En la actualidad casi todas las redes están digitalizadas por lo que podríamos decir en términos absolutos que, lo que se está intercambiando son bits con independencia del contenido de información que los mismos transportan (Castro, 2015).

En la figura 1 se puede observar un esquema general de las redes de comunicaciones, en el cual en el centro se visualiza la red de internet y a su alrededor la interconexión de las otras redes para acceder a los servicios, como son correo electrónico, páginas web, redes de intranet, servicios bancarios, etc.

Figura 1 Esquema general de las redes de comunicaciones.

Tomado de: *(Cisco, 2019)*.

1.1.1. Componentes de las redes de comunicaciones

Los principales componentes de las redes de comunicaciones son el *hardware* y *software*, de las cuales se detallará a continuación.

1.1.1.1. El *hardware*

El *hardware* son los elementos físicos que conforman la red y se componen principalmente de las siguientes partes:

Servidores: son el corazón de las redes y administran la información y los procesos para el manejo de los datos. Los servidores procesan información que se transporta a través de la red, son básicamente computadores y sistemas embebidos de los cuales se ampliará la información en la sección 1.3.

Ruteador o *router*: realizan el enrutamiento y el transporte de la información a través de la red, estos equipos conforman el core del transporte de la red.

Conmutador o *switch*: son los encargados de la interconexión de equipos dentro de una misma red. Otra de sus funciones es la amplificación de la señal que va a ser transmitida en el intercambio de datos.

Gabinetes o *Racks*: son gabinetes metálicos donde se ubican cada uno de los componentes que comprende una red o un centro de datos.

Cableado estructurado: el cableado estructurado provee el medio guiado para el transporte de los datos, son de tipo horizontal y backbone o troncales.

1.1.1.2. El *software*

Cualquier estructura y topología de red, sino cuenta con un complemento que la controle y direccione las acciones, solo sería un conjunto de cables y circuitos sin ninguna función. El complemento o controlador de las redes es el *software*.

El *software* es la parte lógica la cual es programable según la necesidad de las aplicaciones de una empresa y de la red, capaz de controlar la parte física de una red de datos, cada componente de la red tiene un *software* que lo controla y determina las funciones a ejecutar.

Existe gran cantidad de *software* alrededor de la red, que ofrece soluciones, pero hay un *software* que ha logrado que las redes de datos crezcan de manera exponencial, este es el sistema operativo (SO), cuyo objetivo fundamental en la red de datos y computadoras es resolver los problemas del usuario más fácilmente, permitiendo utilizar el *hardware* de forma sencilla mediante programas y aplicaciones que precisan ciertas operaciones comunes, como el control de dispositivos de

entrada y salida (E/S), operaciones usuales de control y asignación de recursos. (Silderchatz, 2006).

El concepto de *software* se ampliará en la sección 1.4.

1.1.2. Tipos de Redes de comunicaciones

Las redes de comunicaciones tienen varias clasificaciones, pero en este trabajo revisaremos la clasificación por cobertura geográfica:

- Redes Personales (PAN)
- Redes Locales (LAN)
- Redes Metropolitanas (MAN)
- Redes Extendidas (WAN)
- El Internet

1.1.2.1. Redes PAN

Es la red de área personal, cuyo alcance es menor a los 100 metros, por lo general conecta dispositivos pequeños y accesorios de los usuarios con las redes LAN u otros dispositivos compatibles cercanos al usuario. En la figura 2, podemos apreciar como un receptor *bluetooth* puede crear una red interconectando periféricos, como un auricular, una impresora, una cámara, un *smartphone*, etc.

Figura 2 Estructura de una red PAN creada por medio de *bluetooth.*

Tomado de (Abelinoelpaspi, 2016).

1.1.2.2. Redes LAN

Su alcance aproximado va desde los 100 metros hasta 1 kilómetro y su función es interconectar las comunidades de una misma organización, basadas en computadores ubicados en un mismo lugar (edificio, escuela, etc) o grupos localizados de edificios. Al encontrarse dentro de un mismo establecimiento, es común mantener la Red LAN y sus datos inaccesibles al exterior, por esta razón se conoce también como red de datos privada (Halsall, 1998) . La figura 3 simplifica el ejemplo de una red LAN, en la cual se observa la interconexión de dispositivos dentro de un edificio que puede pertenecer a un empresa o corporación.

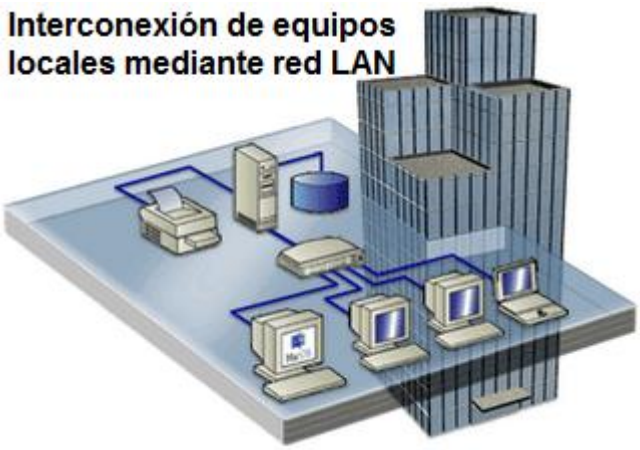

Figura 3 Ejemplo de estructura de red LAN, en una corporación.

1.1.2.3. Redes MAN

Una red MAN puede interconectar varias redes LAN como se muestra en la figura 4, ya que su rango va desde 1 km hasta aproximadamente 50km, en este entorno las interconexiones se realizan dentro de una ciudad en la cual una organización requiera tener conectado a su red las sucursales que tenga desplegadas alrededor de dicha ciudad.

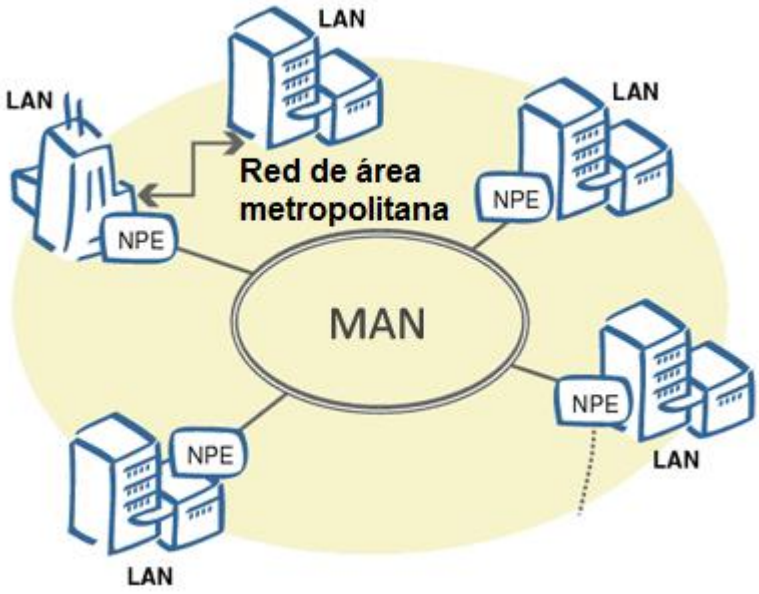

Figura 4 Estructura de una red MAN interconectando varias redes LAN.

Adaptado de (Tiposderedesjt, 2014)

1.1.2.4. Redes WAN

Una red WAN tiene un alcance aproximadamente puede ir desde los 50 hasta los 1000 kilómetros, y su funcione es la interconexión de ciudades, países e inclusive contenientes, como se muestra en la figura 5.

Figura 5 Estructura de una red WAN interconectado países y continentes.

Tomado de (Conceptode, 2020).

1.1.2.5. Internet

La internet es conocida a nivel mundial como la red de redes, está conformada por todos los tipos de redes anteriormente mencionadas, se la podría definir como una red internacional y mundial que están interconectadas mediante protocolos y procedimientos normalizados y estándares de internet.

En esencia el internet no es más que la comunicación y transferencia de datos entre servidores y centro de datos a nivel mundial, donde se albergan información, documentación, contenidos, multimedia y datos que son consumidos por los usuarios finales.

Actualmente si una empresa quiere fortalecer su presencia en el mercado según su modelo de negocio, necesariamente debe actualizarse y ofrecer sus servicios mediante internet, lo cual hasta hace 15 años atrás no sucedía, ya que en muchas de las reuniones gerenciales tan solo se tocaba el tema para decidir si contratar o no internet, mas ahora aquellas reuniones son para decidir qué proveedor contratar, la velocidad y como se va a distribuir la red dentro de la empresa.

Es así como el internet paso de ser un instrumento de promoción de un negocio a ser la razón de la comercialización de los productos, ya que todo está enlazado entre sí por medio de esta red como se muestra una simulación en la figura 6.

Figura 6 Estructura de la red de Internet.

Tomado de (Marketing directo, 2018).

1.1.3. Topologías de Red

Es el mapa lógico de la red donde está habilitada el intercambio de datos, y por lo general la topología de malla suele ser la más utilizada en las redes WAN ya que esto permite la redundancia (Hallsal, 1998). Sin embargo, se presenta las topologías de red más comunes:

- Estrella
- Malla
- Anillo
- Bus

1.1.3.1. Estrella

La topología de red en estrella está diseñada con un nodo central como se muestra en la figura 7, cuya función es concentrar y distribuir todo el tráfico de comunicaciones existentes, al que están conectados el resto de los equipos terminales correspondientes (Castro, 2015).

El principal inconveniente de una red de estrella es que si el equipo de conmutación central falla toda la red deja de funcionar, para ello como solución se debe tener otro equipo de conmutación de respaldo.

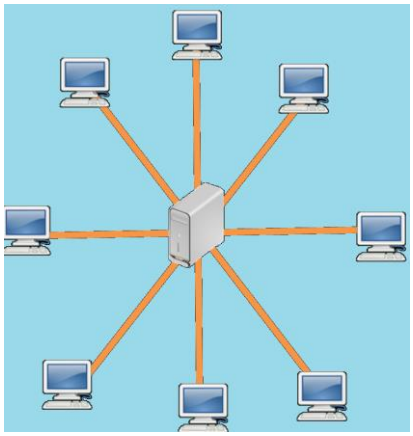

Figura 7 Estructura de la topología estrella.

1.1.3.2. Malla

La topología de malla no cuenta con un nodo central, es decir, todos están conectados entre sí como se puede apreciar en la figura 8, de manera que no existe un predominio de uno sobre otro. En cuanto a la concentración de tráfico, la malla por lo general esta complementada por vínculo entre nodos no adyacentes que se instalan para mejorar las características del tráfico (Castro, 2015).

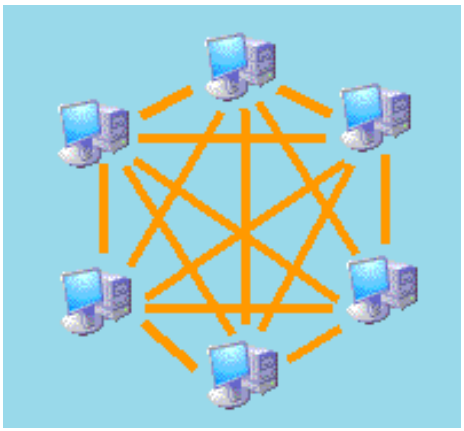

Figura 8 Estructura de la topología malla.

1.1.3.3. Anillo

En esta topología cada equipo terminal está conectado a los dos equipos contiguos formando un anillo, como se muestra en la figura 9, por tanto, si un nodo o elemento de la red se detiene, toda la red podría presentar fallos. (Castro, 2015).

Figura 9 Estructura de la topología de anillo.

Otro inconveniente propio de la configuración de la red anillo radica en que puede quedar disminuida notablemente la velocidad de la red, al depender de la respuesta de los otros nodos, sin embargo, se han tratado de diseñar técnicas para mejorar la confiabilidad de estas redes y existen implementaciones que solucionan el problema de la conexión y la velocidad (Castro, 2015).

1.1.3.4. Bus

En este tipo de red, cada equipo terminal se conecta a un bus como el de la figura 10, por el que circulan los datos que tienen como destino un equipo de la red y que se transmite por el proceso de difusión (Castro, 2015).

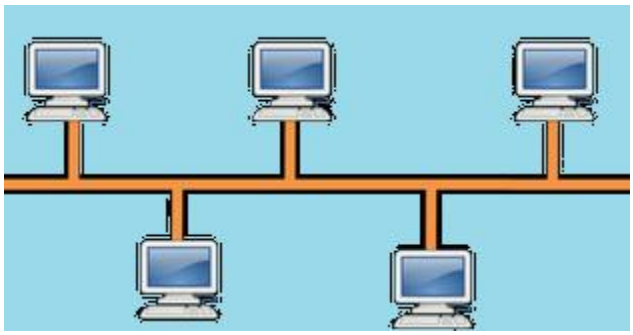

Figura 10 Estructura de la topología bus.

1.2. Centro de datos

Los centros de datos son áreas específicas donde una empresa instala y mantiene los equipos de las tecnologías de la información (TI), mainframes, servidores y bases de datos. Anteriormente las operaciones y la información se mantenían centralizadas en este lugar y todos los sistemas residían en un espacio físico, de ahí el nombre de centro de datos (Gartner, 2020).

El término “sistemas” continúa usándose para referirse al departamento que tiene la responsabilidad de estos administrar esta infraestructura, sin importar cuán dispersos estén (Gartner, 2020).

Las tendencias del mercado y de la industria están cambiando la forma en que las empresas abordan sus estrategias de centro de datos. Varios factores están

impulsando a las empresas a mirar más allá de la infraestructura de tecnología tradicional y transformar la forma en que ven el entorno de su centro de datos y los procesos comerciales, así como se puede evidenciar en la figura 11, los centros de datos se están organizando para prestar estos servicios de forma externa por un costo mensual o anual, a fin de que las empresas puedan prescindir del montaje de estas infraestructuras. Estos incluyen infraestructuras de centros de datos antiguas que corren el riesgo de no cumplir con los requisitos comerciales futuros, una conciencia constante de los costos y la necesidad de ser más eficientes energéticamente (Gartner, 2020).

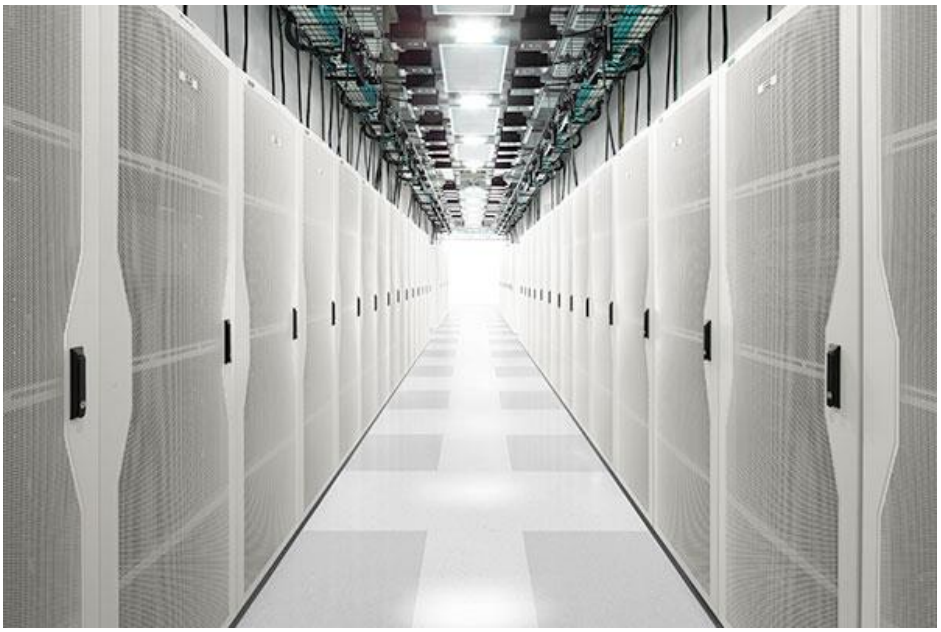

Figura 11 Ejemplo de la estructura interna de un centro de datos.

Tomado de (Cisco DC, 2020).

1.2.1. Componentes del centro de datos

La infraestructura del centro de datos está clasificada en 5 partes fundamentales, como puede apreciarse en la figura 12:

- Computo: es el conjunto de servidores o computadores que se encarga de procesar la información del centro de datos

- Seguridad: puede estar compuesta *hardware* o *software,* y proporcional un esquema de acceso a la información solo a personal autorizado mediante credenciales o técnicas de cifrado de datos.
- Networking: provee la conexión interna entre los distintos componentes del centro de datos y la salida al exterior.
- Almacenamiento: se encarga de albergar la información de los usuarios del centro de datos.
- Virtualización de Aplicaciones: ayuda a optimizar el uso de los recursos informáticos de manera eficiente y eficaz

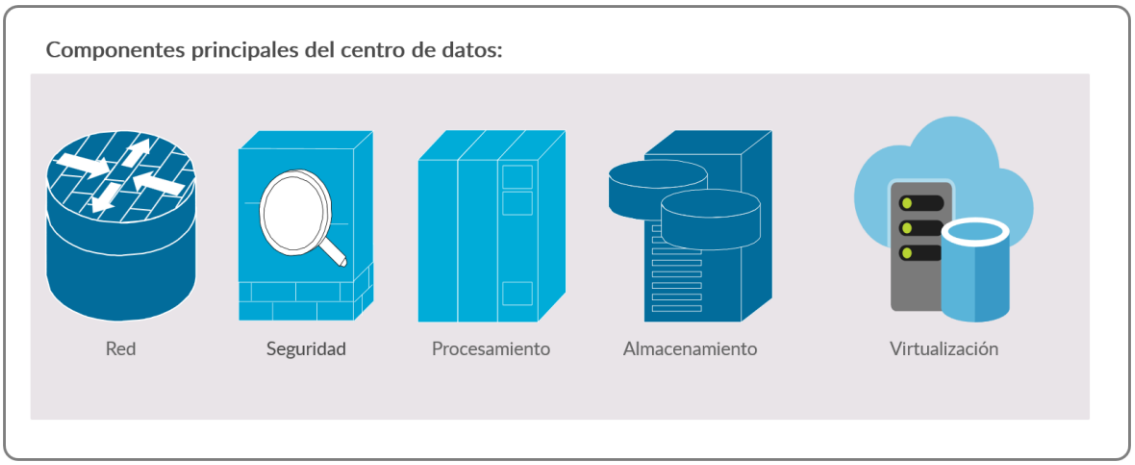

Figura 12 Componentes Básicos de la infraestructura del centro de datos.

Adicional a ello un centro de datos requiere otros elementos complementarios para su funcionamiento según el sitio:

- Cableado Estructurado: son técnicas para manejo correcto de cableado.
- Energía: la energía que alimenta los equipos, adicional al generador principal debe tener un sistema adicional que permita al centro de datos no dejar de operar a pesar de cortes energéticos.

- Enfriamiento: el trabajo realizado por el centro de datos genera calor y esto baja el rendimiento de los equipos por ello es necesario que exista un sistema de enfriamiento.
- Control de accesos: la información manejada en el centro de datos es bastante delicada por ello el acceso debe ser restringido solo al personal autorizado.

Y finalmente para su administración y mantenimiento un centro de datos siempre va a necesitar (Figura 13):

- Personal Operativo: es el personal que se encarga de la operación del centro de datos para dar una respuesta inmediata en caso de algún fallo.
- Procesos: creados para seguir una normal o estándar, con la finalidad de obtener una certificación o calificación por pate de una entidad reguladora.
- Sistemas Autónomos: realizan el monitoreo y la generación de reportes a fin de verificar que el centro de datos no falle y cualquier error pueda ser resulta con la brevedad de caso.

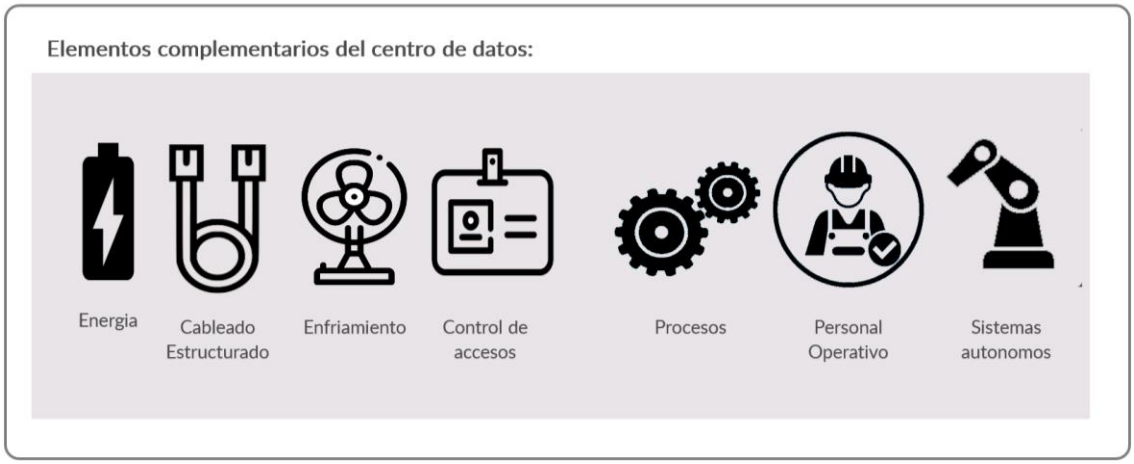

Figura 13 Elementos complementarios del centro de datos.

1.2.2. Arquitectura del centro de datos

Los centros de datos a lo largo del tiempo han sido montados en la arquitectura tradicional 3 tier, sin embargo, los requerimientos actuales han dado nacimiento a la arquitectura Spine leaf, veremos brevemente el detalle técnico de cada una de estas arquitecturas.

1.2.2.1. Tradicional o 3 Tier

Es la arquitectura más usada dentro de los centros de datos debido a su robustez, diseño modular y buen rendimiento. La arquitectura de 3 tier está formada por: la capa de *core* o núcleo, capa de distribución o *aggregation* y capa de acceso o *access*, tal como podemos preciar en la figura 14. A pesar de sus ventajas la arquitectura 3 tier ya no es viable para grandes centros de datos, debido al crecimiento exponencial que han tenido éstos, y el surgimiento del paradigma del *cloud computing*, su baja escalabilidad y eficiencia hacen que un centro de datos con esta arquitectura tenga un alto costo. (Ciscotechblog, 2011).

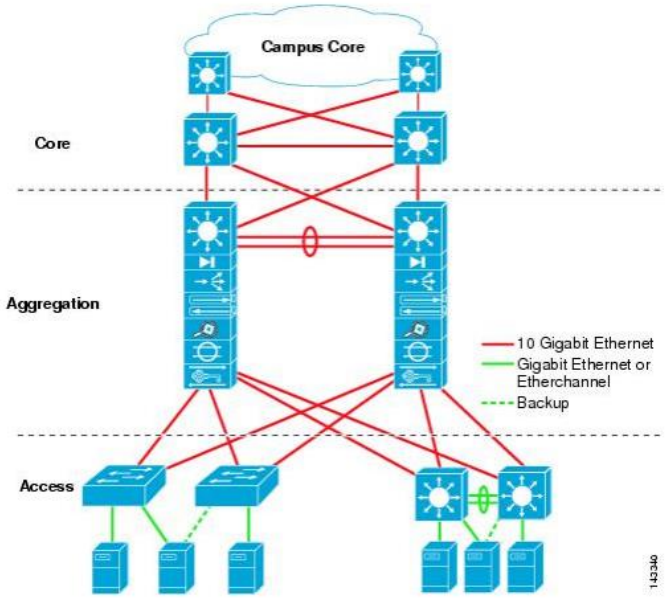

Figura 14 Modelo arquitectura tradicional 3 tier.

Tomado de (Ciscotechblog, 2011).

1.2.2.2. Arquitectura Spine-Leaf

La arquitectura Spine-Leaf es usada en centro de datos con gran necesidad de escalabilidad, cuyo principal objetivo es solucionar los problemas de la arquitectura 3-Tier (Cisco Systems, 2020).

Es una arquitectura de dos niveles tal como se muestra en la figura 15, en donde cada conmutador de nivel inferior (leaf layer) está conectado a cada uno de los conmutadores de nivel superior (spine layer) formando una topología de malla completa. La capa de leaf consta de conmutadores de acceso que se conectan a dispositivos como servidores. La capa Spine, es la columna vertebral de la red y es responsable de interconectar todos los conmutadores de la capa leaf. Cada switch de la capa leaf se conecta a cada switch del Spine formando una malla redundante. La ruta se elige aleatoriamente para que la carga de tráfico se distribuya uniformemente entre los conmutadores de nivel superior. Si uno de los conmutadores de nivel superior fallara, solo degradaría ligeramente el rendimiento en todo el centro de datos (Cisco Systems, 2020).

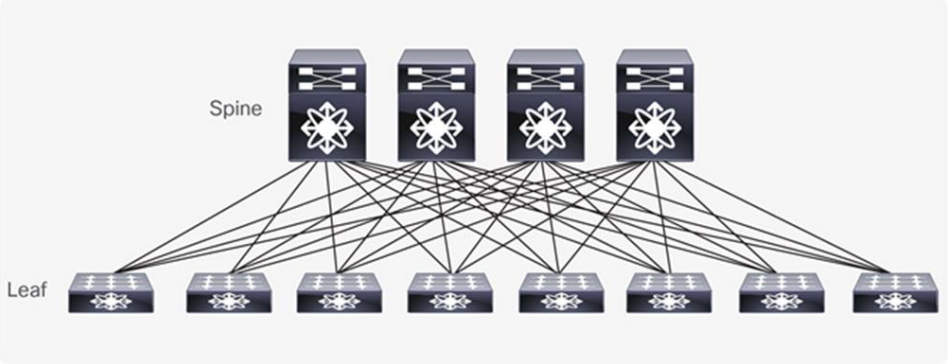

Figura 15 Esquema general de la nueva arquitectura Spine-Leaf.

Tomado de (Cisco Systems, 2020).

1.2.3. Tipos de centro de datos

De acuerdo con la ANSI o Instituto Nacional Estadounidense de Estándares los centros de datos se los puede clasificar por su disponibilidad en, de Tier 1 a Tier 4.

A continuación, se revisará un breve detalle técnico de cada clasificación;

1.2.3.1. Tier 1

Se aplica principalmente en pequeñas y medianas empresas, al no contar con redundancia en su distribución eléctrica y refrigeración, el servicio puede suspenderse de forma planificada o no planificada (Hwaiyu, 2015).

Es necesario paralizar su funcionamiento al menos una vez al año, para su mantenimiento, así se evita pausas más frecuentes y errores de operación o fallas en los componentes de la infraestructura que pudieren causar la paralización del centro de datos. Su disponibilidad es del 99.671%, lo cual se traduce en tiempo fuera de 28,82 horas al año (Pacio, 2014).

1.2.3.2. Tier 2

Es un centro de datos redundante por lo que es menos propenso a interrupciones, ya sean planificadas o no. Su estructura se conforma por los componentes necesario (N) y un sistema contingente por cada elemento de su infraestructura al cual se lo denomina, más uno (+1) por tanto, el modelo es N+1. Cuenta con conexión a una sola línea de distribución eléctrica y de refrigeración. Para su implementación se requiere de 3 a 6 meses y cuenta con suelos elevados, generadores auxiliares (UPS). Para su mantenimiento aún es necesaria la interrupción del servicio. Su disponibilidad es del 99.749%, que corresponde a 22.68 horas al año de interrupción (Pacio, 2014).

1.2.3.3. Tier 3

Un centro de datos tier 3 es redundando y su diseño es N+1 esto permitirá que las operaciones continúen sin detenerse mientras se realizan operaciones de mantenimiento (Hwaiyu, 2015).

Debe contar con suficiente capacidad y doble línea de distribución para todos los componentes, esto permitirá realizar el mantenimiento o las pruebas en una línea mientras otra se ocupa de la totalidad de la carga sin generar interrupciones. El

tiempo de implementación es de 15-20 meses, con una disponibilidad del 99.982%, y son1.57 horas al año de interrupción (Pacio, 2014).

En Ecuador por ejemplo la mayoría de los centro de datos son de Tier 3, un ejemplo de ellos es la CNT EP, que inauguro su centro de datos en la cuidad de Quito, sector La Armenia en febrero de 2016, el mismo se aprecia en la figura 12, mismo que luego de su inauguración obtuvo el certificado ANSI de Tier 3 (CNT EP, 2018).

*Figura 16 C*entro de datos de la CNT.

Tomado de (CNT EP, 2018).

1.2.3.4. Tier 4

Es un centro de datos con capacidad para realizar cualquier actividad planificada o no sin interrupciones. Su funcionalidad tolerante a fallos y su conexión a variadas líneas de distribución eléctrica y refrigeración, permite a la infraestructura continuar operando, ante cualquier evento crítico no planificado (Pacio, 2014).

Tiene múltiples componentes redundantes, y para tener una esta certificación el centro de datos contar cumplir el modelo 2(N+1), esto es, dos líneas de suministro eléctrico de redundancia por cada N+1 (Hwaiyu, 2015).

Este sistema permite realizar el mantenimiento sin interrumpir el servicio de tecnologías críticos y es capaz de afrontar imprevistos. Puede fallar únicamente cuando ocurren al mismo tiempo un corte de energía y el error de dos o más grupos electrógenos en cada una de las líneas de suministro. Su implementación tarda 20 meses, con una disponibilidad del 99.995%, lo cual son 52.56 minutos al año fuera de servicio (Pacio, 2014).

1.3. Sistemas Embebidos

Los sistemas embebidos son parte del *hardware* de las redes de datos y a pesar de que no son muy nombrados están presentes en la mayoría de los dispositivos eléctricos y electrónicos. En esta sección veremos los conceptos básicos hasta ampliar la información de las Raspberry PI las cuales son el objeto de estudio del presente proyecto.

1.3.1. Introducción a los sistemas embebidos

Un sistema embebido puede ser una placa o un controlador lógico programable (PLC o programmable *logical controller*) cuyo sistema está diseñado para ejecutar instrucciones básicas en tiempo real. Al contrario de las computadoras que por su diseño les permite cubrir varias necesidades y ejecutar instrucciones generales, estos sistemas fueron diseñados para satisfacer necesidades específicas.

La mayoría de los componentes de este sistema se encuentran incorporados en la placa base (audio, módem, tarjeta de video), por ello los dispositivos resultantes difieren mucho de una computadora. Algunos ejemplos de sistemas embebidos son: dispositivos del sistema de control de una fotocopiadora, lectores de control de acceso, la electrónica que controla una máquina expendedora o simplemente un taxímetro (Semanticwebbuilder, 2020).

El mercado de desarrollo de los sistemas embebido se ha disparado en los últimos años debido a la necesidad del aprendizaje de la tecnología y con la creciente demanda de los sistemas inteligentes basado en domótica o el internet de las cosas (IoT, *internet of things*).

Una de las primeras placas consideradas como embebida en salir al mercado fue el Arduino que inicio en el año 2005, presentando desde entonces sus placas con microcontroladores dentro de los cuales podemos detallar el Arduino UNO, una de sus placas más emblemáticas que se muestra en la figura 17, diseñado en arquitectura Harvard o AVR.

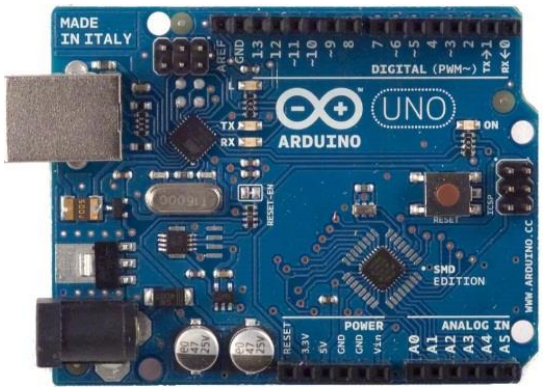

Figura 17 El Arduino uno, es un sistema Embebido de arquitectura AVR.

Tomado de (Arduino, 2020).

En la carrera por este nicho de mercado Google Inc., y la fundación Raspberry lanzaron sus placas con procesadores RISC (*Reduced Instructions Set computer* u ordenador con conjunto reducido de instrucciones) y con ello desarrollando la arquitectura Advanced RISC Machine (ARM). En 2009 Google lanzaron su primera placa denominada ODROID, actualmente su modelo símbolo más reciente es el N2 que se muestra en la figura 18.

Figura 18: Google Inc., y su placa Odroid con arquitectura ARM.

Tomado de (Odroid, 2019).

Lo revolucionario de la tecnología ARM, es que permite el uso de *software* libre para el desarrollo de las aplicaciones y a pesar de que Google dio el primer paso en este campo, la fundación Raspberry en el 2012 lanzo la Raspberry PI que dio mejoras al funcionamiento de sus placas desarrollando un S.O. específico para correr con estas placas llamado Raspberry PI OS. En la figura 19 podemos apreciar la Raspberry PI 1 lanzada en 2012.

Figura 19 Placa Raspberry PI 1 de arquitectura ARM.

Tomado de (Raspberry, 2020).

Las Raspberry PI es la placa objeto de este trabajo por lo cual se detallará mayor información a continuación.

1.3.2. Raspberry PI

La fundación Raspberry se creó en el año 2012 y su objetivo fundamental fue el lanzamiento de las placas Raspberry, cuyo objetivo es el aprendizaje de la computación mediante la experimentación de placas reducidas, que en un inicio se creó para que niños de escuela y colegio aprendieran los fundamentos básicos de la computación sin dañar los computadores.

El proyecto tuvo buena acogida, así como sus resultados fueron muy llamativos que pronto empresas y aficionados informáticos empezaron a tomar mayor interés en las placas y sus prestaciones, y dentro las cuales la que más sorprender en su alto rendimiento a un costo eléctrico bastante bajo ya que el consumo de una placa Raspberry puede ser de tan solo unos 4 W/h a 15 W/h (dependiendo del modelo).

Actualmente las placas Raspberry son utilizadas para una gran cantidad de proyectos escolares tal como lo explica la fundación Raspberry en su página web mientras que usuarios más avanzados han realizado experimentaciones con las placas para crear *cluster* de alta capacidad que pueda emular o funcionar como servidores.

1.3.2.1. Características técnicas del Raspberry PI

El principal componente de la Raspberry PI es su procesador ARM, desarrollado en arquitectura RISC (*Reduced Instrution Set Computer*) u ordenador con conjunto reducido de instrucciones. ARM es una suma de instrucciones, por lo general de 32 bits con gran aplicación en el mercado. Su sencillez lo hace ideal para aplicaciones de baja potencia.

Las especificaciones técnicas de cada Raspberry varían según su modelo y diseño dentro de los cuales podemos encontrar los modelos A y B, cuya principal diferencia radica en que el modelo A por lo general sacrifica muchas de sus funcionalidades para reducir su costo.

Las especificaciones generales o permanentes en la mayoría de las placas son:

-La existencia de circuito integrado Broadcom *BCM2835 system-on-chip (SoC).*

-En todos los modelos la memoria utilizada es la RAM, o memoria de acceso aleatorio que permite la lectura y escritura de datos, misma que es volátil. La cantidad de memoria RAM incorporada varía según el diseño y modelo desde 256 MB hasta 8GB

-Uno de los puertos estándares que se ha incorporado en cada modelo es el *GPIO* (*general-purpose input output*), mismo que puede ser utilizado para energizar la Raspberry PI o conectarla con otro *hardware*.

-También cuenta con puerto de salida de video desde el tradicional puerto HDMI hasta la nueva versión del mini-HDMI

-Los puertos DSI (*Display Serial Interface*) incluido en los primeros modelos permite la conexión al sistema de pantalla digital, y adicional cuenta con un puerto CSI (*Camera Serial Interface*) que permite conectar con un sistema de cámara.

-La alimentación o energización de la placa se lo realiza por medio de un conector que micro-USB tipo B o C según modelo.

-En la parte reversa de la placa se encuentra la ranura para tarjetas SD (Secure Digital) hasta las micro SD, para agregar una tarjeta de memoria SD o micro SD no volátil en donde se almacena el SO y los datos con los cuales se está trabajando.

1.3.2.2. Modelos

Desde su primer modelo las especificaciones técnicas de cada Raspberry han ido mejorando sus prestaciones teniendo hasta la actualidad 4 diseños (Raspberry PI 1 a la 4) y con los modelos A y B, y se incorporan mejoras con los modelos plus (+), los cuales se describen brevemente a continuación.

1.3.2.2.1. PI 1

El primer modelo de Raspberry comercializado, a pesar de tener características básicas no deja de ser poderoso. Se los fabricó en los modelos A y B, con diferencias marcadas en cada uno de ellos, los cuales se explican en la tabla 1:

Tabla 1 *Caracteristicas de las Raspberry PI 1 Modelos A y B.*

Especificaciones	Raspberry PI 1 Modelo A	Raspberry PI 1 Modelo B	Raspberry PI 1 Modelo B+
SoC	Broadcom BCM2835		
CPU	ARM 1176JZF-S a 700 MHz		
Juego Instrucciones	RISC de 32 bits		
GPU	Broadcom VideoCore IV, OpenGL ES 2.0		
Memoria	256 MB (compartidos con la GPU)	512 MiB (compartidos con la GPU)	
Puertos USB 2.0	1	2 (HUB Integrado)	4
Entrada Video	Conector CSI, permite conectar un cámara de video		
Salida Video	Conector RCA (PAL y NTSC), HDMI, Interfaz DSI para panel LCD		
Salida Audio	Jack de 3.5 mm, HDMI		
Almacenamiento	SD / MMC / ranura para SDIO		
Conectividad de red	No tiene	10/100 Ethernet (RJ-45) vía hub USB	
Consumo energético	500 mA (2.5 W)	700 mA (3.5 W)	600 mA (3.0 W)
Fuente de alimentación	5 V vía Micro USB o puerto GPIO		
Sistemas Operativos	GNU/Linux: Raspberry PI OS,, Arch Linux (Arch Linux ARM), Slackware Linux, SUSE Linux Enterprise Server for ARM, RISC OS		

Tomado de (Raspberry, 2020)

En la figura 20 podemos apreciar la primera placa Raspberry lanzada por la fundación.

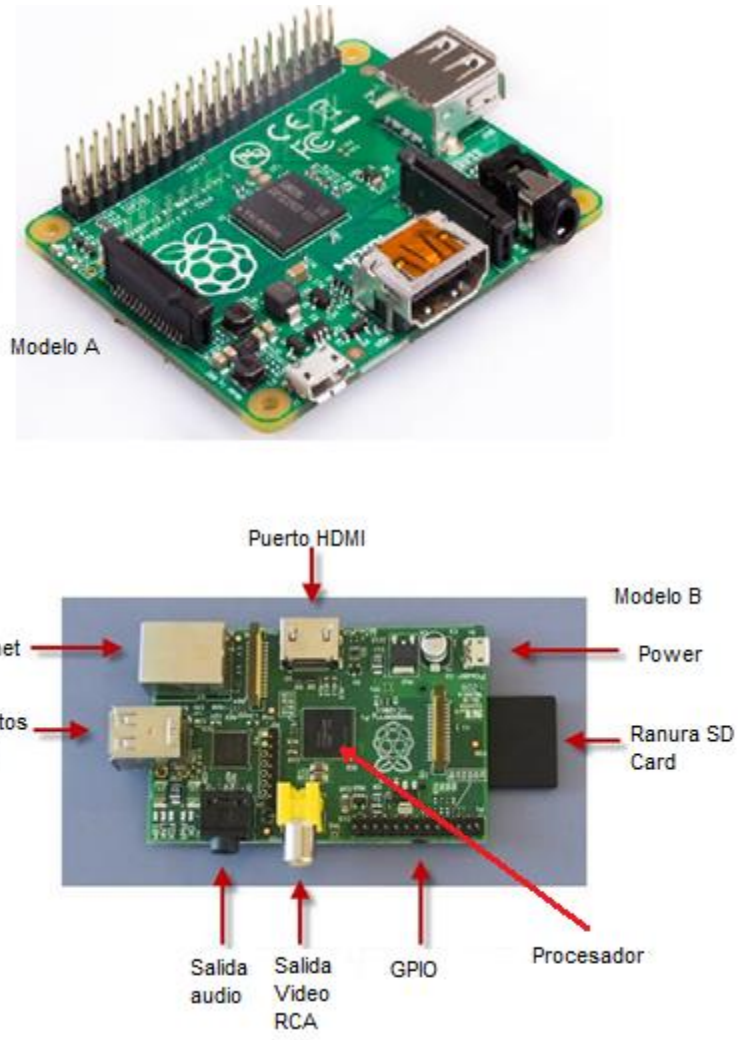

Figura 20 Raspberry PI 1 Modelos A y B y sus componentes.

Adaptado de (Raspberry, 2020).

1.3.2.2.2. PI 2

Este modelo es más pequeño, con algunas mejoras:

Se incorpora la ranura para micro SD dejando atrás la ranura para la tarjeta SD, el usuario lo único que debe hacer es empujar la tarjeta para conectarla y desconectarla de la placa, se mejora el audio y baja su costo de adquisición. Y se presenta únicamente en el modelo B. Las especificaciones técnicas se detallan en la tabla 2.

Tabla 2 *Características Raspberry PI2 Modelo B.*

Especificaciones	Raspberry PI 2 Modelo B
SoC	Broadcom BCM2836
CPU	900 MHz quad-core ARM Cortex A7
Juego Instrucciones	RISC de 32 bits
GPU	Broadcom VideoCore IV, OpenGL ES 2.0, MPEG-2 y VC-1 , 1080p30 H.264/MPEG-4 AVC
Memoria	1 GB (compartidos con la GPU)
Puertos USB 2.0	4
Entrada Video	Conector CSI, permite conectar un cámara de video
Salida Video	Conector RCA (PAL y NTSC), HDMI, Interfaz DSI para panel LCD
Salida Audio	Jack de 3.5 mm, HDMI
Almacenamiento	SD / MMC / ranura para SDIO
Conectividad de red	No tiene
Consumo energético	800 mA (4.0 W)
Fuente de alimentación	5 V vía Micro USB o puerto GPIO
Sistemas Operativos	GNU/Linux: Raspberry PI OS, Fedora (Pidora), Arch Linux (Arch Linux ARM), Slackware Linux, SUSE Linux Enterprise Server for ARM, RISC OS

Tomado de (Raspberry, 2020)

En la figura 21 se puede ver la Raspberry PI 2 modelo B:

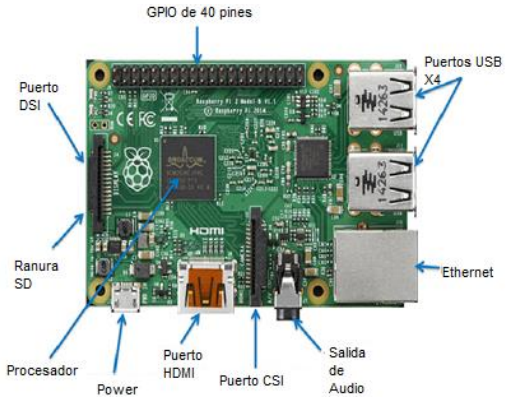

Figura 21 Placa Raspberry PI 2 y sus componentes.

Adaptado de (Raspberry, 2020).

1.3.2.2.3. PI 3

Las Raspberry PI 3 ha tenido mejoras considerables en sus prestaciones con respecto a su predecesor como es la incorporación de un puerto RJ45 y tarjeta para conexión WiFi con estándar 802.11 b/g/n para conectividad de red, un Procesador *Quad Core* más potente de 1.2 GHz. Se presento en los modelos A y B, para los cuales se detallas las especificaciones técnicas en la tabla 3.

Tabla 3 *Características Raspberry PI 3 modelos A y B.*

Especificaciones	Raspberry PI 3 Modelo B	Raspberry PI 3 Modelo B+	Raspberry PI 3 Modelo A+
SoC	Broadcom BCM2837	Broadcom BCM2837B0	
CPU	Broadcom Quad Core 1.2GHz 64bit CPU	Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz	
Juego Instrucciones	RISC de 64bits		
GPU	Broadcom VideoCore IV, OpenGL ES 2.0, MPEG-2 y VC-1 , 1080p30 H.264/MPEG-4 AVC		
Memoria	1 GB (compartidos con la GPU)		512 MiB (compartidos con GPU)
Puertos USB 2.0	4		1
Entrada Video	Conector CSI, permite conectar un cámara de video		
Salida Video	Conector RCA (PAL y NTSC), HDMI, Interfaz DSI para panel LCD		
Salida Audio	Jack de 3.5 mm, HDMI		
Almacenamiento	Ranura para Micro SD		
Conectividad de red	Puerto RJ-45 (Ethernet) de 10/100 Wi-Fi 802.11bgn Bluetooth 4.1	Puerto Ethernet de 10/100/1000Mbps WiFi 802.11 ab/g/n/ac, Bluetooth 4.2, BLE	2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2/BLE
Consumo energético	800 mA (4.0 W)		
Fuente de energía	5 V vía Micro USB o puerto GPIO		
Sistemas Operativos	GNU/Linux: Raspberry PI OS, Fedora (Pidora), Arch Linux (Arch Linux ARM), Slackware Linux, SUSE Linux Enterprise Server for ARM		

Tomado de (Raspberry, 2020).

En la figura 22 se muestra el modelo B+ de la Raspberry PI 3

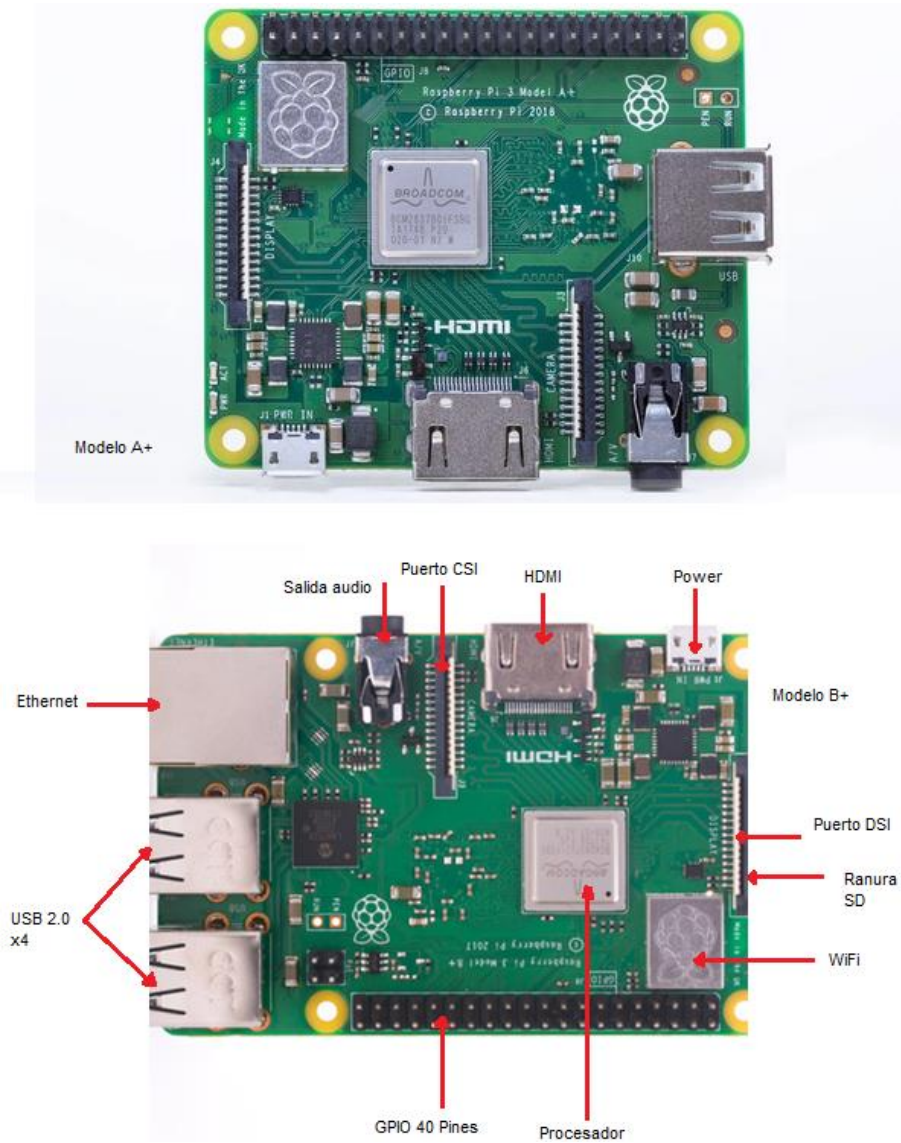

Figura 22 Raspberry PI 3 Modelo A+ y B+ y sus componentes.

Adaptado de (Raspberry, 2020).

1.3.2.2.4. PI 4

La tarjeta más reciente desarrollada por la fundación Raspberry, hasta el momento se ha presentado únicamente el modelo B, con mejoras en las prestaciones como son el procesador de 1.5 GHz Quad Core, la memoria RAM directamente presenta

4 versiones de 1, 2, 4 y 8 GB, se incorporan los puertos USB 3.0 y 2 puertos mini-HDMI con capacidad de transmisión en calidad 4K. Las especificaciones técnicas del modelo B se detallan en la tabla 4.

Tabla 4 *Características técnicas Raspberry PI 4.*

Especificaciones	Raspberry PI 4 Modelo B
SoC	Broadcom BCM2711
CPU	Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
Juego Instrucciones	RISC de 64 bits
GPU	Broadcom VideoCore VI, OpenGL ES 3.0, 1080p30 H.264/MPEG-4 AVC, 4kp60 H.265
Memoria	1 GB, 2GB, 4BG, 8GB (compartidos con la GPU)
Puertos USB 2.0	2
Puertos USB 3.0	2
Entrada Video	Conector CSI, permite conectar un cámara de video
Salida Video	Conector RCA (PAL y NTSC), microHDMI hasta calidad 4k60, Interfaz DSI para panel LCD
Salida Audio	Jack de 3.5 mm, 2 puertos microHDMI
Almacenamiento	Micro SD
Conectividad de red	Puerto RJ-45 10/100/1000Mbps Wi-Fi 802.11ac de doble banda Bluetooth 5.0 BLE
Consumo energético	Máximo 3A (15.3 W)
Fuente de alimentación	5 V vía Micro USB o puerto GPIO, Alimentación a través de Ethernet (PoE) habilitada (requiere un PoE HAT por separado)
Sistemas Operativos	Raspberry PI OS

Tomado de (Raspberry, 2020).

En la figura 23 se muestra el último modelo de Raspberry, la PI 4, disponible en las versiones de 1, 2, 4 y 8 GB de RAM:

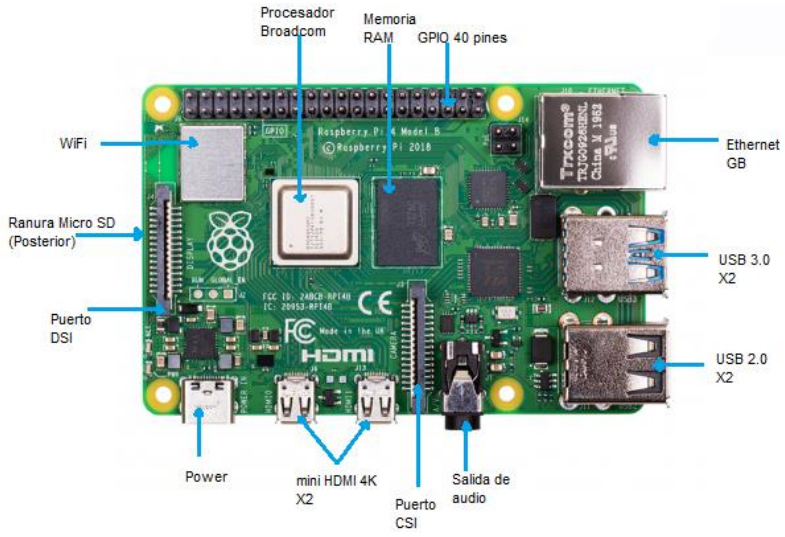

Figura 23 Raspberry PI4 y sus componentes.

Adaptado de (Raspberry, 2020).

1.4. El *software* para sistemas embebidos

Esta sección se centrará específicamente en el *software* libre y las distribuciones compatibles con las placas Raspberry recopilando sobre información sobre el SO Raspberry PI OS.

1.4.1. Kernel Linux

El Kernel Linux es el núcleo o modelo base, que ha sido tomado para la creación o desarrollo de las distribuciones Linux, las cuales son utilizadas para la creación de proyectos de código abierto bajo licencia GNU. Linux comenzó a principios de los 90 como un simple proyecto realizado por Linus Torvalds, un estudiante de la Universidad de Helsinki. Actualmente este sistema se ha difundido por todo el mundo, tomando fuerza sobre todo en empresas y corporaciones, gracias a su facilidad para el desarrollo de *software* orientado a las necesidades (Catalina, 2006).

En la actualidad cuando se habla de Linux inmediatamente se relaciona con las distribuciones libres que las empresas pueden desarrollar para satisfaces sus necesidades. Sin embargo, para que todo esto pueda ser posible se debe entender

la estructura básica y componente de un sistema Linux, mismos que se pueden apreciar en la figura 24 (Silderchatz, 2006).

Figura 24 Componentes del sistema Linux.

Adaptado de (Silderchatz, 2006).

Cuando se arranca un sistema hay varias cosas que ocurren antes de la carga del sistema operativo (SO). El proceso de arranque se realiza en las siguientes seis etapas las cuales son detalladas a continuación:

-Sistema básico de entrada / salida (BIOS)
-Registro de arranque maestro (MBR)
-GRand Unified Bootloader (GRUB)
-Kernel de Linux
-Nivel de ejecución
-BIOS (entrada y salida)

1.4.1.1. BIOS

El BIOS es un *firmware* que realiza la inicialización del *hardware* cuando el sistema está encendido. El BIOS inicializará y probará el *hardware* del sistema. El BIOS contiene suficiente información para cargar un controlador para el teclado.

Dentro de la configuración del BIOS (CMOS) hay un orden del dispositivo de arranque. El control se pasará a cada dispositivo de arranque en orden de prioridad.

Se comprueba cada dispositivo para el MBR. Cuando se encuentra el BIOS pasa el control a los dispositivos MBR.

1.4.1.2. MBR

El (MBR) no es técnicamente necesario en algunos casos, pero el MBR es un sector de arranque colocado al principio de un disco fijo o extraíble. El MBR contiene los datos para que el sistema conozca las particiones de un disco. También hay código ejecutable que es un cargador (gestor de arranque). No todas las particiones contienen un gestor de arranque y si no se encuentra un gestor de arranque, entonces no se cargará un sistema operativo. El MBR existirá sin el código ejecutable o cargador de inicio.

Cuando se instala un sistema operativo, el código de ejecución de un gestor de arranque se colocará en el MBR. Sin sistema operativo, el sistema no se iniciará correctamente. El MBR existe sin el cargador de arranque.

1.4.1.3. GRUB

La primera etapa del cargador de arranque se coloca en el MBR que llama al gestor de arranque principal que está instalado en una partición de arranque (/boot). GRUB, si está configurado para hacerlo, mostrará un menú y le permitirá pasar el control a un Kernel específico. Esta característica permite que un sistema tenga varios sistemas operativos instalados en él. El archivo de configuración GRUB se almacena en “/boot/grub/grub.conf”.

Cada entrada puede pasar opciones al Kernel que pueden cargar el sistema operativo o realizar diagnósticos. Una vez que se selecciona un elemento de menú o se acepta el valor predeterminado, el control se pasa al kernel especificado.

1.4.1.4. Núcleo o Kernel

Cuando el Kernel tiene control, el Disco de RAM Inicial (initrd) se utiliza como un sistema de archivos raíz temporal hasta que se arranque el Kernel. Una vez

arrancado, el Kernel monta el sistema de archivos raíz real. La ubicación del sistema de archivos ROOT se pasa desde GRUB tiene una opción root =. El núcleo como controladores incluidos para acceder a las particiones del disco duro, así como algún otro *hardware*. Los controladores compilados en el Kernel le permiten montar otros sistemas de archivos.

1.4.1.5. Nivel de ejecución

Los niveles de ejecución son los siguientes:
0 – Detener
1 – Modo de un solo usuario
2 – Multiusuario, sin NFS
3 – Modo multiusuario completo
4 – Sin usar
5 – X11
6 – Reiniciar
Los programas configurados para arrancar automáticamente al arrancar se pueden encontrar en uno de los siguientes:

Ejecutar nivel 0 – /etc/rc.d/rc0.d/
Ejecutar nivel 1 – /etc/rc.d/rc1.d/
Ejecutar nivel 2 – /etc/rc.d/rc2.d/
Ejecutar nivel 3 – /etc/rc.d/rc3.d/
Ejecutar nivel 4 – /etc/rc.d/rc4.d/
Ejecutar nivel 5 – /etc/rc.d/rc5.d/
Ejecutar nivel 6 – /etc/rc.d/rc6.d/

1.4.2. Raspberry PI OS

Cada *hardware* para su funcionamiento y operación necesariamente requiere la existencia de un SO, controlador que permita a cada componente cumplir con las funciones para la que fue diseñado, en el caso de los sistemas embebidos para

placas Raspberry PI, la fundación Raspberry recomienda el uso del S.O Raspberry PI OS del cual se revisará su detalle técnico.

1.4.2.1. Características de SO Raspberry PI OS

El Raspberry PI OS cuyo nombre anterior era Raspbian es una distribución basada en Debian bajo licenciamiento GNU/Linux por lo cual es *software* libre y está encaminado a la enseñanza de la informática. Fue lanzado inicialmente en junio de 2012 y desde entonces se ha ido mejorando hasta declararlo en 2015,como el S.O. oficial para las placas Raspberry PI. Hay varias adaptaciones del Raspberry PI OS, siendo la más reciente la Buster (Raspbian, s.f.).

Conjuntamente con el paquete de instalación, el Raspberry PI OS trae preinstalado los programas básicos para la educación, programación y uso general, como por ejemplo Python, Scratch, Sonic Pi, Java, etc (Raspbian, s.f.).

Técnicamente el sistema operativo es un port no oficial de Debian ARMhf (ARM hard float) para el procesador (CPU) de Raspberry PI, con soporte optimizado para cálculos en coma flotante por *hardware*, lo que permite dar más rendimiento en caso de que el procesador sea utilizado programas que realicen cálculos con números reales y juegos con gráficos no tan complejos. (Raspbian, s.f.).

1.4.2.2. Componentes del Raspberry PI OS

Referencias de Debian: Es una guía que brinda una explicación detallada de la distribución Debian Linux y como los programadores/desarrolladores pueden aportar a su desarrollo (Raspbian, s.f.).

Administrador de archivos: es un gestor predeterminado de fichero rápido y robusto, el cual ofrece varias funciones como la navegación por pestañas, con un uso reducido de recursos.

La gran mayoría de herramientas están desarrolladas para el entorno gráfico, y allí es donde se puede ver los iconos en el escritorio, donde se puede encontrar las herramientas como son el LXTerminal, para la ejecución del terminal; el Leafpad o

editor simple de texto que permite la escritura de notas rápidas; Xarchiver es un aplicativo para la descompresión de archivos .zip; QjackCtl que es una herramienta grafica para controlar la configuración del sonido; y las herramientas IDLE, Scrath y Wolfram que son herramientas para el aprendizaje y desarrollo de la programación.

Así también dispone de herramientas graficas para la visualización de imágenes, como el GpicView; Xpdf para visualización de archivos PDF, y los navegadores más populares en Raspberry como son el Dillo y NetSurf.

El sistema además posee un administrador de tareas que permite administrar los recursos del sistema, las aplicaciones, y así mismo el usuario puede modificar el entorno gráfico, cambiando el fondo de escritorio, temas o simplemente el color y la apariencia de las herramientas del sistema, así como también su funcionamiento.

1.4.3. Instaladores y herramientas para Raspberry PI

Entre los instaladores y herramientas existe plataformas o programas que están avalados y recomendados directamente por la fundación Raspberry.org. Otros que desarrollan la compatibilidad necesaria para poder trabajar con las Raspberry, listamos a continuación:

1.4.3.1. Instaladores

A continuación, se presenta los instaladores y sistemas operativos que funcionan sobre Raspberry PI.

New Out Of the Box Software (NOOBS)**:**

NOOBS es un instalador fácil del sistema operativo que contiene Raspberry PI OS y LibreELEC. Facilita además una selección de los sistemas operativos alternativos que pueden descargarse de Internet y se instalan (Raspberry.org, 2020).

Balena Etcher:

Existen varios programas y herramientas que no permite instalar el SO en la micro SD, sin embargo, directamente la Fundación Raspberry recomienda el *software* Balena Etcher.

Esta herramienta gráfica que permite formatear una micro SD e instalar el SO, estando disponible para Windows, Mac OS y Linux, y es la opción más fácil para la mayoría de los usuarios ya permite la escritura de imágenes directamente desde el archivo zip, sin necesidad de descomprimir.

1.4.3.2. Virtualización

La virtualización se ha convertido en una herramienta fundamental para el desarrollo y crecimiento tecnológico ya que permite la optimización de recursos y el ágil despliegue de aplicación a través de las redes, además de permitir el ahorro de espacio físico.

Para lograr la virtualización en un inicio se crearon hipervisores que controlan las máquinas virtuales que toman recursos de un equipo físico para crear una simulación y ejecutar en él un sistema operativo y permitir el despliegue de aplicaciones.

Actualmente para optimizar los recursos de un equipo se ha desarrollado otro tipo de virtualización como son los contenedores, mismos que mejorar el despliegue de aplicaciones ya que las ejecutan sin necesidad de tener un S.O. intermedio.

1.4.3.2.1. Hipervisores: Xvisor

Los hipervisores son programas que permiten la creación de máquinas virtuales para levantar servicios en ellas, por con siguiente es necesario la adquisición de licencias tanto para el sistema operativo como para los programas.

Además, Xvisor es un hipervisor de código abierto, que tiene como objetivo proporcionar una solución de virtualización monolítica, liviana, portátil y flexible.

Proporciona una solución de virtualización de huella de alto rendimiento y poca memoria para ARMv5, ARMv6, ARMv7a, ARMv7a-ve, ARMv8a, x86_64 y otras

arquitecturas de CPU. En comparación con otros hipervisores ARM, es uno de los pocos hipervisores que proporciona soporte para CPU ARM que no tienen extensiones de virtualización ARM.

El código fuente de Xvisor es altamente portátil y se puede portar fácilmente a la mayoría de las arquitecturas de 32 o 64 bits de uso general, siempre que tengan una unidad de gestión de memoria paginada (PMMU) y un puerto del compilador GNU C (GCC).

Xvisor admite principalmente la virtualización completa, por lo tanto, admite una amplia gama de sistemas operativos invitados no modificados. La paravirtualización (virtualización sin SO anfitrión.) es opcional para Xvisor y se admitirá de manera independiente de la arquitectura (como los dispositivos *VirtIO PCI / MMIO*) para garantizar que no haya cambios en el SO invitado para usar la paravirtualización, tal como se muestra en la figura 25.

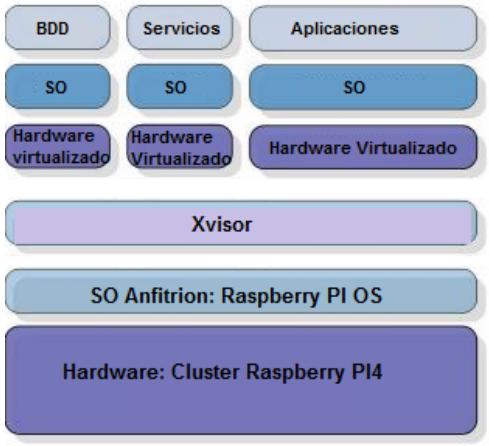

Figura 25 Modelo de XVisor en *cluster* Raspberry PI.

Adaptado de (Xvisor, 2020).

1.4.3.2.2. Contenedores en *Docker*

Los contenedores son programas que agilitan el despliegue de programas, directamente utilizando los recursos de los equipos sin necesidad de crear un SO controlador.

Docker cuyo logo podemos apreciar en la figura 26, es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de *software*, proporcionando una capa adicional de abstracción y automatización de virtualización de aplicaciones en diversos sistemas operativos (Docker, s.f.).

Figura 26 Logo Docker.

Tomado de (Docker, s.f.).

Docker evita la sobrecarga de iniciar y mantener máquinas virtuales, a través del uso de características de aislamiento de recursos del kernel Linux, tales como cgroups y namespaces (espacios de nombres) para permitir que "contenedores" independientes se ejecuten dentro de una sola instancia de Linux, (Docker, s.f.).

Docker permite la virtualización de aplicaciones sin necesidad de crear una máquina virtual dentro del sistema. En la figura 27 se muestra las diferencias entre el *Docker* y la virtualización (Docker, s.f.).

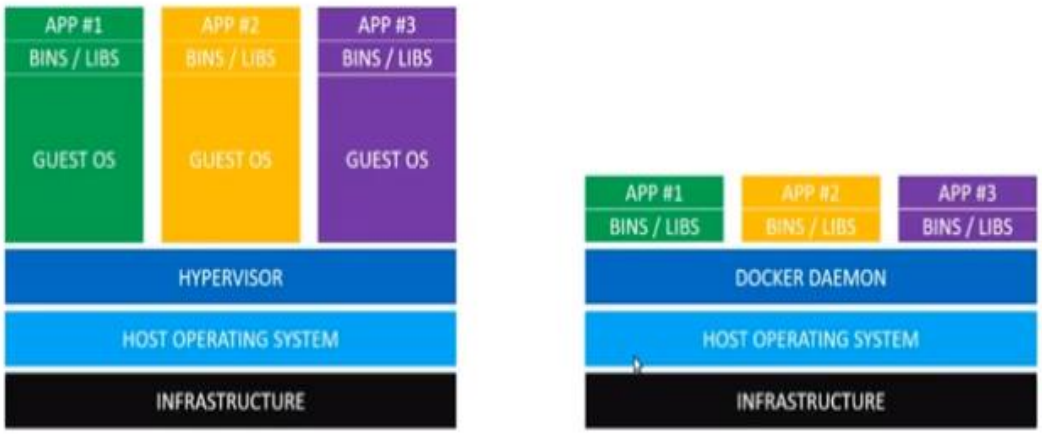

Figura 27 Diferencias entre Hypervisor y *Docker.*
Tomado de (Docker, s.f.).

1.4.3.2.3. Hipervisores vs Contenedores

Ambos tipos de virtualización están diseñados con el propósito de optimar los recursos físicos de los equipos y reducir la necesidad de adquirir nuevos servidores. En la tabla 5 se detalla las ventajas de cada uno de ellos:

Tabla 5 Hipervisores vs Contenedores.

Tecnología	Ventajas	Desventajas
Hypervisores	-Gestión y aprovisionamiento más fácil y rápido -Uso más eficiente del *hardware*, se utiliza los recursos de éste. -Se recomienda su uso para mantener consistencia y compatibilidad en entornos existentes. -Pruebas simplificadas que permite dedicar un servidor de pruebas para así evitar daños en los ambientes de producción.	-Si no se utiliza *software* libre cada SO y sus aplicaciones requiere licenciamiento -El alojamiento de otro SO, consume mayor cantidad de recursos. -El arranque de una máquina virtual toma mucho tiempo y recursos. -Las máquinas virtuales ocupan mucho espacio de almacenamiento y no son portátiles
Contenedores	-Desarrollados en *software* libre, por tanto, su uso y desarrollo puede o no tener costos adicionales. -No requiere de un SO para lanzar las aplicaciones. -Los contenedores son livianos . Los contenedores usan solo los binarios y bibliotecas que necesitan. -Los contenedores son portátiles . -Un contenedor arranca mucho más rápido que las máquinas virtuales.	-En general están dedicados para el desarrollo de microservicios. -Acoplar o migrar los servicios existentes puede ser una tarea muy compleja, ya que no todos lo servicios y aplicaciones son compatibles con los contenedores. -Todos los cambios generados en el contenedor debe ser guardados en imágenes para que no se pierdan. - Al ser *software* libre no se tiene mucho soporte ni expertos en el desarrollo de este tipo de tecnología.

1.4.3.3. Portainer

Portainer es una herramienta de código abierto y *software* libre, que permite la gestión y administración local de contenedores *Docker* mediante la interfaz web que

se puede apreciar en la figura 28. Tiene un sin número de funciones, pero entre las principales se puede mencionar las siguientes (Portainer, 2020):

- Vista de un *cluster swarm*
- Gestión de contenedores *Docker*
- Acceso a las consolas de los contenedores
- Gestion de imágenes *Docker*
- Etiquetar y subir imágenes *Docker*
- Gestionar volúmenes de *Docker*
- Navegar por los eventos de *Docker*
- Preconfigurar templates de contenedores

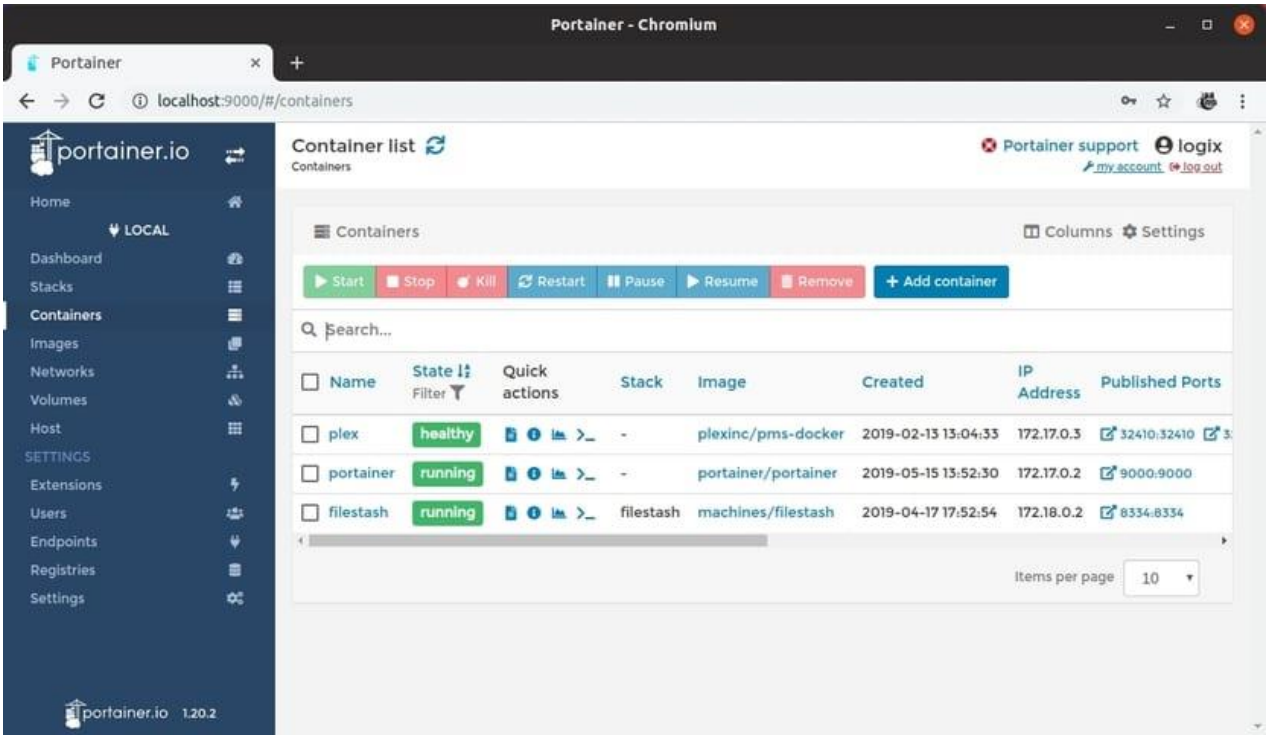

Figura 28: Pantalla de gestión de Contenedores.

1.4.3.4. Open Media Vault

Open media vault es una herramienta que permite la creación de almacenamiento conectado a la red, Debian Linux, por tanto es una herramienta de código abierto y

software libre, que utiliza servicios como *secure shell* (SSH), *file transfer protocol* (FTP), *server message block* (SMB), el cual ha sido renombrado como *common internet file system* (CIFS), servidor de medios *digital audio access protocol* (DAAP), RSync, BitTorrent, etc (Open Media Vault, 2020).

Open media vault está diseñado principalmente para usarse en pequeños negocios, ya que es una solución simple y fácil de usar debido a que la instalación y configuración de un dispositivo de almacenamiento en red o *network attached storage* (NAS) no requiere conocimiento profundo del tema, ya que puede ser administrado mediante la herramienta grafica de su interfaz web como se muestra en la figura 29 (Open Media Vault, 2020).

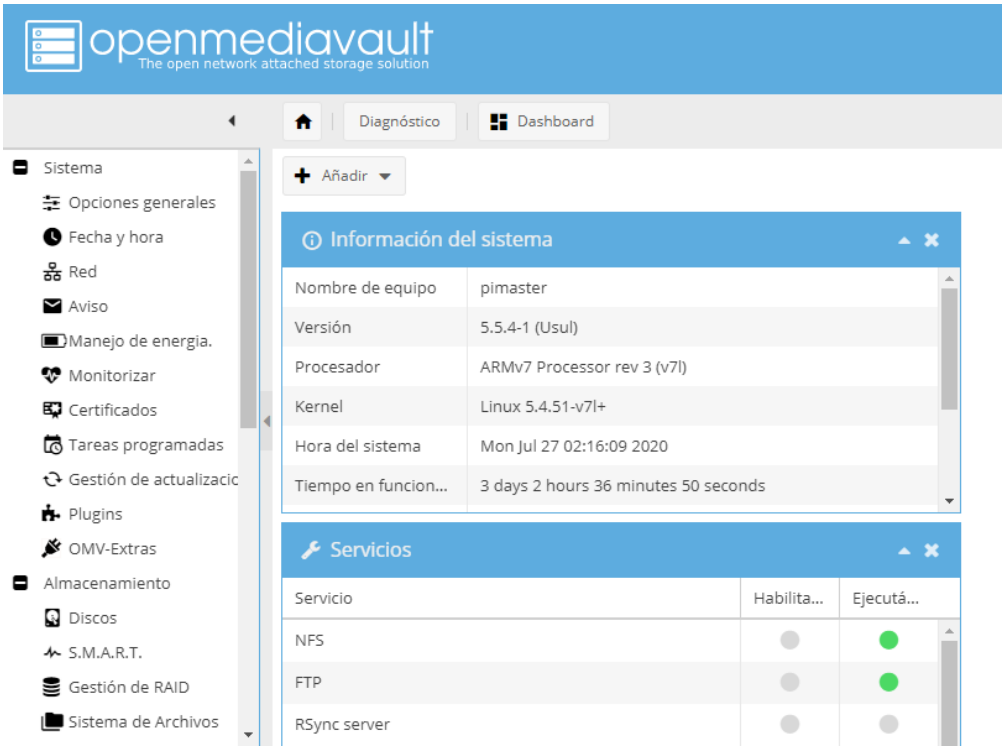

Figura 29: Pantalla principal de Open Media Vault.

Tomado de (Open Media Vault, 2020).

1.4.3.5. Samba

El proyecto Samba o simplemente Samba, es *software* libre que opera bajo la Licencia Pública General de GNU, que desde su implementación en 1992 ha permitido la incorporación de servicios de impresión y archivos seguros, estables y rápidos mediante el uso de protocolos SMB / CIFS, multiplataforma (Samba, 2020).

A pesar de que Samba es multiplataforma su uso más frecuente se da como parte importante de los servidores y escritorios Linux/Unix, pudiendo funcionar como un controlador de dominio (Samba, 2020).

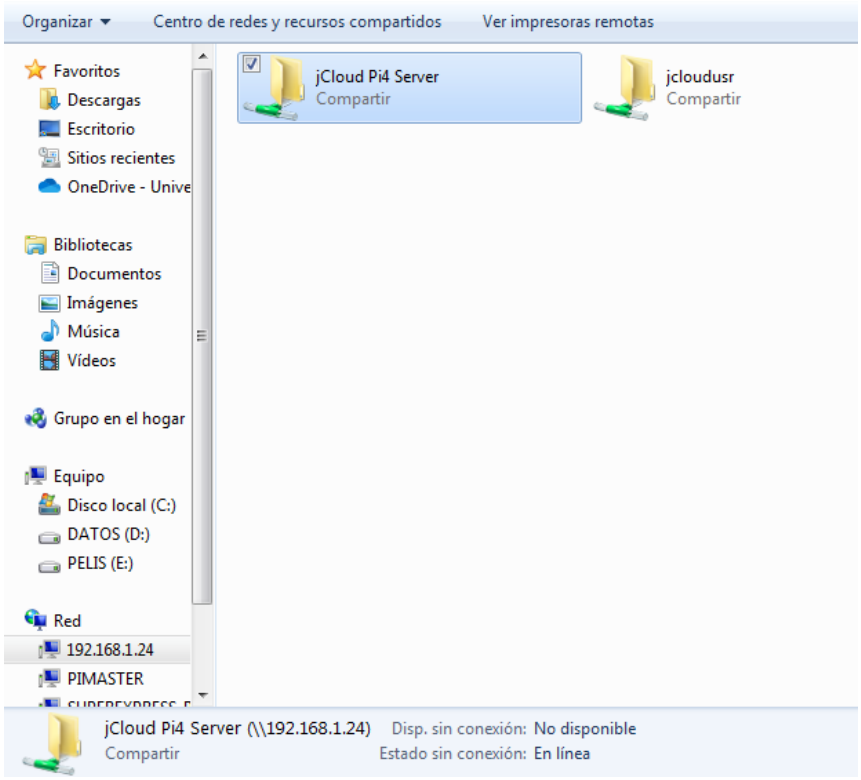

Figura 30 El servicio Samba convirtiendo un disco duro externo en NAS.

1.4.3.6. *Web services*

El *web service* permite la publicación y acceso d un servicio o página web a Internet, siendo esta una interfaz mediante la que (o aplicaciones) se comunican entre sí. Se caracteriza principalmente por ser multiplataforma y ser distribuida. El uso de un *web*

service consiste en que un cliente manda una solicitud a un servidor, que puede ser una consulta de una página web o un servicio, al cual el servidor inmediatamente da una respuesta (Open Source, 2020).

Entre el web server más populares por su facilidad de uso y por ser *software* libre podemos mencionar principalmente a Apache 2 http Server y Nginx, cuyos logos se puede apreciar en la figura 31.

Figura 31 Nginx y Apache, 2 de los web servers de open source más populares.

Tomado de (Open Source, 2020).

Para las pruebas de funcionamiento en sistemas con arquitectura ARM, se utilizó el web service Apache, ya que puede integrar directamente en un solo contenedor *hypertext preprocessor (PHP),* el cual puede ser cargado desde su imagen oficial del *Docker Hub*, para luego configurarlo y ponerlo a prueba en cualquier navegado como se muestra en la figura 32.

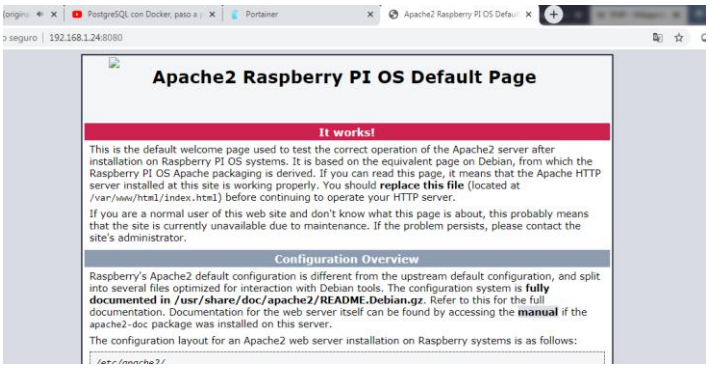

Figura 32 Muestra de la instalación del contenedor Apache.

1.4.3.7. Gestores de Bases de Datos

Un Sistema Gestor de Base de Datos (SGBD) o *DataBase Managenent System* (DBMS) permite la gestión y administración de bases de datos, así también la elección y manejo de la estructura de para el almacenamiento y consulta de información allí almacenada (Inesem, 2020).

En modelo almacenamiento más utilizado en la actualiza es el SQL y se basa esencialmente en el establecimiento de vínculos entre los datos, y asociándola a una posible tabla aparte en cada relación creada con sus propios registros y atributos (Inesem, 2020).

Entre los principales gestores de bases de datos de *software* libre actualmente podemos nombrar a MySQL, Postgre SQL y Maria DB.

De la misma manera para la ejecución de un contenedor de bases de datos, es necesario cargar su imagen desde el docker hub, y crear el contenedor para luego proceder con la creación de las bases de datos dentro de sí, como se aprecia en la figura 33 el contenedor de Postgres versión 10.13.

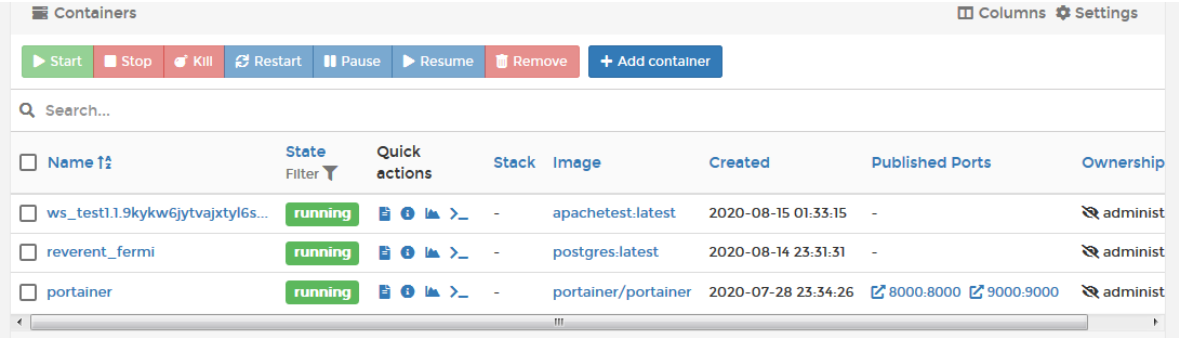

Figura 33 Visualización de los contenedores del web server y Postgres en Portainer.

1.5. Sistemas convergentes e hiperconvergentes

En la actualidad la mayoría de los sistemas en centro de datos o en las organizaciones son de tipo convergente, y las empresas tratan de captar clientes que

deseen migrar a hiperconvergencia a continuación revisaremos conceptos básicos de ambos.

1.5.1. Convergencia

La infraestructura convergente es uno de los segmentos de más rápido crecimiento del sector informático. Los "paquetes" de infraestructura integrada reducen los retos que presenta la implementación, disminuyen las inversiones asignadas a la adquisición de equipos, facilitan la capacidad de ampliación y reducen la complejidad de la gestión. La *International Dockworkers Council* (IDC*)* prevé que las inversiones globales en sistemas convergentes se incrementarán de 2.000 millones de dólares en 2012 a 17 800 millones en 2016, una tasa de crecimiento compuesta anual del 55 %. Estas soluciones suelen diseñarse con módulos individuales, pero de fácil integración, para servidores, almacenamiento y redes, así como con servicios de gestión integrada (Dell, 2014).

Su configuración se puede realizar en varios ambientes físicos y formatos, incluidos los chasis tradicionales para centros de datos y los blades independientes. Esta configuración estrictamente alineada ha sido diseñada para facilitar la ampliación de la infraestructura física y servir de base para los centros de datos híbridos al combinar arquitecturas físicas, virtualizadas y basadas en cloud (Dell, 2014).

1.5.2. Hiperconvergencia

La infraestructura hiperconvergente (HCI) permite acoplar los recursos informáticos, el almacenamiento y la red en un solo sistema. Esta solución simplificada utiliza *software* y servidores x86 para sustituir el *hardware* costoso y diseñado con fines específicos, como se muestra en la figura 34, abarcando todo los elemento en una misma infraestructura.. Gracias a la infraestructura hiperconvergente, es posible reducir la complejidad del centro de datos y aumentar su escalabilidad (VM Ware, 2020).

La implementación de una arquitectura tradicional de tres niveles es muy costosa, su funcionamiento es complejo y su escalabilidad, difícil. La infraestructura de TI podría no responder a los requisitos de las aplicaciones. Adoptar la HCI ayuda a mantener el control y los costes, sin renunciar a la seguridad (VM Ware, 2020).

Figura 34 Componentes de la infraestructura hiperconvergente.

Tomado de (VM Ware, 2020).

1.6. Dispositivos de almacenamiento

Los dispositivos de almacenamiento se refieren al espacio o lugar físico donde se almacena la información, ya que a pesar de que la tendencia sea que todo se ejecute en cloud, esta información y los dispositivos deben estar albergados en algún lugar físico. A continuación, se revisará los conceptos y tipos de dispositivos y los sistemas que se puede crear con ellos.

1.6.1. Tipos de dispositivos de Almacenamiento

Los dispositivos de almacenamiento pueden ser, discos duros o *hard disk drive* (HDD) y unidades de estados solido o *solid state drive* (SSD).

1.6.1.1. Dispositivo HDD

El HDD es un dispositivo magnético de tipo no volátil en el cual se almacena toda información de un computador o una red; que puede ser datos o programas

informáticos. Este compuesto por lo general de varios discos o platos que son controlados por dos cabezales de lectura/grabación, uno para cada lado como se muestra en la figura 35 (Cisco Storage, 2011).

La capacidad de almacenamiento de datos es medida desde *megabytes* (MB), *gigabytes* (GB) y en la actualidad hasta el orden de los *terabytes* (TB). Desde su inicio los HDD se instalan internamente en los computadores con los puertos *integrated drive electronics* (IDE) y *serial advanced technology attachment* (SATA), aunque en la actualidad se puede tener un HDD externo para respaldar información importante mediante conector *universal serial bus* (USB) (Cisco Storage, 2011).

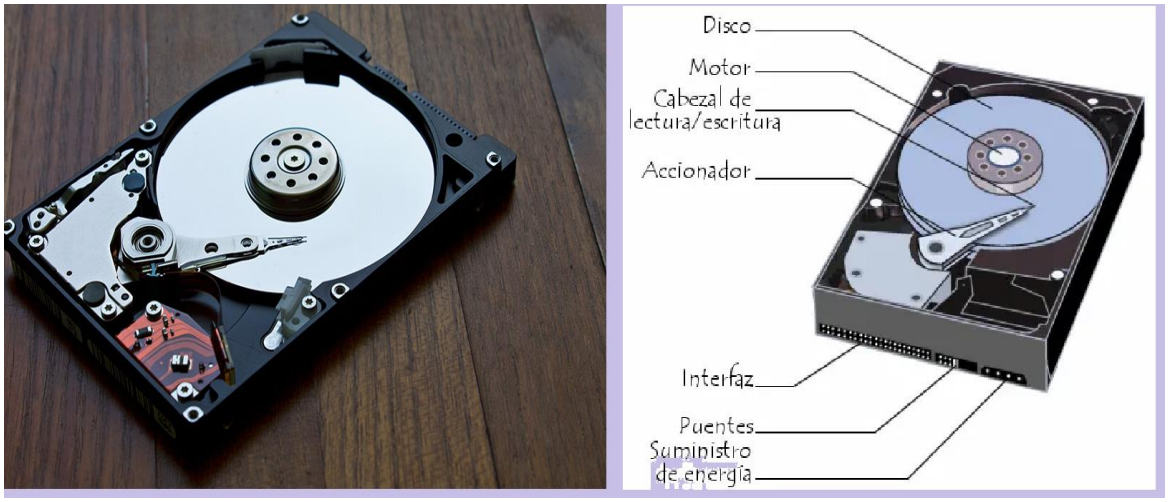

Figura 35 Estructura interna de un Disco Duro y sus componentes.

Tomado de (Cisco Storage, 2011).

1.6.1.2. Dispositivo SSD

La SSD es un dispositivo de almacenamiento no volátil, basado chips de memoria flash como se puede visualizar en la figura 36, que a diferencia de los HDD no contienen partes móviles ni mecánicas, por lo cual la escritura y lectura es mucho más rápida, así mismo los hace mucho más resistentes a golpes o caídas (Cisco Storage, 2011).

Los SSD utilizan el principio de las memorias EEPROM para su funcionamiento, pero incorporando la función de lectura y escritura en múltiples posiciones al mismo tiempo, mejorando totalmente el tiempo de respuesta o consulta de información por lo que son mucho más rápidas que cualquier HDD (Cisco Storage, 2011).

Figura 36 Unidades de estado sólido y sus partes.

Tomado de (Cisco Storage, 2011).

1.6.2. Sistemas de almacenamiento

Con los HDD o los SDD se puede crear sistemas de arreglos de discos en un *redundant array of independiente disks* (RAID) y sistemas de almacenamiento compartido en red, *network attached storage* (NAS).

1.6.2.1. RAID’s

Un RAID es un grupo de dispositivos físicos de almacenamiento independientes, que ofrece una mejora en el rendimiento con el aumento de la cantidad de unidades, para lo cual puede utilizarse dispositivos HDD o SSD, e inclusive tener un RAID de con ambos tipos de dispositivos. (Dell Technologies, 2020).

Según las necesidades se puede crear varios tipos de RAID, sin embargo, los más utilizados son:

RAID 0: Utiliza bandas de disco para proporcionar datos alta utilidad, principalmente para cantidades grandes de información, en un entorno que no requiere redundancia de datos. En la figura 37 se visualiza el modelo del RAID 0 Requiere mínimo 2 dispositivos de almacenamiento que pueden ser HDD o SSD (Dell Technologies, 2020).

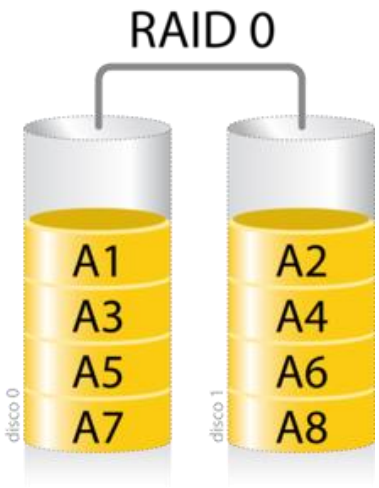

Figura 37 Modelo RAID 0.

Tomado de (Dell Technologies, 2020).

RAID 1: Es un espejo ya que se realiza replicación de los datos escritos en un disco físico hacia otro disco físico, como se aprecia en la figura 38. Este modelo espejo se puede utilizar en ambientes de alto rendimiento que requieren redundancia de datos. Requiere al menos 2 dispositivos de almacenamiento para implementarlo, el arreglo puede ser de HDD o SDD (Dell Technologies, 2020).

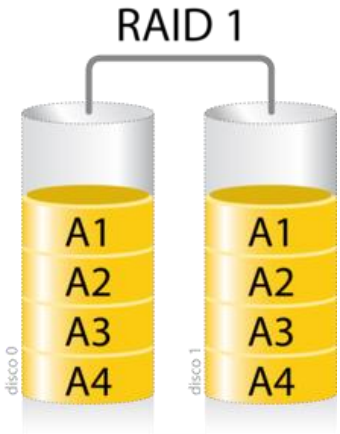

Figura 38 Modelo de RAID 1.

Tomado de (Dell Technologies, 2020).

RAID 5: Utiliza bandas de disco y datos de paridad en todos los discos físicos para proveer alto rendimiento y redundancia de datos como se muestra en la figura 39, recomendable para para ambientes pequeños y aleatorios. Requiere mínimo 3 disco para su implementación (Dell Technologies, 2020).

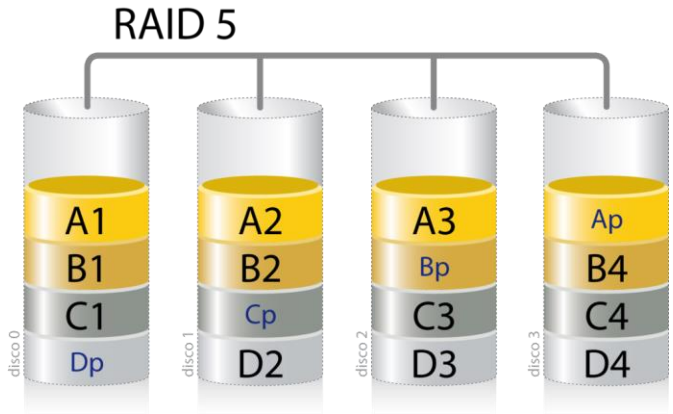

Figura 39 Modelo de RAID 5.

Tomado de (Dell Technologies, 2020).

RAID 6: Es una extensión de RAID 5 pero con un bloque paridad adicional. RAID 6 usa bandas a nivel de bloque, con dos bloques de paridad distribuidos por todos los discos miembro para proporcionar protección contra errores de discos dobles y

errores cuando un único disco está sincronizando (Dell Technologies, 2020). En la figura 40 podemos apreciar el modelo del RAID 6.

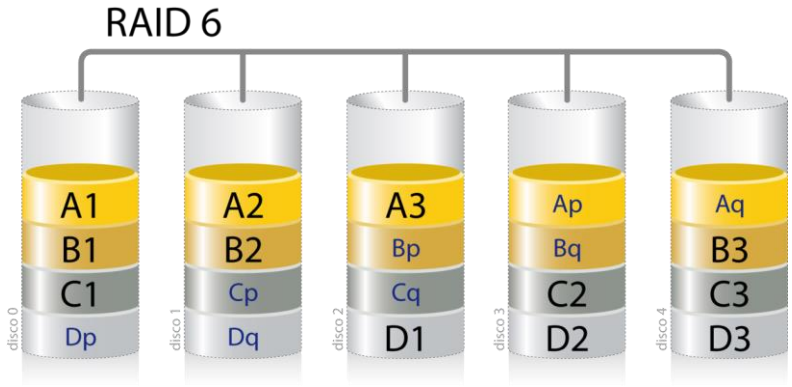

Figura 40 Modelo RAID 6.

Tomado de (Dell Technologies, 2020).

1.6.2.2. NAS

El almacenamiento en red y uno de los más populares en la actualidad es mediante los sistemas NAS, se trata de un dispositivo de almacenamiento en red de alto rendimiento que satisface las necesidades de las pequeñas empresas (Cisco Storage, 2011).

Un NAS tiene algunos componentes como son las unidades de almacenamiento que pueden ser tipo HDD o SDD como se muestra en la figura 41, el procesador integrado y el controlador de red. Estos elementos influyen significativamente en la rapidez con la que el sistema transfiere los datos, hacia y desde la red (Cisco Storage, 2011).

Figura 41 Dispositivos NAS con Unidades de HDD y SSD.

Tomado de (Cisco Storage, 2011).

2. CAPÍTULO II. ANALISIS TECNOLÓGICO DE HIPERCONVERGENCIA

En el mercado se encuentra variedad de ofertas para el servicio de hiperconvergencia, sin embargo, al contratar esos servicios el costo puede ser elevado ya que la información debe almacenarse en un nodo final que puede ser un servidor de un centro de datos. Aparte de los costos no se tiene el 100% del controla de la información y puede verse comprometida en cualquier momento.

En este capítulo se analizarán las tecnologías y costos de los elementos del prototipo, así como el de otras soluciones en el mercado para luego realizar una comparativa con ventajas y desventajas de cada una de ellas.

2.1. Análisis de costo inicial del prototipo

El prototipo cuenta con varios elementos con los cuales forman los componentes básicos de un centro de datos, en la tabla 6 se detalla cala elemento y su costo inicial:

Tabla 6 *Detalle de elementos y costo.*

Componente	Elemento	Cantidad	Total	Observaciones
Computo	Raspberry PI 4	4	$ 272,00	Procesamiento de datos, formaran el hardware del cluster
	Fuentes de poder	4	$ 60,00	Alimentación para cada Raspberry PI 4
Almacenamiento	HDD 2TB o SSD 50 GB	1	$ 81,76	Sera la unidad de almacenamiento de red o NAS
	Micro SD 16GB+	4	$ 20,00	Para albergamiento del Sistema operativo de cada Raspberry PI4
Red	Switch	1	$ 15,00	Ayudará a la expansión de la red
	Patch Cord Cat 5e	6	$ 6,00	Conexión física de las Raspberry PI en la red
	Router	1	$ 20,00	Permite el enrutamiento de paquetes entre elemento de la red LAN y hacia afuera de la red
Energía	Cortapicos	1	$ 5,00	Ayuda a crear expansión energética y regular que la energía no tenga sobrevoltajes
	Regulador Voltaje	1	$ 20,00	Equipos opcionales ya que permite que la energía que ingresa al prototipo no tenga sobrevoltajes.
		Total	$ 499,76	

El costo total del prototipo es de $499.79, el tiempo de vida útil estimado de cada Raspberry PI4 y sus componentes es de 3 años por lo que el costo anual de una implementación así tendría un valor de $166,58**,** por año. Adicional a este valor es necesario considerar el costo de la implementación o mano de obra.

2.2. Análisis de consumo energético

Para realizar el cálculo de consumo energético del *cluster* de hiperconvergencia se tomó como referencia las especificaciones dadas por el fabricante, el cual especifica

que el consumo máximo seria de 15,3 W y se aplicó la formula #W x 24 horas de operación y por 30 días al mes como se muestra en la ecuación 1 (CentroSur, 2020), tomada como referencia de la empresa eléctrica CentroSur de la ciudad de Cuenca:

$$15{,}3Wx\frac{24h}{1d}x30\ d = \frac{11.016\ Wh}{1000Wh/1kWh} = 11{,}016\ kWh$$

Ecuación 1.

El consumo mensual de cada Raspberry es de 11,016 kWh, por tanto, el total del prototipo es de 44,064 kWh al mes ya que son 4 Raspberry que componen el prototipo. Finalmente, al año se tendrá un consumo de 528,768 kWh, y para calcular el valor anual por consumo de energía considerando que en Ecuador el costo por kWh es de 0,08, el valor final lo adquirimos mediante la siguiente ecuación.

$$\frac{528{,}768kWh}{1\ año}\ x\ \frac{\$0{,}08}{1\ kWh} = \$42{,}30\ /\ año$$

Ecuación 2.

Un valor que se puede agregar como costo adicional del prototipo es de $42,30 por consumo de energía, mismo que comparado con el consumo de otros dispositivos representa un costo muy bajo.

2.3. Análisis de soluciones de hiperconvergencia

En el mercado actual se brindan soluciones para empresas que ofrecen los servicios de hiperconvergencia a un costo que por lo general desborda el presupuesto de las microempresas y por lo tanto es muy difícil acceder a estos servicios, a continuación, en la tabla 7 se hace un análisis breve dentro del cual se extrae un análisis costo beneficio de cada implementación con características similares:

Tabla 7 Ejemplo de Servicios de Hiperconvergencia.

Plataforma	Características	RAM	Almacenamiento	Costo Mensual	Anual
Amazon *Web service*s	Google High CPU 2 vCPU n1-custom-2-4096	4GB	80 GB SSD	$ 70,05	$ 840,60
Google Cloud	AWS 2 vCPU Cálculo optimizado c5.large	4GB	80 GB SSD	$ 63,38	$ 760,56
Linode	Linode 2 vCPU Dedicado	4 GB	80 GB SSD	$ 30,00	$ 360,00
Cisco UCS 6200	UCS-FI-6248UP	n/e	-	$ 33464,99	$ 11154,99

2.4. Ventajas y desventajas.

En la tabla 8 se detalla las ventajas y desventajas de cada una de las opciones anteriormente expuestas, al final se realizará un análisis de cada una de las opciones detalladas:

Tabla 8 Ventajas y desventajas de cada implementación.

Plataforma	Ventajas	Desventajas
Hiperconvergencia Raspberry PI 4	-Bajo Costo de implementación y mantenimiento, menor a los $ 200 por año. -No requiere mantenimiento permanente -Diseño modular, cada nodo es independiente y tiene toda la información replicada. -Bajo consumo energético anual, <$50 al año por cada prototipo - Control total de la información - No requiere licenciamiento ya que se	-Requiere personal especializado para el levantamiento de los dominios y páginas.. - Para redundancia total se requiere el levantamiento de otro prototipo similar y eso implica un mayor consumo energético. -Requiere una fuente alterna de energía, como puede ser un UPS para tener redundancia total, esto se traduce en un costo adicional.

	utiliza *software* libre para su implementación. -Lo elementos son pequeños por tanto no requiere mucho espacio para la implementación. -Se puede agregar o eliminar nodos inclusive en caliente, sin detener las operaciones.	
Plataformas pagadas (Amazon, Google, Linode)	- Solo se paga la tarifa y el proveedor se encarga de los mantenimientos -Panel de configuraciones fácil de utilizar. -Redundancia total de la información que se encuentra disponible 24/7 - La asignación de recursos es escalable y por tanto permite agregar información	-Alto Costo anual por encima de los 800 solo el espacio y los recursos. -Requiere la adquisición de licencias y compra de dominios esto implica costos adicionales. -La información puede estar comprometida al no tener el control total de ella.
Centro de datos propio	- Todos los servicios en un solo equipo -Todas las configuraciones pueden guardarse en una tarjeta y simplemente levantarlas - Diseño escalable ya que se puede agregar o eliminar elementos. - Desde la interfaz se levanta fácilmente máquinas virtuales y se puede guardar la configuración de los servidores. -Así mismo la asignación de recursos a cada máquina virtual es bastante sencilla y dinámica ya que si se requiere más recursos se puede agregar en cualquier momento.	-Requiere personal especializado para el levantamiento de los dominios y páginas. -Alto costo de implementación un UCS puede costar alrededor de los 20000 a 40000 (Itprice, 2020). -Ocupa gran espacio físico por lo que es necesario levantar racks y por tanto eso requiere otros gastos. - Requiere el pago de licencias para el uso de *software*. -Si es UCS sufre daño mientras se repara puede quedar mucho tiempo fuera de servicio. -Para redundancia se requiere la adquisición de otro equipo UCS esto implica un doble en lo económico como en el consumo energético.

La implementación de la propuesta del prototipo de hiperconvergencia tiene ventajas que en general va por lo económico, y con integridad y seguridad de saber que la información se encuentra bajo el control de la misma organización, a diferencia de las plataformas de cloud que, si bien ofrecen implementaciones fáciles, el costo anual es mucho más elevado y la integridad de la información podría también verse comprometida. Implementar un centro de datos propio será la mejor opción por el control de la información sin embargo es la más costosa por la adquisición de los equipos.

3. CAPÍTULO III. DISEÑO DEL PROTOTIPO DE HIPERCONVERGENCIA

En el diseño del prototipo se determinará las especificaciones técnicas de *hardware* y *software* para posteriormente implementar la solución de hiperconvergencia, que está compuesto de dos capas: capa física donde se tienen todos los dispositivos físicos interconectados entre sí y la capa lógica que es *software* que se utiliza para programar la funcionalidad de cada dispositivo físico.

Para entender el prototipo se presenta un resumen en la figura 42 donde se puede apreciar que cuenta con 2 capas principales que son la física o *hardware* y la capa lógica donde se encuentra el *software*.

En la parte inferior de *hardware* se encuentran las Raspberry PI 4, que son el componente de procesamiento, además están el *router*, *switch* y cableado estructurado que conforman el componente de red y finalmente el HDD que conforma el componente de almacenamiento.

En la parte superior se encuentra alojado el *software* el cual el cual es la parte lógica y programable, la cual en su mayor parte se encuentra controlada por *Docker* misma que se encarga del funcionamiento del *cluster* mediante su función *swarm*. Además, permite el control de los contenedores que permitirán la ejecución de los programas y servicios.

Los programas que se puede ejecutar en el *cluster* está limitado únicamente por la compatibilidad con el *software Docker*, para ello cada servicio y programa debe tener una imagen oficial en la página del *Docker* Hub, la cual tiene una librería completa de *software* de tipo servidores web, bases de datos, herramientas para programación, etc.

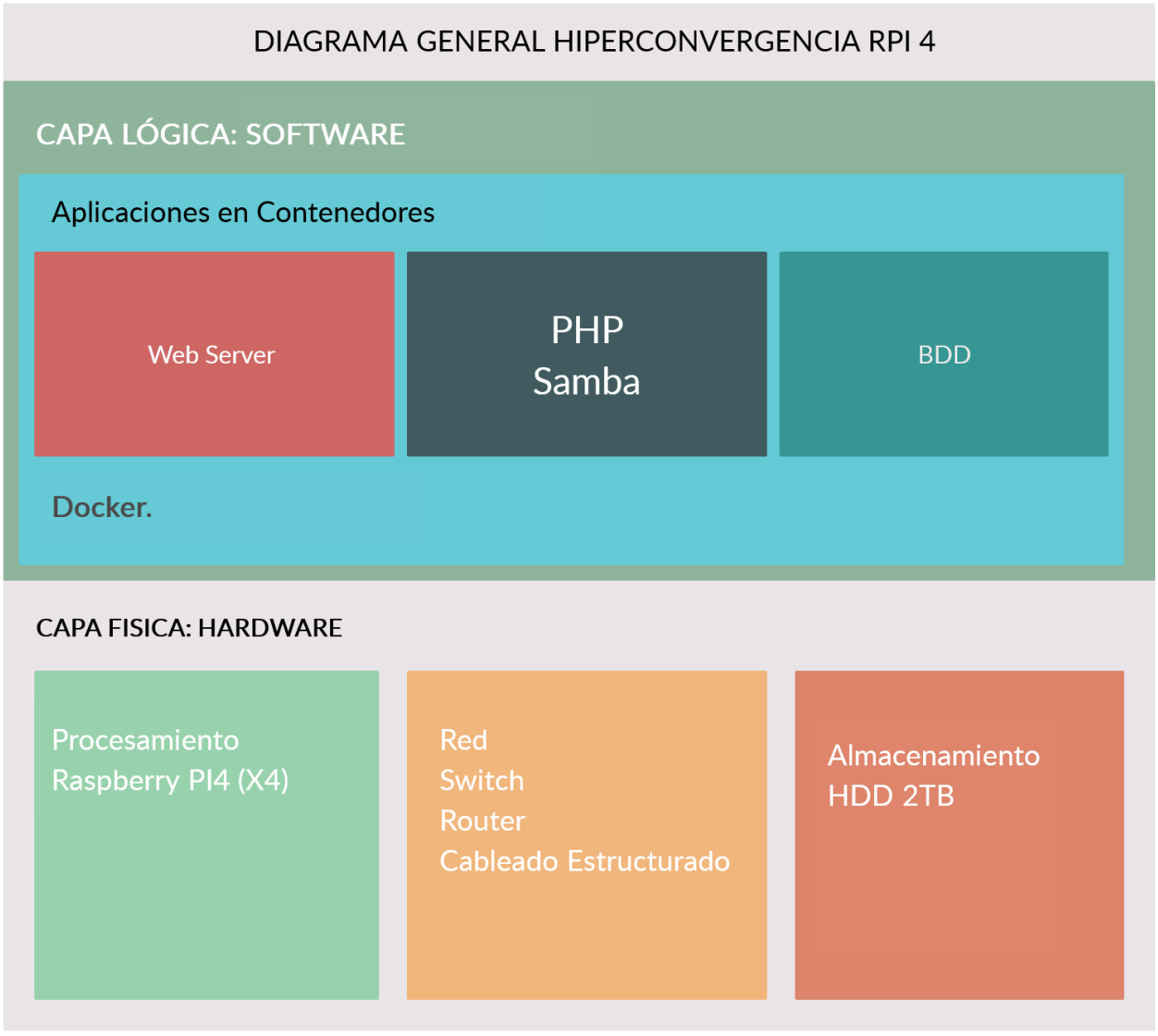

Figura 42 Esquema general de *cluster* Hiperconvergencia RPI4.

3.1. Especificaciones técnicas del prototipo

En la tabla 9 se detallan las especificaciones técnicas del *cluster* una vez que se encuentre trabajando en conjunto mediante la unión lógica del *software Docker*.

Tabla 9 Especificaciones del *cluster* Raspberry.

CPU	RAM	Almacenamiento
Broadcom, 1,5 GHz quadCore Total 16 Núcleos	4GB (1GB por RPI4)	HDD 2TB O SSD 500GB

Cada Raspberry PI4 Tiene un procesador Broadcom de 4 núcleos y al unificarlos en el *cluster* tenemos un total de 16 núcleos para el procesamiento, esta unidad se puede visualizar en el Portainer que administrador grafico del *cluster*.

La memoria RAM de cada RPI4 es de 1GB, al combinarlas tenemos un total de 4 GB totalmente ampliable debido a que se puede adquirir más placas de mayor capacidad de RAM y unirlas al *cluster*.

El almacenamiento para el Cloud, se lo puede manejar en disco duro externo de 2 TB o un Disco de estado Solido de 500GB cuyo costo es de lo $81, en cada caso.

3.2. Diseño de la capa *Hardware*

El *hardware* es la parte física del prototipo el cual esta interconectado mediante cables UTP de red RJ45, hacia un switch y posteriormente para salir a internet a un router, básicamente se compone de 3 elementos que se describen a continuación.

3.2.1. Procesamiento o Computing

Como se muestra en la figura 43 la parte de red tiene 2 tipos de uniones, física que se realiza a través de los cables ethernet y un switch y la unión lógica que lo realiza el aplicativo docker *swarm* mediante el uso de las direcciones IP asignadas a cada tarjeta de red.

Dentro del *Docker swarm* cada elemento tiene su dirección IP y su rol predefinido como se detalla en la tabla 10:

Tabla 10 Detalle de elementos del *cluster* y su rol.

Hostname	IP	Rol
pimaster	192.168.1.24	*Manager* principal
pimaster2	192.168.1.28	*Manager* secundario (Standbye) para redundancia
*worker*1	192.168.1.25	Esclavo 1
*worker*2	192.168.1.26	Esclavo 2

En la figura 43 se muestra el diseño de conexión de la parte de computing o procesamiento

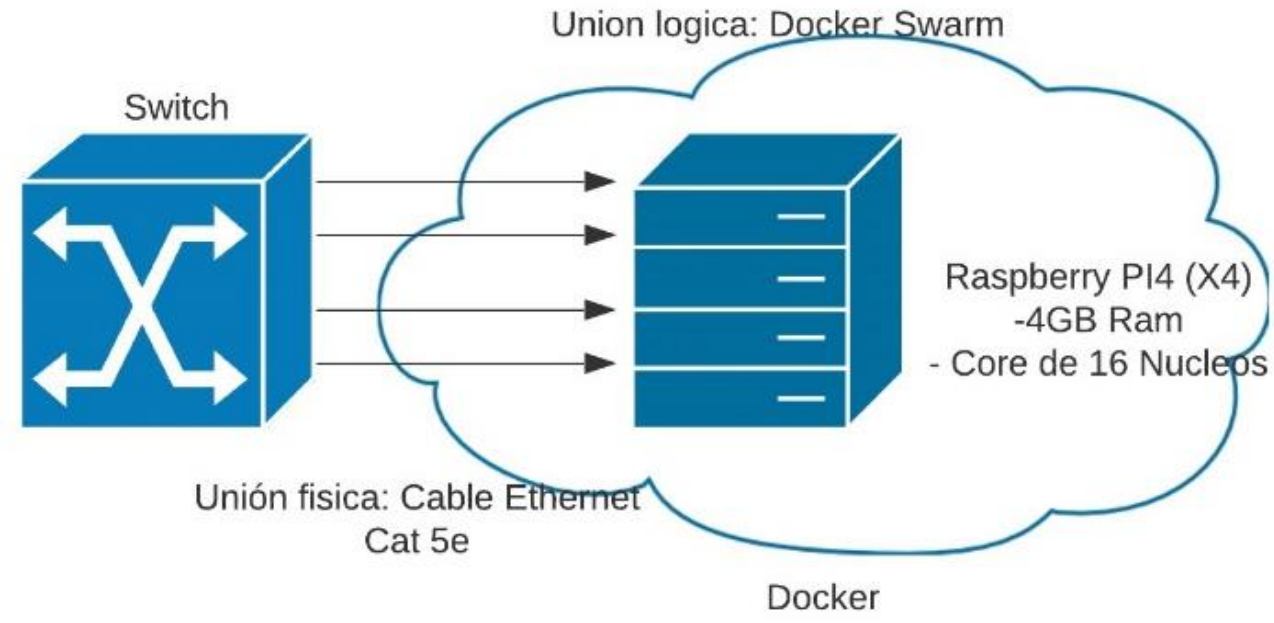

Figura 43 Diseño del computing.

3.2.2. Red

El componente de red como se muestra en la figura 38, realiza la conexión física del *cluster* entre cada una de las placas y además mediante el uso del switch y de cables ethernet y el switch permite al *cluster* salir hacia la internet.

Cada una de las placas cuenta sus direcciones IP correctamente definidas según la tabla 10.

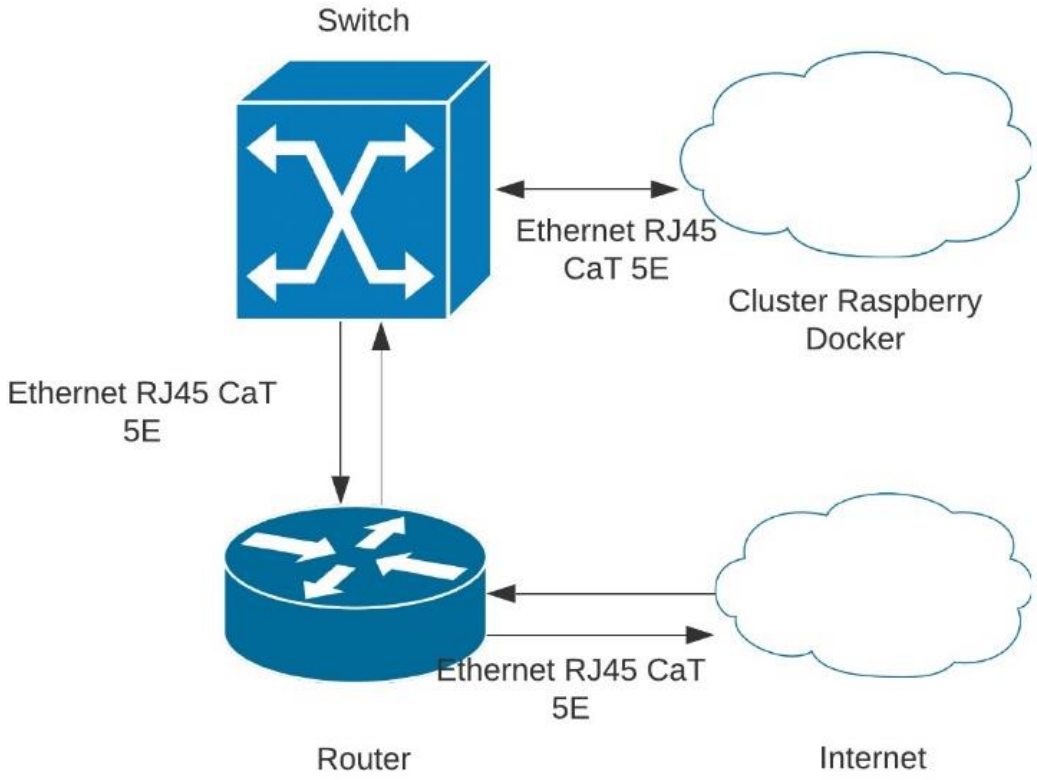

Figura 44 Diseño de la red.

3.2.3. Almacenamiento

El almacenamiento es la parte que se encargara de contener la carga pesada de información y para que funcione como un sistema hiperconvergente es necesario la creación de RAID interconectado a cada uno de los componentes del clúster, por tanto, se creará únicamente un diseño de su funcionamiento con el *software* Open Media Vault.

Para demostrar el funcionamiento básico del sistema de almacenamiento se usará un disco duro externo de 2TB conectado mediante USB 3.0 a la Raspberry PI del nodo *manager* y el diseño de su funcionalidad NAS será creada mediante el *software* Samba.

3.2.3.1. RAID con *software* Open Media Vault

Con la ayuda del *open media vault* se puede crear RAID de discos duros para mejorar el rendimiento y con ello también tener redundancia y respaldo de la información guardada en los discos. En la figura 45 tenemos el diseño de un RAID 5 que puede ser creado con 3 discos, conectados a las Raspberry por medio de un adaptador GPIO-SATA, y su administración se lo realizará con el *software Open Media Vault* (OVM) .

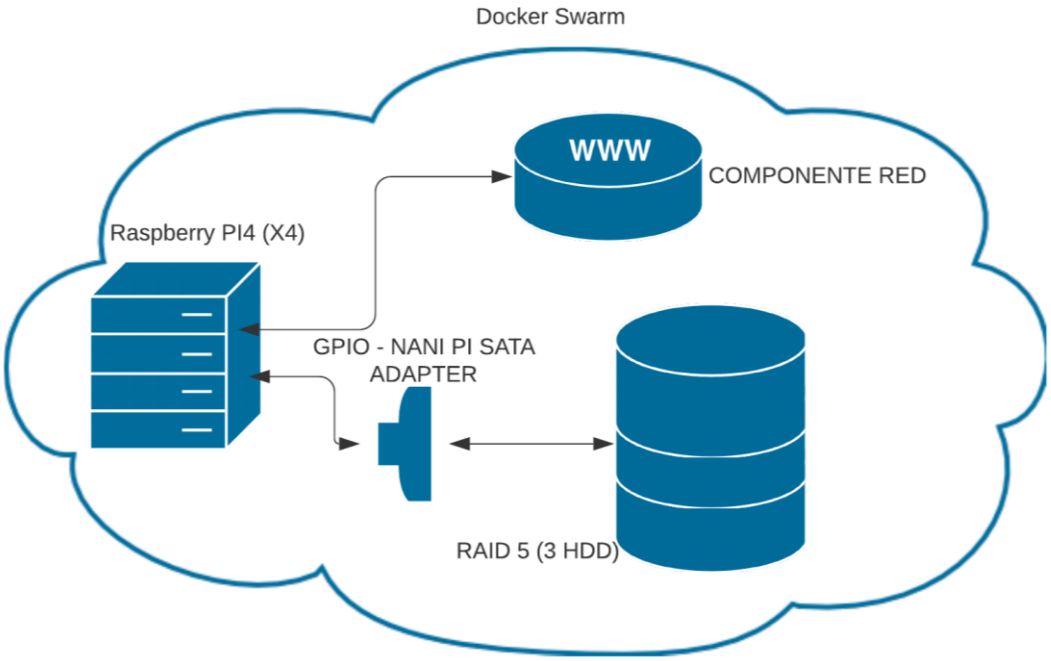

Figura 45 Diseño de conexión RAID.

La conexión de los discos para su funcionamiento se lo puede realizar mediante un adaptador GPIO-Nano Pi Sata, el cual se conecta a los pines del GPIO y permite la conexión de hasta 4 disco SATA al *cluster* y el control del RAID se lo realiza por medio del OMV.

3.2.3.2. NAS con *software* samba

El componente de almacenamiento mostrado en la figura 46, es un disco duro externo que se conecta al *cluster* mediante el puerto USB 3.0 y utilizando por medio de Samba se lo convierte en un elemento de red que permite tanto la lectura y escritura en el disco a través de la red**.**

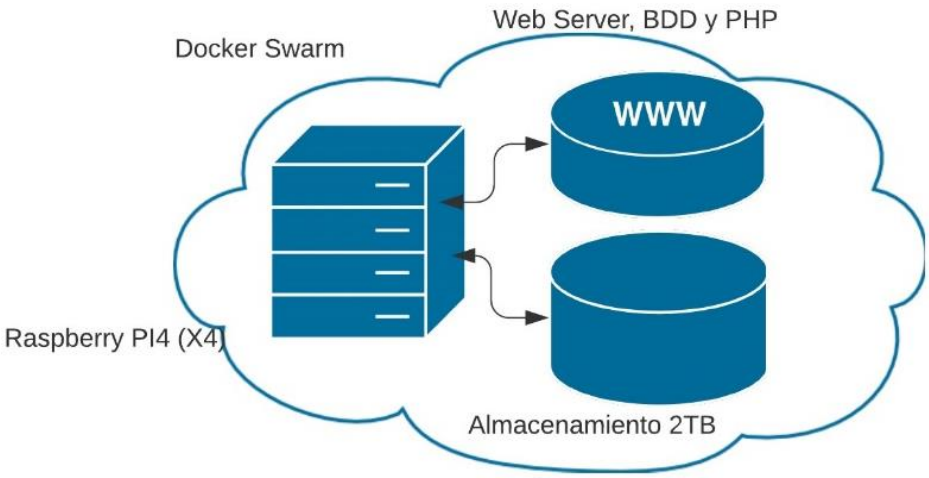

Figura 46 Diseño del componente de almacenamiento.

3.3. Diseño de la capa *software*

Todos los componentes físicos para poder trabajar de manera conjunta requieren de una programación mediante aplicativos o sistemas, en este caso para el trabajo conjunto de las placas Raspberry se utiliza como sistema operativo base el Raspberry PI OS y el *Docker*. El *Docker* se encarga de asignar las tareas y realizar balanceo de carga para que las 4 placas distribuyan su trabajo de manera equitativa.

La administración del almacenamiento se lo efectúa mediante Samba, misma que es un gestor de archivos en red que utiliza el protocolo ftp, para permitir la exploración del contenido de los discos.

3.3.1. *Docker* (Contenedor)

El *Docker* es el *software* que se encarga de realizar la comunicación entre las Raspberry PI4 mediante su función *swarm*, misma que crea un sistema unificado de recursos de *hardware* para trabajar en conjunto como un solo dispositivo. En la figura 47 podemos apreciar el modelo de trabajo del *Docker swarm*, manteniendo un nodo *manager* que es quien asigna las tareas a los demás nodos que estén disponibles, además el *Docker* se encarga de administrar y ejecutar las aplicaciones como contenedores que son los que prestarán finalmente los servicios a los usuarios.

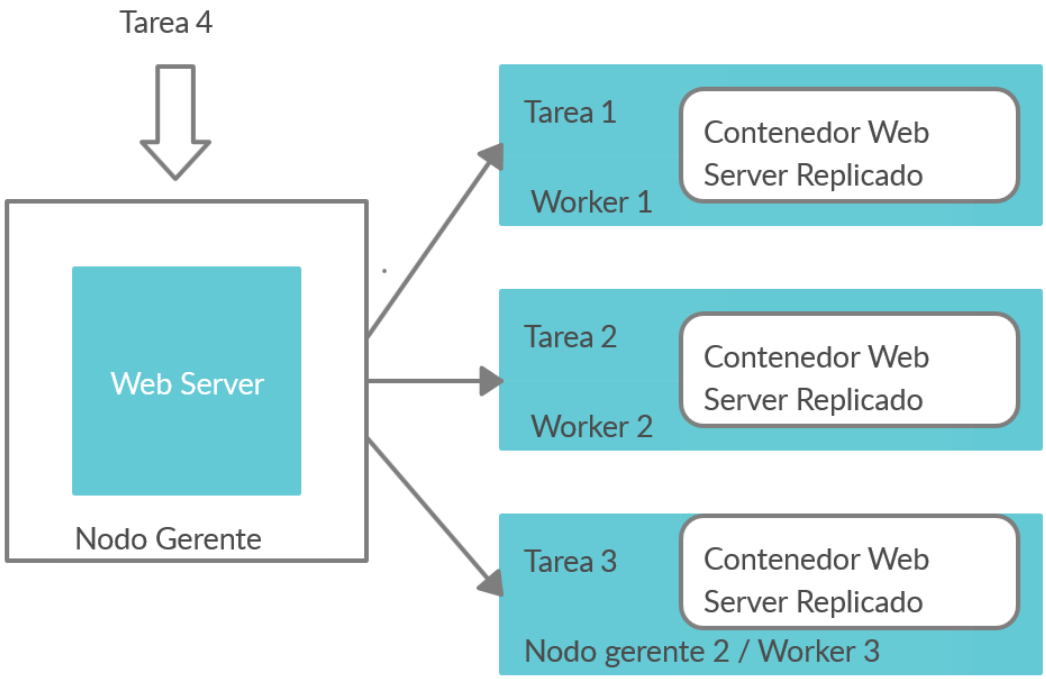

Figura 47 Diseño de balanceo de carga y redundancia en docker *swarm.*

Docker proporciona el servicio de clusterización, dentro del cual se utiliza el modelo maestro esclavo, por tanto, tenemos 3 tipo de nodos como se puede apreciar en la figura 48:

-El nodo gerente o *manager*: Es el gerente o controlador del sistema y se encarga de asignar la carga de trabajo a cada nodo disponible, es decir que actúa como un balanceador de carga.

-El nodo esclavo o *worker*. Un nodo *manager* puede tener una cantidad indeterminad de esclavo o *workers*, ya que son estos quienes se encargar de ejecutar las tareas y consultas solicitadas por los usuarios.

-El nodo gerente / esclavo: Es un nodo especial que se encuentra en *standbye*, como nodo gerente redundante que se hace cargo del *cluster* en caso de que el gerente principal falle. Mientras se encuentre en *standbye* este nodo también puede trabajar como un nodo esclavo.

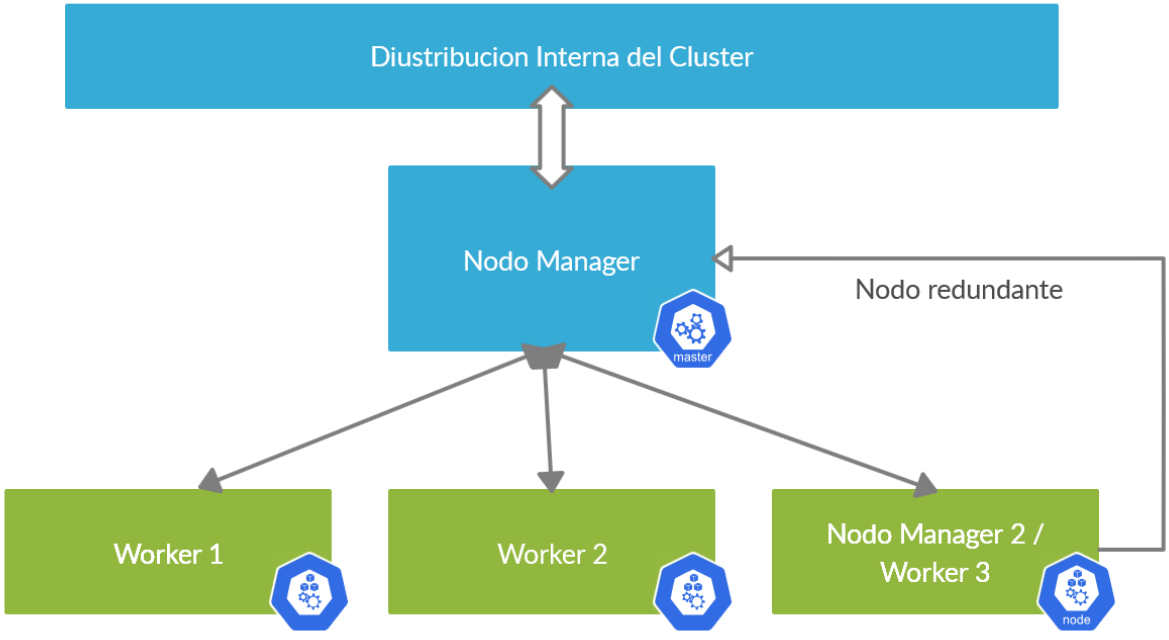

Figura 48 Tipos de nodos en *Docker swarm.*

3.3.2. Open Media Vault (OMV)

En la figura 49 se puede observar el diseño de un espacio de almacenamiento redundante que puede ser creado y administrado mediante OMV, para este diseño

se lo realiza mediante un RAID 5, para el cual es necesario la existencia de 3 dispositivos de almacenamiento de tipo SATA, con los cuales se creará el RAID a cuyo detalle de su configuración se muestra en el Anexo 6.

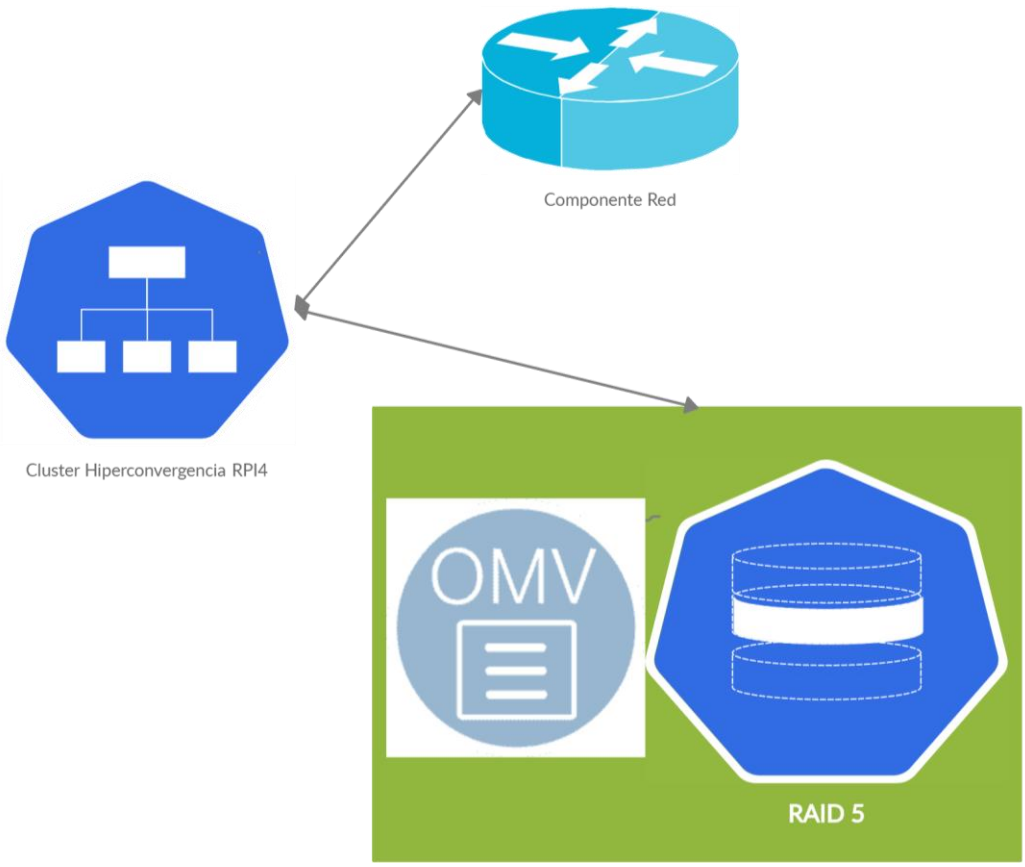

Figura 49 Diseño de RAID 5 con Open Media Vault.

3.3.3. Samba

La creación del almacenamiento se lo realiza a través de Samba que es un *software* libre que puede ser utilizado es las Raspberry. Samba debe ser instalado en cada uno de las Raspberry PI 4 y el disco deberá estar conectado a cada uno de ellos para garantizar la redundancia en cada de que alguno de los nodos falle.

Del lado del cliente se puede utilizar y configurar cualquier gestor de archivos compatible con el servicio de samba para poder visualizar el contenido del disco

duro. En la figura 50 la unidad de almacenamiento está controlada por el *software* Samba instalado en la Raspberry PI por medio de un puerto USB 3.0, y con el *software* puede prestar el servicio de almacenamiento en red.

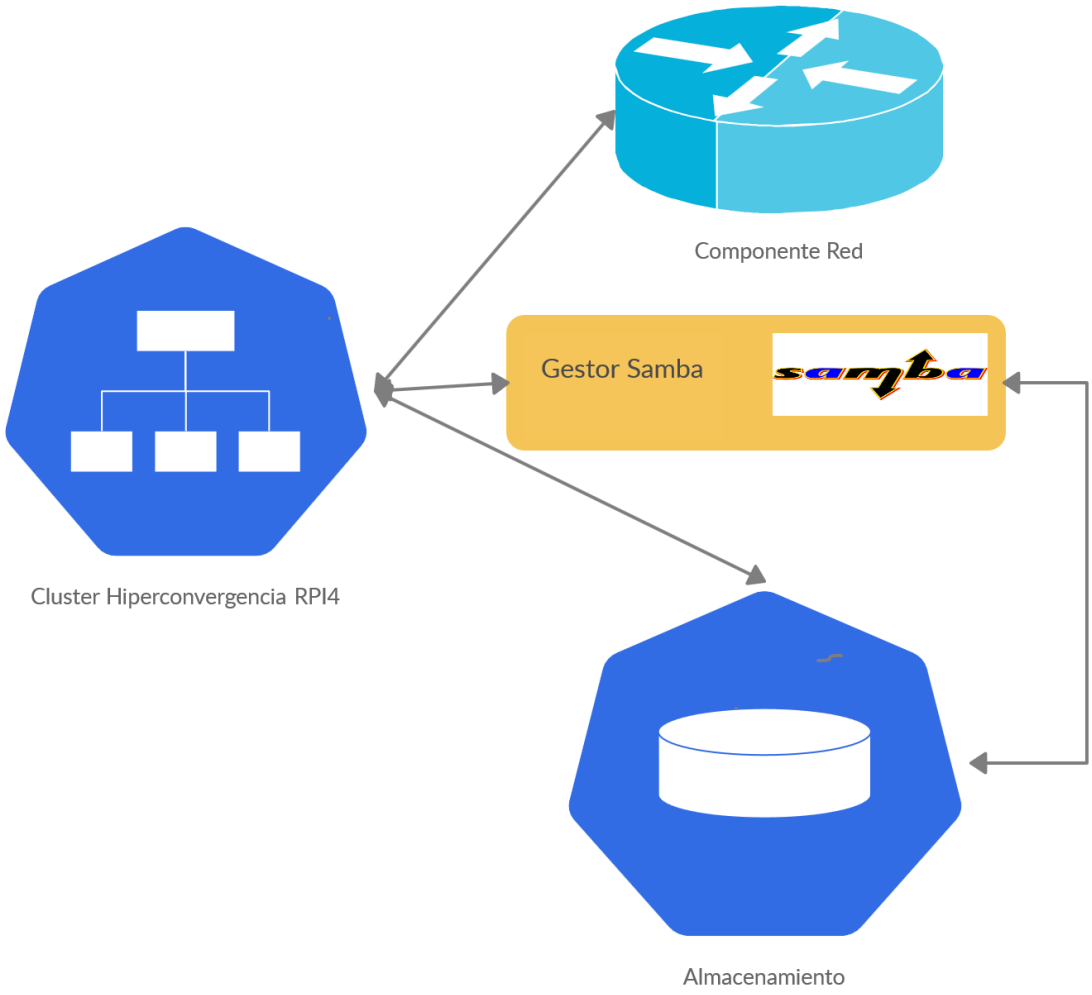

Figura 50 Almacenamiento en red controlado mediante Samba.

4. CAPÍTULO IV. IMPLEMENTACION DEL PROTIPO

Para lograr que todos los elementos del prototipo puedan trabajar en conjunto se requiere de configuraciones especificas desde la instalación del Sistema Operativo Raspberry PI OS, hasta la instalación y configuración de los servicios.

4.1. Implementación del *hardware*

Todos los elementos y recursos de *hardware* cumplen un rol específico para el prototipo, por lo tanto, se detallará el proceso de conexión físicas de cada uno ellos.

4.1.1. Procesamiento o computing.

La conexión entre la1s Raspberry PI y el switch se realiza mediante los puertos ethernet con cables RJ45, como se puede observar en la figura 51 para garantizar la transmisión de datos con bajas pérdidas.

Las Raspberry se encargarán de procesar los requerimientos de la red y resolverlos por tanto tiene la conexión con la red y el almacenamiento para poder dar respuesta a las solicitudes. El nodo maestro es el que se encargará de administrar el *cluster* y asignar las tareas a cada nodo.

Figura 51 Conexión física de cada Raspberry a la red.

4.1.2. Red.

El switch realiza el enlace o comunicación entre las Raspberry PI y la red, y aquí también se incluye el router que realiza el enrutamiento de los paquetes dentro y fuera de la red. La conexión se visualiza en la figura 52.

Figura 52 Conexión de los componentes de Red.

4.1.3. Almacenamiento.

La conexión del almacenamiento se lo realiza mediante un cable USB hacia la Raspberry, como se muestra en la figura 53. Para el ejemplo se conecta un disco duro de 2TB, en el puerto USB 3.0 del Raspberry PI principal y mediante los servicios de samba este disco será un dispositivo de almacenamiento en red.

Figura 53 Conexión de la unidad de almacenamiento.

4.2. Implementación del *software*

El *software* es fundamental para el funcionamiento del clúster ya que éste controla las funciones de cada elemento del *cluster* según la programación y la necesidad que tenga el usuario; las configuraciones se las realiza en 3 partes:

-Instalación y configuración del Sistema Operación Raspberry PI OS.
-instalación y configuración del *Docker*
-Instalación y configuración del gestor Samba

4.2.1. Instalación y configuración de Sistema Operativo Raspberry PI OS

El sistema Operativo que por defecto o recomendación de Raspberry se utiliza es el Raspberry PI OS el cual puede descargarse desde la página oficial de la fundación Raspberry como se muestra en la figura 54, mismo que se lo puede instalar por medio NOOBS, o descargando el archivo zip del Raspberry PI OS(*Raspberry Pi OS (32-bit) Lite*), para posteriormente cargarlo en la micro SD.

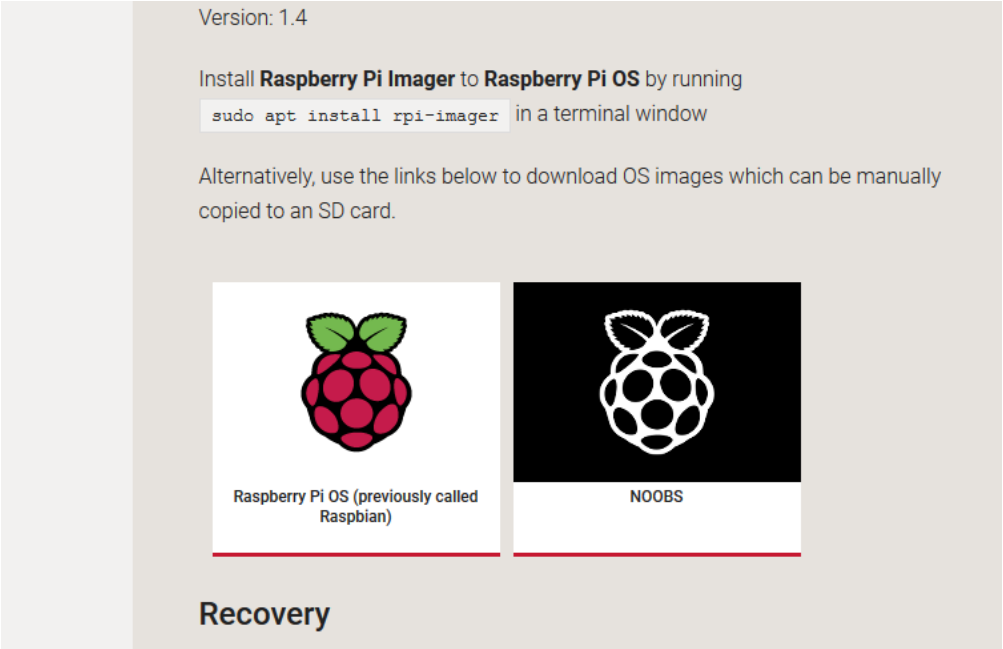

Figura 54 Opciones de instalación del SO Raspberry PI OS.

La opción por la cual se optó realizar la instalación del SO, fue descargarlo en el archivo .zip, desde la página oficial de Raspberry tal como se muestra en la figura 55:

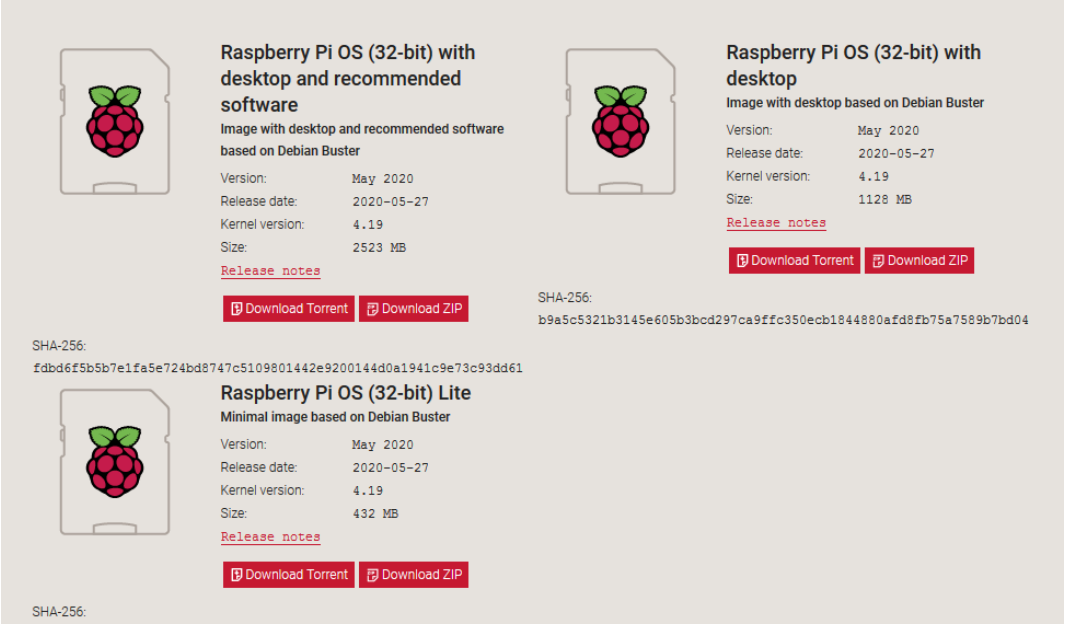

Figura 55 Selección del SO.

Debido a que las Raspberry se van a utilizar en un entorno de desarrollo o de servidores la versión a descarga fue *Raspberry PI OS Buster Lite*, misma que no tiene entorno grafico solo consola lo cual ayuda a optimar el uso de los recursos de cada placa, por supuesto cada placa se va a configurar remotamente utilizando el programa *PuTTY* y el protocolo *secure shell* (SSH).

Una vez descargado el archivo debemos formatear la micro SD para posteriormente instalar el SO en cada una de las micro SD, y finalmente se inserta la micro SD en la ranura de la Raspberry PI. El detalle del proceso y comandos se encuentra en el anexo 1.

4.2.1.1. Conexión mediante *secure Shell (*SSH)

Por seguridad el sistema operativo Raspberry PI OS por defecto tiene deshabilitado el SSH, por lo cual es necesario habilitar dicho servicio para que se ejecute desde el arranque del SO.

Existen varios métodos para habilitar este servicio, pero se utilizó las más sencillas, cuyo detalle se encuentra en el anexo 2:

-Mediante la creación de un archivo con nombre “ssh” sin extensión, en la carpeta boot la cual podemos hacerlo accediendo a la micro SD, de esta manera estamos dando la instrucción al sistema operativo que arranque el servicio ssh desde el inicio.

-Utilizando la herramienta “raspi-config”.

Con cualquiera de las dos opciones el SSH quedará habilitado para su inicio automático desde el arranque del Raspberry PI OS

Posteriormente se establece la conexión a cada Raspberry PI utilizando el PuTTY, hasta tener el resultado que son las 3 consolas habilitadas para iniciar las configuraciones, como podemos observar en la figura 56.

El usuario por defecto que viene incluido en cada Raspberry PI4, es “pi” y su contraseña es raspberry, con estas credenciales podremos ingresar a la consola de cada placa para realizar las configuraciones necesarias.

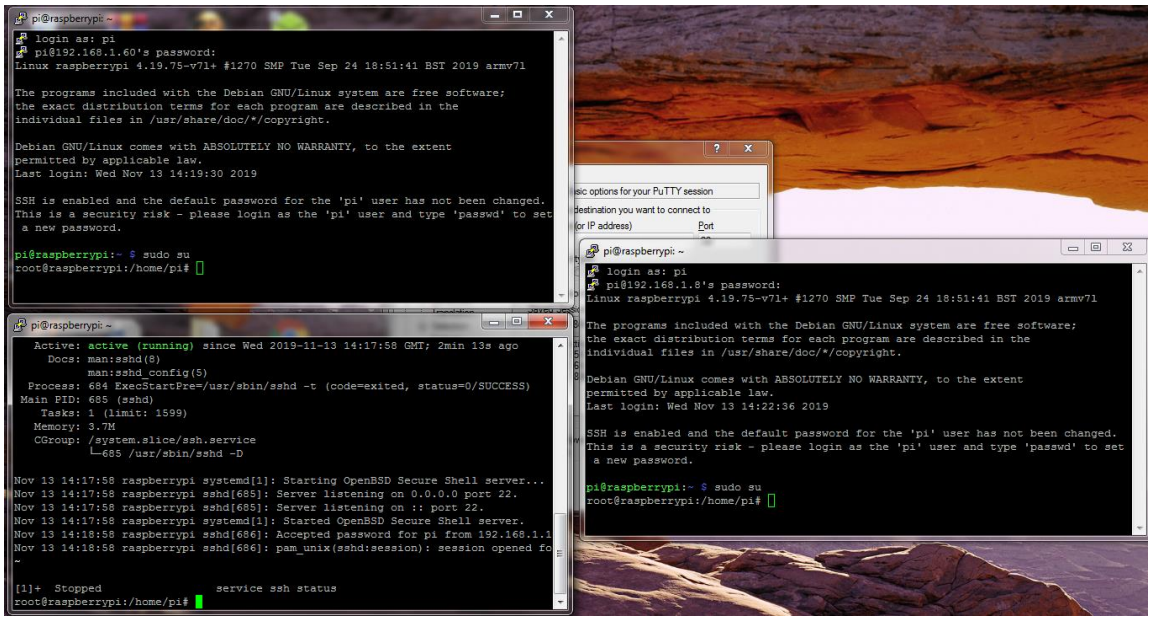

Figura 56 Control de las 3 placas por medio de SSH en la Plataforma PuTTY.

4.2.1.2. Preparación del Raspberry PI OS.

En general para poder realizar la instalación o configuración en los sistemas basados en Linux es necesario realizar la actualización de los paquetes y librerías.

Posterior a esta acción debemos realizar la identificación de cada Raspberry, ya que por defecto cada una de las Raspberry viene con un nombre estándar en cada placa, por lo cual esto puede llegar a ser confuso si se maneja gran número de placas. Todo el proceso de la preparación del Raspberry PI OS se detalla en el anexo 3.

4.2.2. Instalación y configuración de *Docker*

Una vez preparada la Raspberry se debe instalar el *software Docker* en cada uno de los nodos, para posteriormente crear el *swarm* o enjambre que controlará el *cluster* para que opere como si fuera un solo elemento, más detalle se muestra en el anexo 4.

Finalizada la configuración del *Docker swarm* podemos comprobar que los nodos este conectados y disponibles usando el comando el comando -*docker node ls,* y el resultado obtenido se muestra en la figura 57:

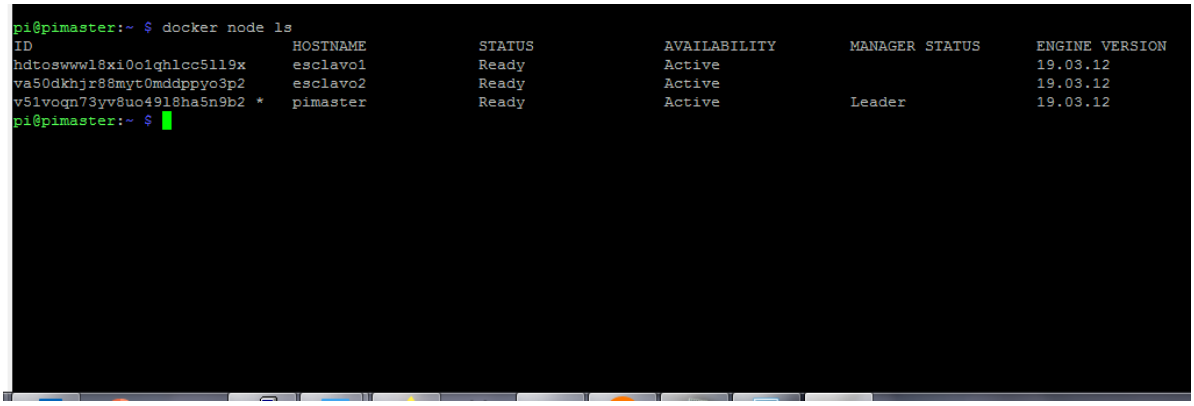

Figura 57 Docker swarm con 3 nodos.

En cada uno de ellos vemos el *hostname* y su estado que en ese momento están todos listos para recibir instrucciones por parte del *manager.*

4.2.2.1. Administrador grafico del clúster

Una herramienta útil para la administración y visualización del estado del *cluster* es Portainer misma que es de *software* libre cuya interfaz se muestra en la figura 58 y para utilizarla únicamente debemos instalar en los nodos *manager* y autorizar su acceso mediante el puerto 9000. El detalle y comandos de instalación mediante el terminal se detalle en el anexo 4.

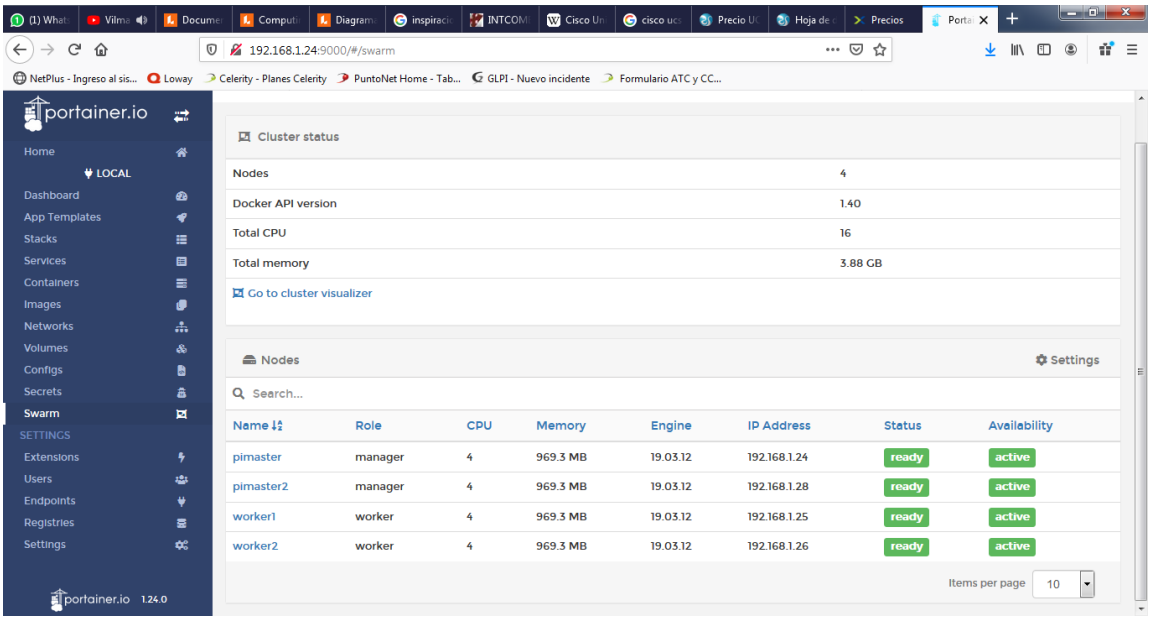

Figura 58 Interfaz gráfica Portainer.

4.2.3. Instalación y configuración de gestor Samba

Para el almacenamiento utilizamos un disco externo Toshiba con interfaz de acceso USB a fin de convertirlo en una unidad de red, para ello utilizamos varias extensiones y el servicio samba, la unidad va a ser montada en el *manager* como se detalla en él.

Para acceder a los archivos mediante la red debemos utilizar samba que permite el acceso por medio del protocolo ftp, en samba configuramos tanto el usuario y contraseña para acceder al servicio. Todo el proceso de configuración se detalla en el anexo 6

5. CAPÍTULO V. PRUEBAS Y RESULTADOS OBTENIDOS.

En este capítulo se revisará los resultados obtenidos con el diseño y las implementaciones efectuadas en los capítulos anteriores, el objetivo es demostrar que los componentes actualmente se encuentran trabajando conjuntamente como un mismo dispositivo:

5.1. Procesamiento mediante docker *swarm*

Mediante el comando *docker node ls,* podemos visualizar los componentes del *cluster* que se encuentran corriendo y disponibles como se detalle en la figura 59*,* en el cual se puede visualizar que existen 2 nodos *manager* y 2 *worker*, uno de los nodos *manager* se encuentra en standby, y si el *manager* principal presenta fallos, el *manager* 2 tomará el control del *cluster* para continuar trabajando normalmente.

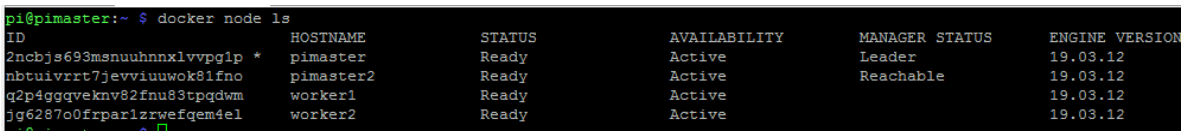

Figura 59 Estado de *cluster* con *manager* 1.

Mientras que en la figura 60 se muestra al *manager* 2 tomando el control del *cluster* debido a la falla del *manager* principal.

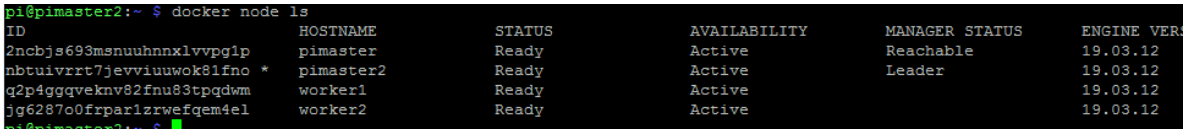

Figura 60 Estado del *cluster* con *manager* 2.

Con esto se garantiza redundancia en el funcionamiento del *cluster*, ya que si el *manager* principal falla los proceso continuaran ejecutándose lo cual será transparente para el usuario.

5.2. Prueba de acceso al almacenamiento

Para probar el funcionamiento del almacenamiento simplemente hay que digitar la dirección IP del *manager* en el ejecutar o directamente en el explorador, como se muestra en la figura 61:

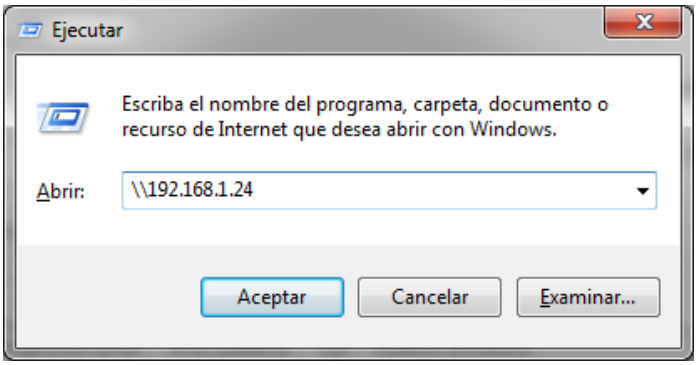

Figura 61 Dirección IP del servicio de almacenamiento a ejecutar.

Luego tendremos la pantalla de la capeta compartida como se puede visualizar en la figura 62 y para acceder a los datos almacenados en ella, el sistema pedirá usuario y contraseña para conceder el acceso.

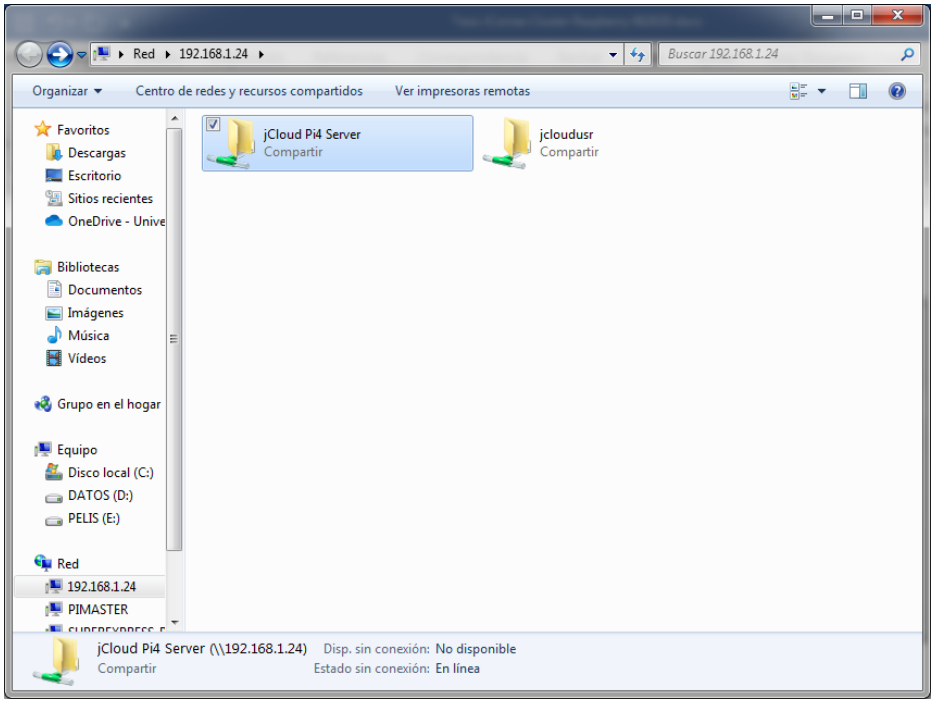

Figura 62 Interfaz del almacenamiento dentro del servidor.

En la figura 63 se ingresa las credenciales previamente creadas en la configuración de Samba (Figura 63):

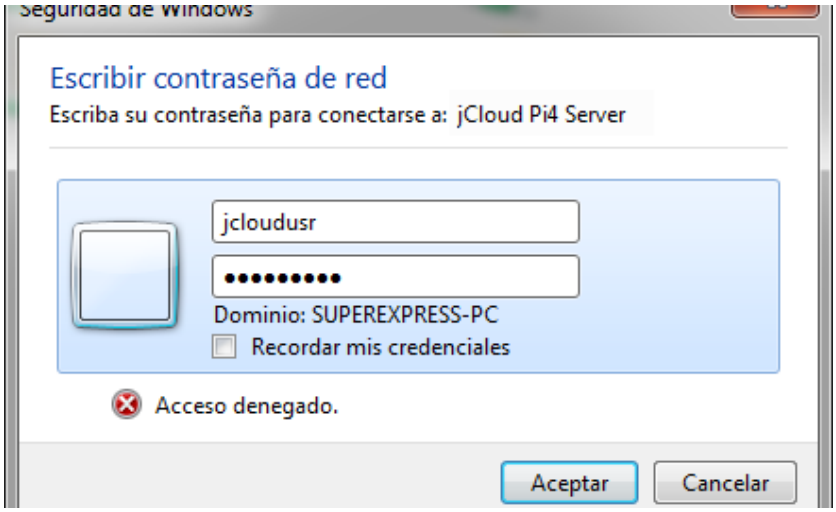

Figura 63 Pantalla para el ingreso de credenciales.

Finalmente, ingresado a los archivos almacenados en el disco duro, como se detalla en la figura 64.

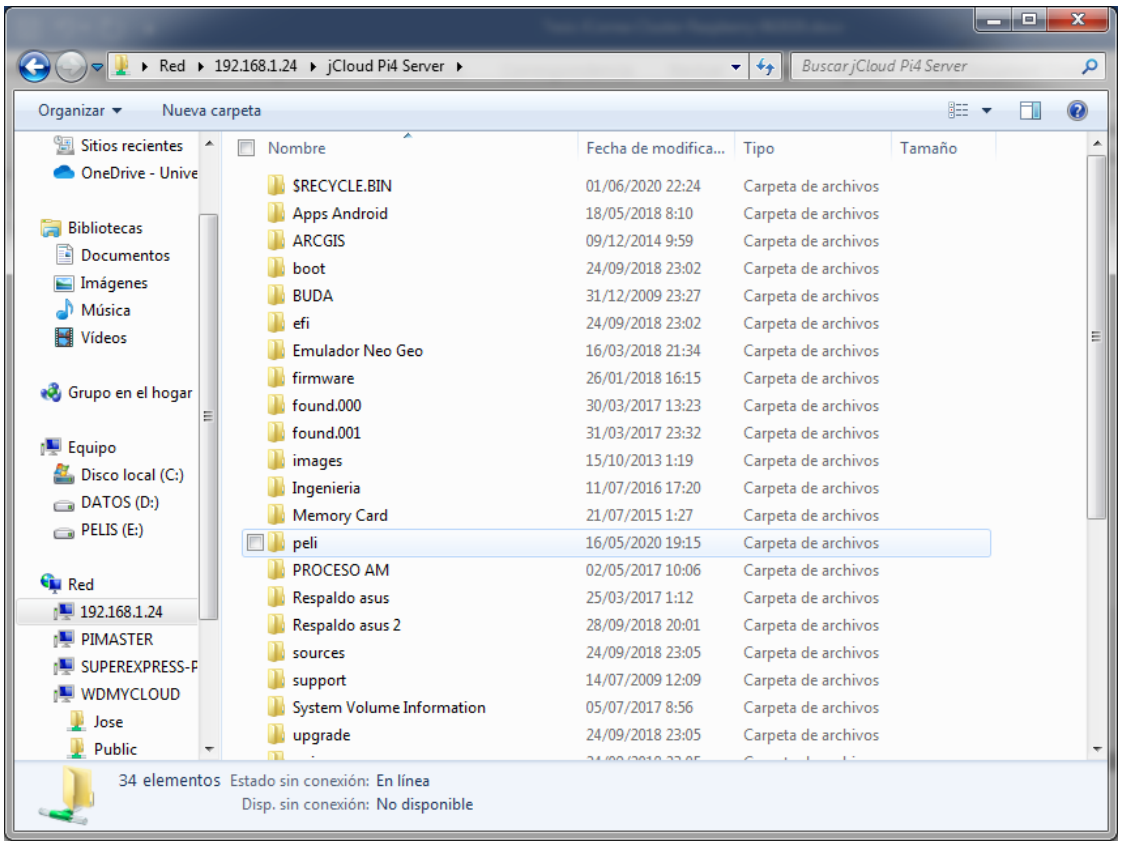

Figura 64 Visualización del contenido del dispositivo de almacenamiento.

5.3. Prueba de transmisión de datos al almacenamiento

Para probar el funcionamiento del almacenamiento desde un computador accedemos a la carpeta compartida y pegamos allí un archivo de formato .mkv que es una película HD en definición 1080 ps. Esto permite probar el tiempo y la velocidad de la transmisión

Este archivo pesa aproximadamente 2.60 GB, y su tiempo de transmisión es de aproximadamente 5 minutos y 30 segundos, y dando como resultado una tasa de transferencia de 8.16 MBps, como se puede apreciar en la figura 65.

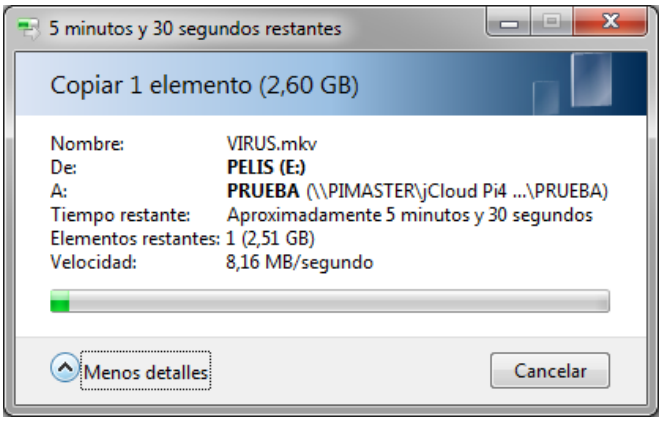

Figura 65 Detalles de la transferencia de un archivo al almacenamiento.

La transmisión del archivo fue exitosa y para verificar que el archivo si se transfirió vamos a revisar en el servidor en siguiente dirección /PiServer2/Prueba, allí realizamos un ls y podemos visualizar el archivo almacenado con su peso correcto como se observa en la figura 66.

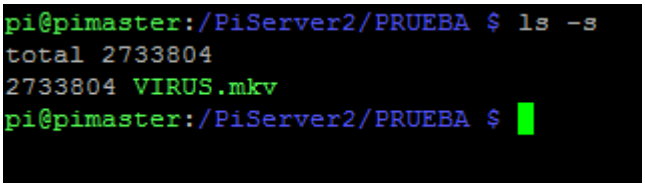

Figura 66 Verificación del archivo almacenado.

5.4. Prueba de *web service* redundante

Para verificar que los servicios creados en el sistema puedan ser redundantes ya que las réplicas deben ser creadas en cada uno de los nodos, para esto se realiza la prueba de funcionamiento del *web service* alojado en el nodo *manager* como se observa en la figura 67:

Digitamos en un navegador web: 192.168.1.24: 8080

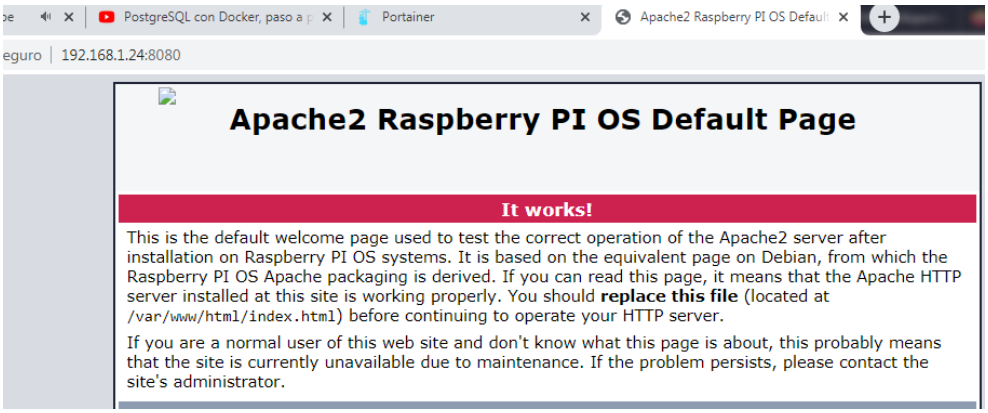

Figura 67 Comprobación del *web service* en el nodo principal.

Ahora digitando en uno de los nodos podemos verificar que el servicio está corriendo correctamente ya que en caso de que falle el *manager* los otros nodos continuaran trabajando, como se muestra en la figura 68:

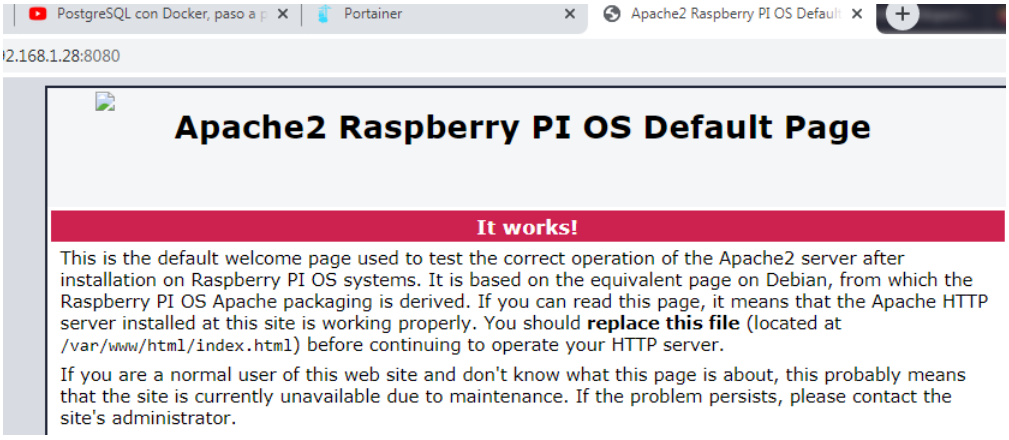

Figura 68 Comprobación de disponibilidad del *web service* en el nodo *worker.*

6. CONCLUSIONES Y RECOMENDACIONES

6.1. Conclusiones

El prototipo de hiperconvergencia presentado está basado en placas Raspberry y con la utilización de *software* libre para su funcionamiento está compuesto de 3 elementos, procesamiento o computing donde se incorpora las placas Raspberry PI 4; la red cuyos componentes son el cableado estructurado, *routers* y *switches*; y el almacenamiento que cuenta con un disco duro de 2 TB de capacidad todo esto puede ser escalable de acuerdo con la necesidad del usuario.

Se aplicó la metodología descriptiva al crear un diseño basado en diagramas de flujo y de bloques que permite entender el funcionamiento del prototipo. Con la metodología exploratoria se analizó las diferentes opciones con el fin de utilizar la mejor para obtener los objetivos propuestos. Con la metodología explicativa se detalla la implementación de cada uno de los componentes paso a paso para así facilitar que el diseño y su implementación sean fáciles de aplicar.

Todos los componentes del prototipo son escalables, por tanto, se puede agregar “n” cantidad de placas Raspberry PI y configurarlos en el *Docker swarm* para trabajar colaborativamente, cada placa sumará sus núcleos de procesamiento memoria RAM para unificarlos como recursos únicos de *swarm*, así mismo la red puede ser ampliada incorporando nuevos *switches* y *routers*, así como también se puede incorporar más discos o unidades de almacenamiento en caso de ser necesario.

En el análisis tecnológico de la hiperconvergencia, se muestra las diferentes opciones existentes en el mercado sobre este tipo de servicios y los costos de adquisición que tiene cada uno de ellos, dejando en evidencia que la implementación de un sistema de hiperconvergencia con estas placas puede representar una alternativa mucho más económica a largo plazo.

La tecnología de contenedores a diferencia de la virtualización es una tecnología relativamente nueva que están entrando con fuerza en el campo del desarrollo de los microservicios, ya que permite el rápido despliegue de estos aplicativos su desarrollo aun es completo, pero podría significar un aliado perfecto para el despliegue rápido de servicios y aplicaciones, ya que para este proyecto únicamente fue necesario cargar las imágenes de los diferentes servicios y probar su funcionamiento.

El componente de almacenamiento estuvo planteado para funcionar de manera redundante y descentralizado y para poder trabajar conjuntamente con todo el sistema. Sin embargo, por tema económico se presenta una solución centralizada en una de las placas, quedando su diseño plasmado en este documento con el objetivo de que pueda ser implementado a futuro.

A lo largo del desarrollo del este trabajo se ha evidenciado la robustez de las placas Raspberry para cumplir las funciones de cómputo y de mantener disponible los servicios durante las 24 horas del día, y el consumo energético no resulta muy elevado a diferencia de los servidores tradicionales que pueden consumir 10 veces más, considerando que el tiempo de depreciación en ambos casos es de aproximadamente 3 años.

El levantamiento y pruebas de funcionamiento dentro del centro de datos Experimental de la UDLA estuvieron contemplado en el inicio de este trabajo. Sin embargo, sin embargo, debido a la pandemia mundial de COVID-19 y el impedimento de acercarse a las instalaciones de la Universidad no se pudieron ejecutar por tanto se realizaron las pruebas de funcionamiento localmente.

Conjuntamente con el trabajo final se adjuntó una carta de compromiso por parte del estudiante para realizar la entrega del prototipo en la universidad, una vez que los laboratorios sean habilitados para el acceso a los estudiantes.

Las pruebas de funcionamiento efectuadas demuestran la viabilidad de prototipo, así que se realizó una transmisión y almacenamiento de un archivo de 2,60 GB en tan

solo 5 minutos y 30 segundos en el cual se tuvo una tasa de transferencia promedio de 8,16 MB por segundo, el cual representa un buen indicador tomando en cuenta que un archivo de este tamaño a través de internet puede tomar un tiempo aproximado de 2 a 3 horas en transferirse.

También se efectuó una prueba sobre los nodos *worker* para verificar si el *web service* que se encuentra replicado en todo el *cluster* se encuentra trabajando con normalidad en dichos nodos. Las respuestas de los nodos al momento de realizar las consultas de los servicios web es rápida y la existencia de los 4 nodos garantiza la redundancia en la red si en tal caso uno de los nodos fallase, o tuviera algún daño y fuese necesario reemplazarlo, mientras se efectúa el reemplazo que puede realizar inclusive en caliente, sin detener las operaciones.

La implementación del prototipo en general no es demasiado compleja, ya que existe documentación y guías, el costo de los elementos es bastante reducido como se demostró en el análisis de costo hecho en el capítulo 2 y de acuerdo con las necesidades de la organización el prototipo puede ser escalable tanto en las unidades de procesamiento, red y los elementos que conforman el almacenamiento.

6.2. Recomendaciones

Se recomienda que para mejorar los planes de estudio se podría incorporar las tecnologías de contenedores dentro de los temarios de la malla curricular, ya que representan una importante área de conocimiento que podría ser aprovechado no solamente con placas Raspberry sino también con sistemas ya existentes, creando la posibilidad de tener más y mejores alternativas de desarrollo de *software*.

Debido a que el diseño del prototipo es modular para obtener mejor procesamiento de datos se puede agregar más placas y configurarlas como nodos *worker*, la tecnología de *Docker swarm* permite el adicionamiento de nodos nuevos al *swarm,* dependiendo de la necesidad se podría ampliar hasta 8 placas Raspberry PI ya que el prototipo y diseño presentado tiene esa capacidad de ampliación.

Las placas trabajan con una fuente de 15.3 W, y con una intensidad de corriente de 3.1 A. Sin embargo, en el mercado nacional no existe una fuente capaz de emitir la intensidad de corriente requerida por las placas, por lo que se debería trabajar en la factibilidad del desarrollo de una fuente de estas características que permitan así agilitar el proceso de implementación del esto diseños.

Se debería trabajar con una fuente de energía persistente como podría ser un sistema de alimentación ininterrumpida o *uninterruptible power supply* (UPS) para poder garantizar la redundancia total en caso de cortes de energía y a más de que este UPS pueda cumplir las funciones de regulador de voltaje para de la evitar daños en las placas Raspberry PI por las variaciones de voltaje.

Para mejorar tanto tasas de transmisión como fiabilidad del componente de almacenamiento se debería trabajar como una unidad flash, tales como son los discos de estado sólido o SDD, mismos que pueden mejorar tanto el rendimiento de almacenamiento y transferencia de datos, así como la velocidad de ejecución de los distintos aplicativos que se almacenarán en él.

La implementación del prototipo en general no requiere comandos complejos ya que existe la facilidad de manuales y documentación que pueden facilitar este proceso. Pero adicionalmente se requiere amplios conocimientos sobre servicios web, bases de datos y programación, para tal efecto se podría asignar dicha tarea de configuración a un especialista de la organización.

REFERENCIAS

Abelinoelpaspi. (2016). *Red Pan*. Recuperado el 15 de enero de 2020, de http://abelinoelpaspi.blogspot.com/2016/02/red-pan.html

Arduino. (2020). *Arduino uno*. Recuperado el 15 de junio de 2020, de https://www.arduino.cc/

Castro, A. R. (2015). *Comunicaciones, una introducción a las redes digitales de transmisión de datos y señales isócronas.* (Primera Edición. ed.). Bogotá, Colombia: Alfaomega.

Catalina, A. (2006). *Unix / Linux, iniciación y referencia* (Primera Edición. ed.). Madrid, España: McGraw-Hill.

CentroSur. (2020). *Calcular Consumo*. Recuperado el 20 de julio de 2020, de https://www.centrosur.gob.ec/calcular-consumo/

Cisco. (2019). *Centro de datos.* Recuperado el 20 de enero de 2020, de Cisco: https://www.cisco.com/c/es_ec/solutions/smb/data-center-virtualization/infographic-basic-concepts.html#~stickynav=2l.

Cisco DC. (2020). *Solutions data-center*. Recuperado el 5 de julio de 2020, de https://www.cisco.com/c/es_ec/solutions/data-center-virtualization/index.html#~stickynav=3

Cisco Storage. (2011). *Cisco.com*. Recuperado el 5 de julio de 2020, de https://www.cisco.com/c/dam/en/us/td/docs/storage/nass/csbcdp/smart_storage/admin/localization/latin_american_spanish/NSS32X_Admin_Guide_Spanish.pdf

Cisco Systems. (2020). *Cisco Data Center Spine-and-Leaf Architecture: Design Overview White Paper*. Recuperado el 1 de julio de 2020, de https://www.cisco.com/c/en/us/products/collateral/*switches*/nexus-7000-series-*switches*/white-paper-c11-737022.html

Ciscotechblog. (2011). *Cisco Data Center Architecture*. Recuperado el 25 de enero de 2020, de http://ciscotechblog.blogspot.com/2011/10/cisco-data-center-architecture.html

CNT EP. (2018). *Data Centers de la CNT lideran en la región*. Recuperado el 2 de febrero de 2020, de https://corporativo.cnt.gob.ec/data-centers-de-la-cnt-lideran-en-la-region/

Conceptode. (2020). *Redes WAN*. Recuperado el 15 de enero de 2020, de https://concepto.de/red-wan/

Dell. (2014). *Decisiones inteligentes a la hora de evaluar y seleccionar un proveedor de infraestructura convergente*. Recuperado el 05 de julio de 2020, de https://i.dell.com/sites/doccontent/shared-content/data-sheets/es/Documents/Making-Smart-De-ES.pdf

Dell Technologies. (2020). *Descripción de los tipos de disco duro, RAID y controladoras RAID en los Servidores Dell PowerEdge y chasis del servidor Blade*. Recuperado el 15 de junio de 2020, de https://www.dell.com/support/article/es-ve/sln129581/descripci%C3%B3n-de-los-tipos-de-disco-duro-raid-y-controladoras-raid-en-los-servidores-dell-poweredge-y-chasis-del-servidor-blade?lang=es

Docker. (s.f.). *¿Qué es Docker?* Recuperado el 15 de junio de 2020, de https://www.docker.com/why-docker

Gartner. (2020). *Gartner Glosary, Data Center*. Recuperado el 1 de julio de 2020, de https://www.gartner.com/en/information-technology/glossary/data-center

Hallsal, F. (1998). *Comunicación de datos, redes de computadoras y sistemas abiertos* (Cuarta Edición ed.). Mexico D.F., Mexico: Editorial Pearson.

Hwaiyu, G. (2015). *Data Center Handbook* (X Edición ed.). New York, Estados Unidos: Editorial Wiley.

Inesem. (07 de 25 de 2020). *Los gestores de bases de datos más usados en la actualidad.* Recuperado el 20 de julio de 2020, de https://revistadigital.inesem.es/informatica-y-tics/los-gestores-de-bases-de-datos-mas-usados/

Itprice. (2020). *Tienda virtual Cisco GPL 2020*. Recuperado el 06 de julio de 2020, de https://itprice.com/es/cisco-gpl/ucs-fi-6248

Marketing directo. (2018). *20 curiosos datos sobre la red de redes en este Día Mundial de Internet*. Recuperado el 15 de enero de 2020, de https://www.marketingdirecto.com/digital-general/digital/20-curiosos-datos-sobre-la-red-de-redes-en-este-dia-mundial-de-internet

Odroid, M. (2019). *Odroid N-2*. Recuperado el 15 de junio de 2020, de https://magazine.odroid.com/es/issue/201903/

Open Media Vault. (2020). *What is openmediavault?* Recuperado el 15 de julio de 2020, de https://www.openmediavault.org/

Open Source. (2020). *Top 5 open source web servers*. Recuperado el 10 de julio de 2020, de https://opensource.com/business/16/8/top-5-open-source-web-servers

Pacio, G. (2014). *Data Center hoy, protección y administración de datos en la empresa* (Primera Edición ed.). México D.F., Mexico: Editorial Alfaomega.

Portainer. (2020). *Introduction to portainer*. Recuperado el 15 de julio de 2020, de https://www.portainer.io/

Raspberry. (2020). *Raspberry products*. Recuperado el 15 de junio de 2020, de https://www.raspberrypi.org/products/

Raspbian. (s.f.). *Raspbian FAQ*. Recuperado el 5 de julio de 2020, de http://www.raspbian.org/RaspbianFAQ

Samba. (2020). *About Samba*. Recuperado el 05 de junio de 2020, de https://www.samba.org/

Semanticwebbuilder. (2020). *Sistemas Embebidos: Innovando hacia los Sistemas Inteligentes*. Recuperado el 2 de febrero de 2020, de http://www.semanticwebbuilder.org.mx/es_mx/swb/Sistemas_Embebidos_Innovando_hacia_los_Sistemas_Inteligentes_

Silderchatz, A. (2006). *Fundamentos de Sistemas Operativos.* (Séptima Edición ed.). Madrid, España: McGraw-Hill Interamericana de España SAU.

Tiposderedesjt. (2014). *Redes MAN*. Recuperado el 15 de enero de 2020, de http://tiposderedesjt.blogspot.com/2014/10/redes-man.html

VM Ware. (2020). *¿Qué es la infraestructura hiperconvergente?* Recuperado el 5 de julio de 2020, de https://www.vmware.com/latam/products/hyper-converged-infrastructure.html

Xvisor. (2020). *What is Xvisor*. Recuperado el 5 de febrero de 2020, de http://xhypervisor.org/

ANEXOS

ANEXO 1. Instalación del Raspbian PI SO

Raspberry.org. recomienda el uso de la herramienta *Balena Etcher* cuya interfaz se visualiza en la figura 69 y 70, para realizar el formateo e instalación del SO, por tanto, lo primero es descargar el archivo e instalar el programa se explica a continuación.

Al abrir el programa tenemos un entorno grafico en donde nos pide seleccionar el archivo a cargar en la micro SD.

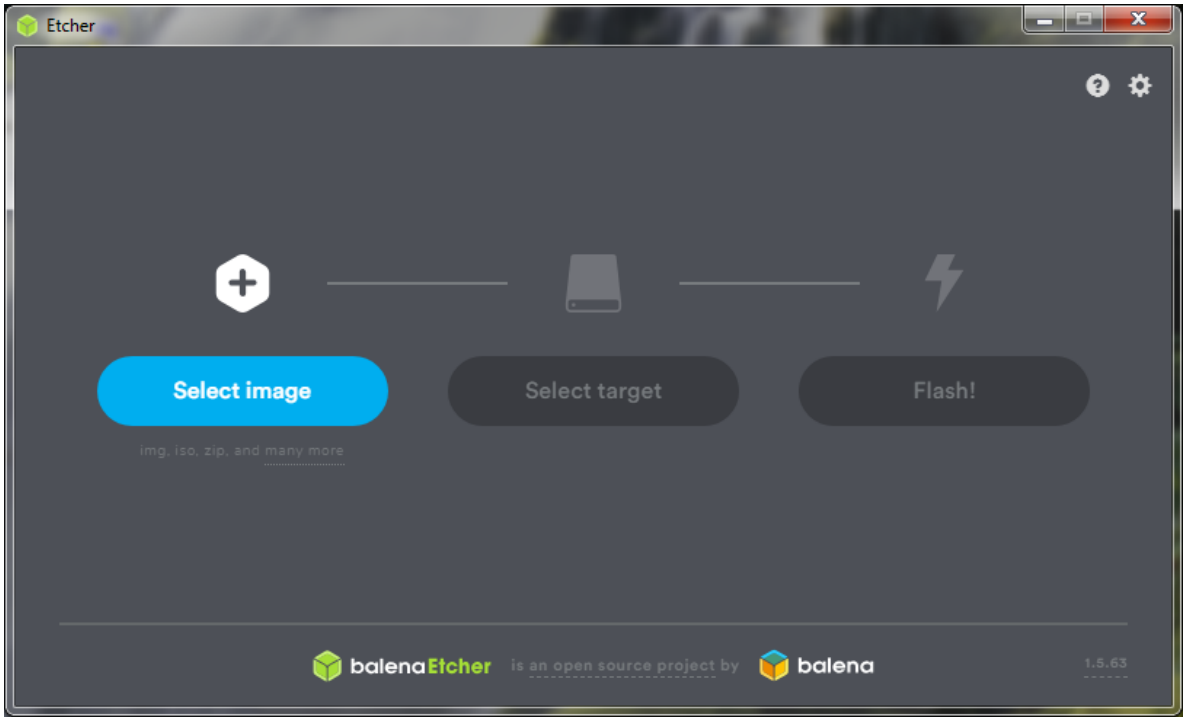

Figura 69 Balena Etcher es el *software* para la instalación del SO.

Hay que seleccionar el archivo .zip del Raspberry PI OS *Buster Lite* que se descargó anteriormente

Ahora conectamos la micro SD al computador, para ello se puede usar el adaptador SD si el computador tiene la ranura correspondiente, o se puede utilizar cualquier otro adaptador USB, ya que al momento de insertar el dispositivo el *Balena Etcher* lo reconocerá automáticamente como se observa en la figura 70.

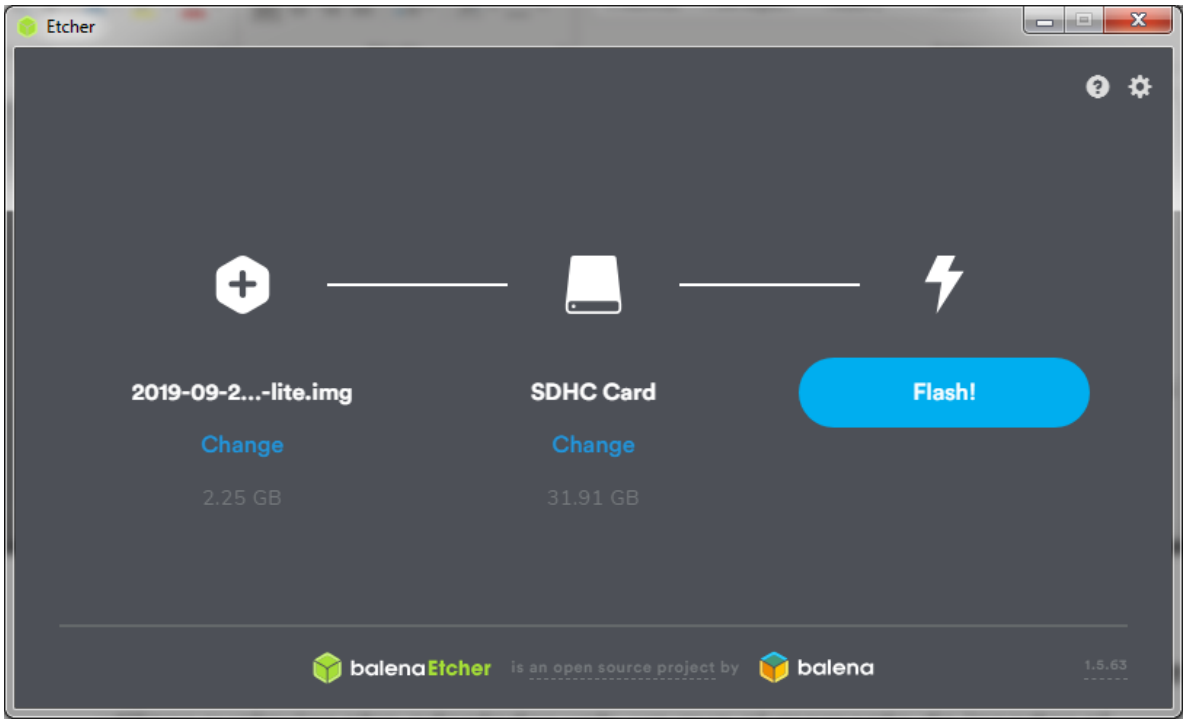

Figura 70 Pantalla de inicio para grabar el SO en la Micro SD.

Finalmente le damos clic en Flash!, y el *Balena Etcher* hará todo el trabajo esto incluye la descompresión de los archivos y la creación del booteo para que al finalizar solo sea necesario conectar la micro SD a la Raspberry PI.

ANEXO 2. Habilitación del servicio SSH

Opción 1: En la carpeta boot de la micro SD que previamente tiene instalado el Raspberry PI OS, se da clic derecho en un espacio vacío y se selecciona nuevo archivo de texto (debe estar vacío) y lo nombramos “ssh” y debe estar sin extensión, como se muestra en la figura 71:

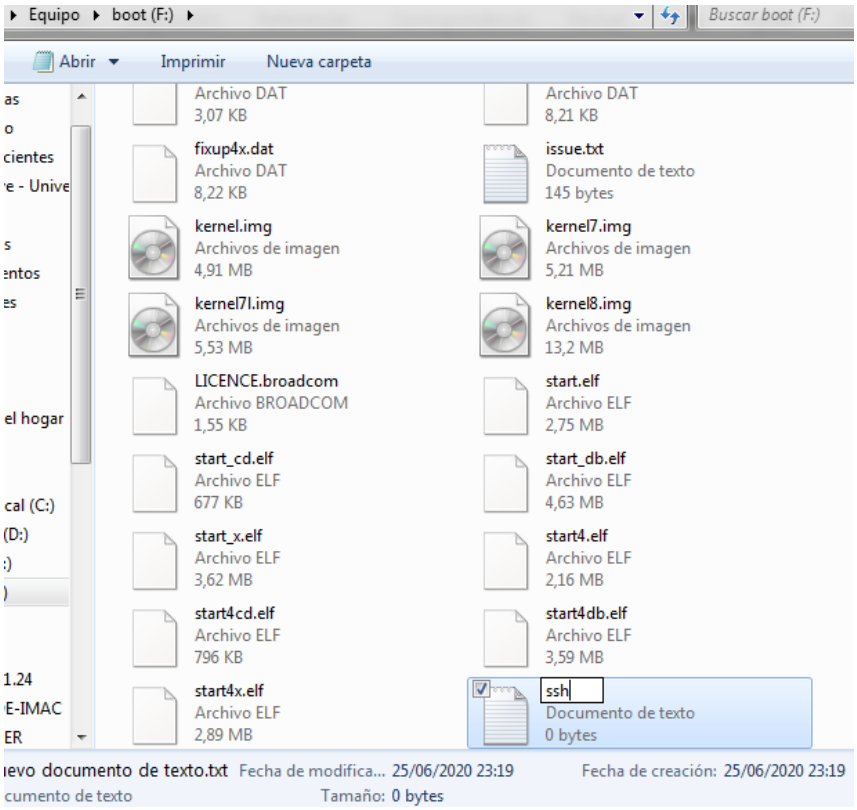

Figura 71 Creación del archivo sin extensión ssh.

Opción 2. Uso de la herramienta “raspi-config”

Al iniciar el sistema e ingresar las credenciales de usuario pi y su contraseña “raspberry”, ejecutamos el siguiente comando:

-raspi-config /Permite acceder a la configuración grafica de la Raspberry PI

Se abre una pantalla con un menú de opciones y seleccionamos la opción:

-5 “Interfacing Options”

Luego la opción:

-P2 SSH

Y finalmente seleccionamos la opción <Yes>, como se muestra en la figura 72:

- <*Yes*>

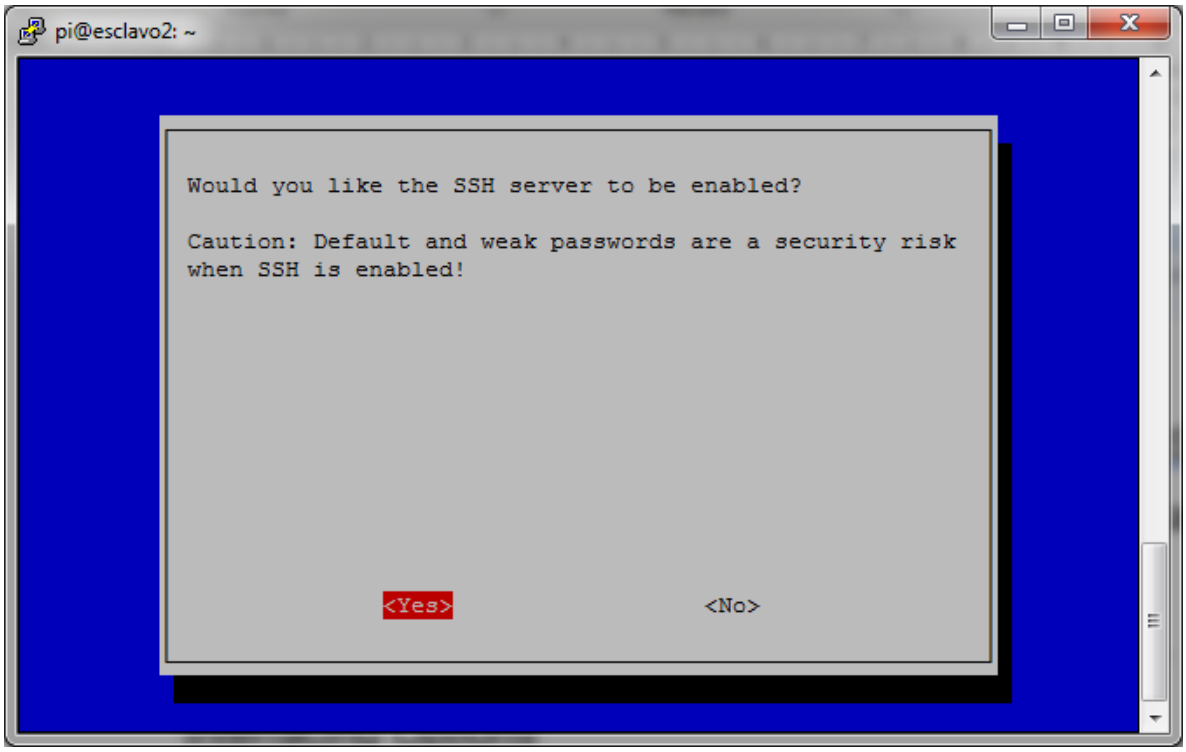

Figura 72 Panel grafico de la herramienta *"raspi-config".*

ANEXO 3. Preparación Del Raspberry PI SO

Actualizar los repositorios, como se muestra en la figura 73, con los siguientes comandos:

-sudo apt-get update / Actualiza la lista de paquetes y sus versiones disponible, pero sin instalarlos.

-sudo apt-get upgrade / Con el listado obtenido del comando anterior instala todas las actualización y paquetes disponibles.

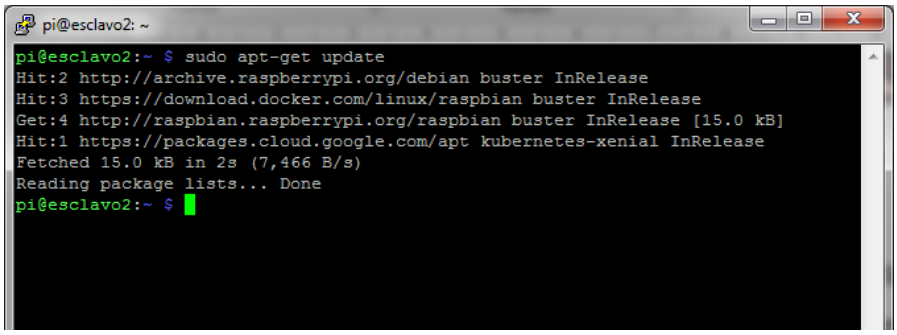

Figura 73 Actualización de librerías.

Una vez efectuado estos cambios debemos cambiar el nombre de nuestros dispositivos ya que por lo general vienen como raspberrypi, de acuerdo con la figura 74 y 75. Los cambios debemos efectuarlos con los siguientes comandos:

-sudo nano /etc/hostname

-sudo nano /etc/hosts

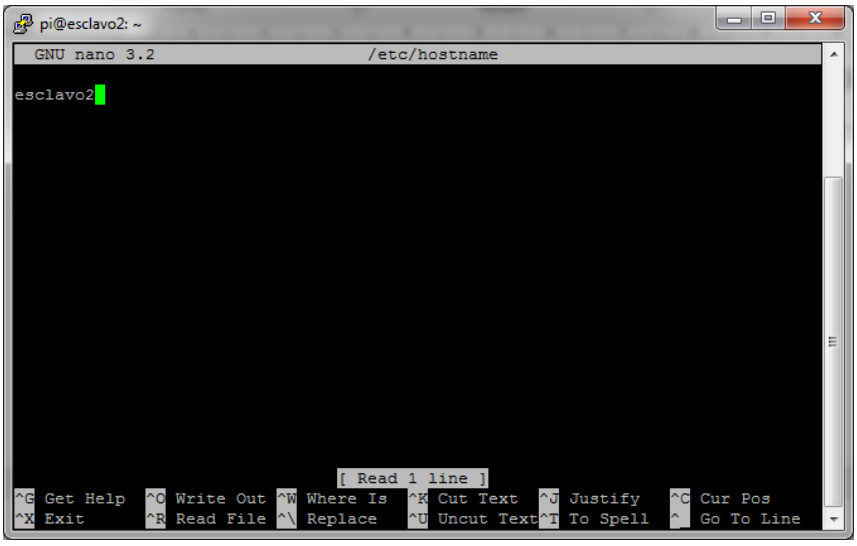

Figura 74 Actualización de nombre de las placas, archivo hostname.

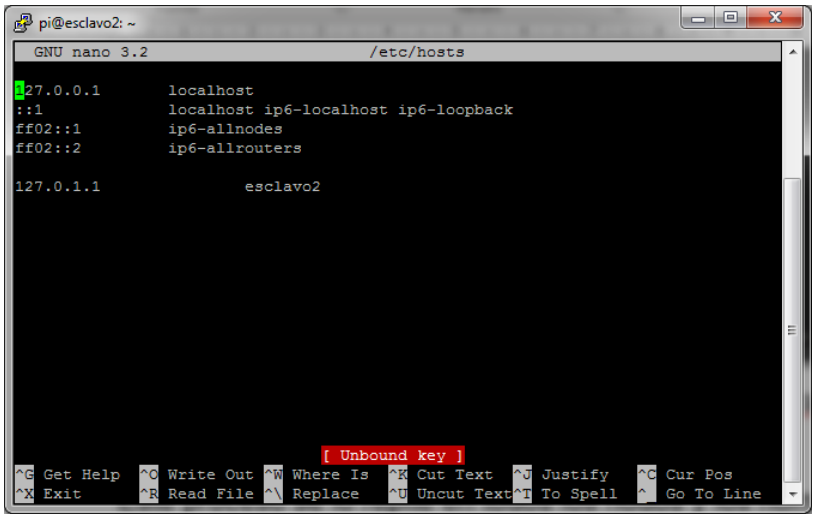

Figura 75 Actualización de nombre de las placas, archivo hosts.

Escribimos el nombre que queremos darle a nuestra Raspberry y guardamos con ctrl+o y finalmente cerramos con ctrl+x. Este proceso se lo repite en todos los nodos y los nombres definimos lo nombres de cada uno de la siguiente manera:

-pimaster (nodo *Manager*)

-pimaster2 (nodo manager secundario /worker 3)

-worker1

-worker2

ANEXO 4. Instalación de *Docker* y creación del *swarm*

Para la instalación y ejecución del *Docker s*e ejecuta el siguiente comando:

-curl -sSL get.docker.com | *sh*

Luego se debe dar al usuario pi los privilegios de administración y para ello se ejecuta el comando:

-sudo usermod -aG docker pi

Ya instalado *Docker* procedemos a crear el *cluster* con la función *Docker swarm,* con el comando *init* que se debe ejecutar únicamente en el nodo *manager*:

- docker swarm init

Con esto el *Docker* lanza un token para poder agregar otros nodos al clúster, este código se lo ejecuta en cada uno de los nodos que va a formar parte del clúster:

- docker swarm join --token SWMTKN-1-5kh4w7p84xbtaueb05mntl583pj6x2tlgj3yvx1ytqzz19qooj-9dk9n85pmjy1bmjvc2yb87cql 192.168.1.24:2377

Y también el código para poder agregar otros *manager* al clúster:

-docker swarm join-token manager

ANEXO 5. Instalación de la herramienta grafica portainer

Para la instalación únicamente se ejecuta el siguiente comando:

-docker run -d -p 8000:8000 -p 9000:9000 --name=portainer --restart=always -v /var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer

Dentro del cual estamos creando un contenedor que se va a ejecutar en el puerto 9000.

En un navegador debemos digitar la dirección IP de la Raspberry donde se instaló el Portainer, seguido del puerto 9000. Ejemplo: 192.168.1.24:9000.

En e l primer acceso Portainer pedirá crear una contraseña para el usuario *admin*, luego de esto ya ser posible ingresar a la interfaz.

ANEXO 6 Creación de almacenamiento

Diseño de RAID 5 con Open Media Vault.

Para la instalación del OMV (Open Media Vault), se ejecuta el siguiente comando:

-wget -O - https:// github.com/OpenMediaVault-Plugin-Developers/installScript/raw/master/install | sudo bash

Posterior a ello se ingresa por medio de la ip del dispositivo mediante un navegador de internet, con las credenciales por default:

Usuario: *admin*
Contraseña: *openmediavault*

Y se tiene la interfaz gráfica e inmediatamente vamos al menú de almacenamiento para visualizar los discos como se muestra en la figura 76:

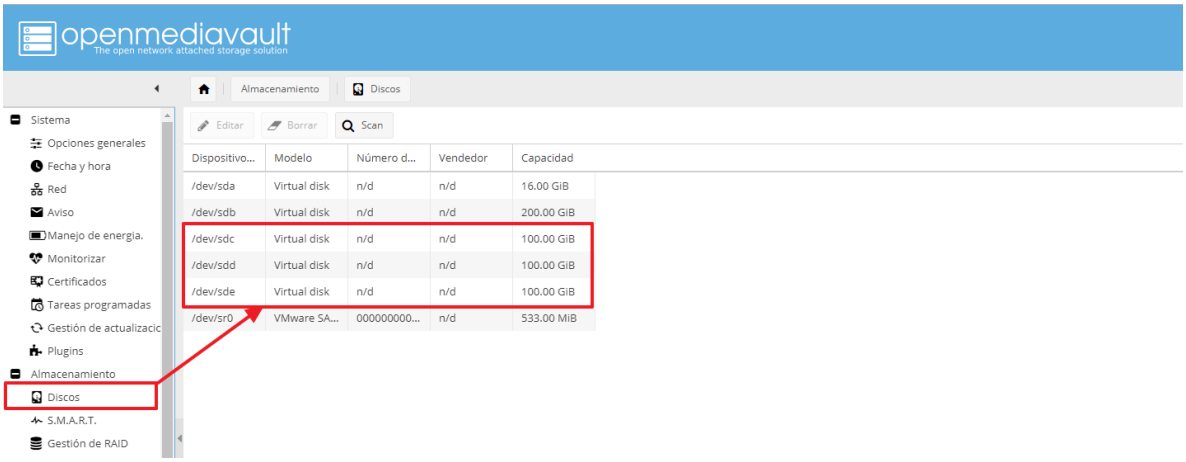

Figura 76 Herramienta Grafica Open Media Vault.

En la parte izquierda del menú se encuentra la opción Gestión de RAID, y al ingresar se selecciona la opción crear y se desplegara un menú para seleccionar los discos y el tipo de RAID (RAID 5), como se visualiza en las figuras 76 y 77.

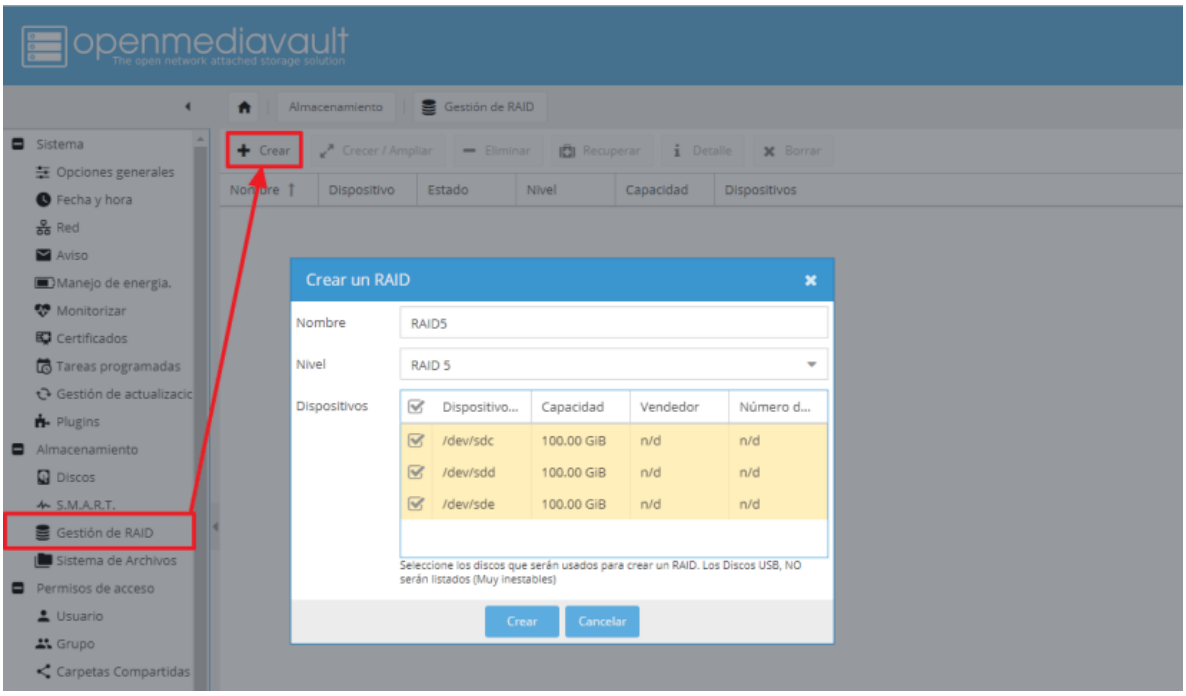

Figura 77 Selección de los Discos del RAID.

Posteriormente le damos clic en crear y pedirá confirmar la creación del RAID, se confirma y con ello iniciará la creación del RAID que puede tardar unos minutos. Posteriormente se empezarán a sincronizarse los discos y solo se aplican los cambios, como se muestra en la figura 78.

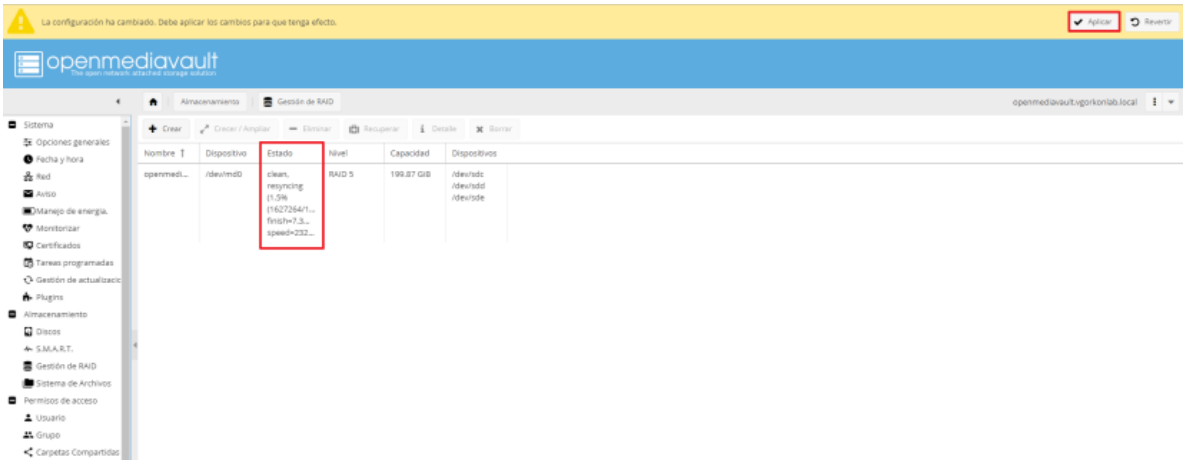

Figura 78 RAID creado.

Posteriormente se procede con la creación del sistema de archivos, y se seleccionar Crear, luego de seleccionar el RAID creado se ingresa una etiqueta y como sistema de archivos XFS, como se muestra en la figura 79.

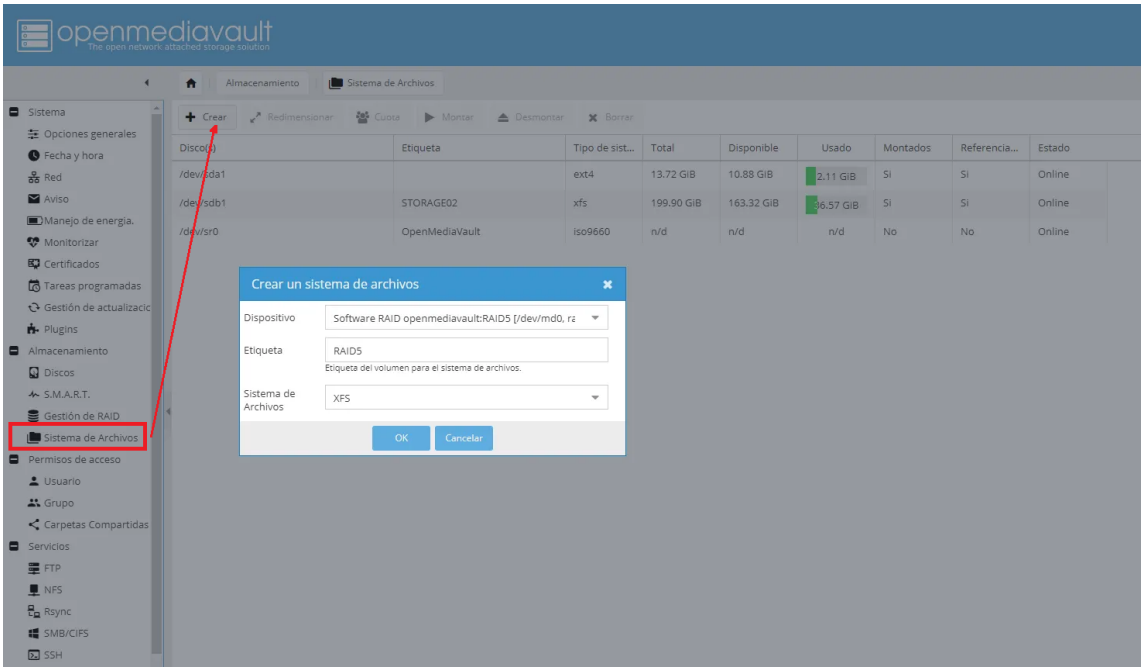

Figura 79 Creación de carpeta compartida.

El siguiente paso es montar el RAID al sistema de archivos, para únicamente se da clic en el botón montar como se observa en la figura 80.

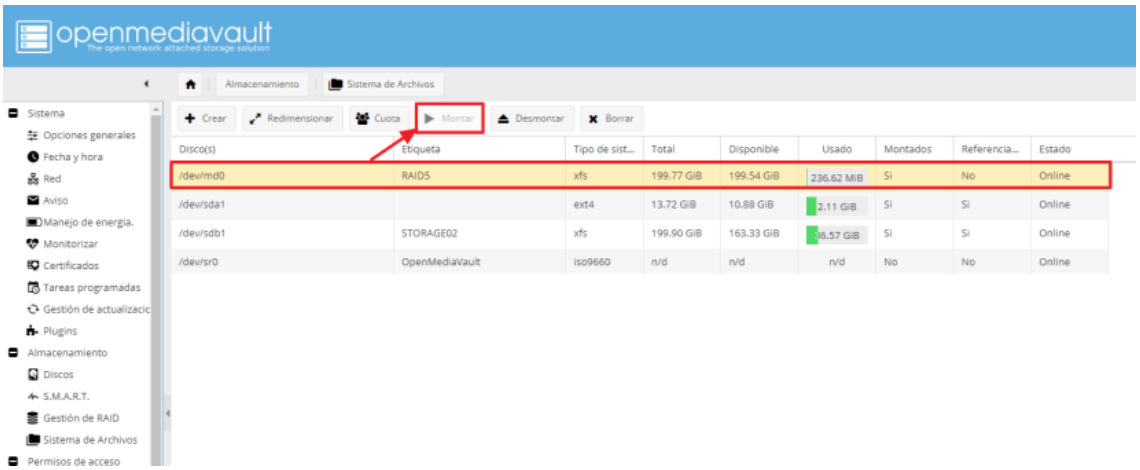

Figura 80 Montaje del RAID al sistema de archivos.

Completada la acción se procede a crear la carpeta compartida para su utilización por parte de los clientes de la red. Como se muestra en la figura 81 se agrega un nombre, se selecciona el RAID, la ruta se asigna automáticamente y los permisos que deseamos que tenga la carpeta que puede ser de lectura-escritura.

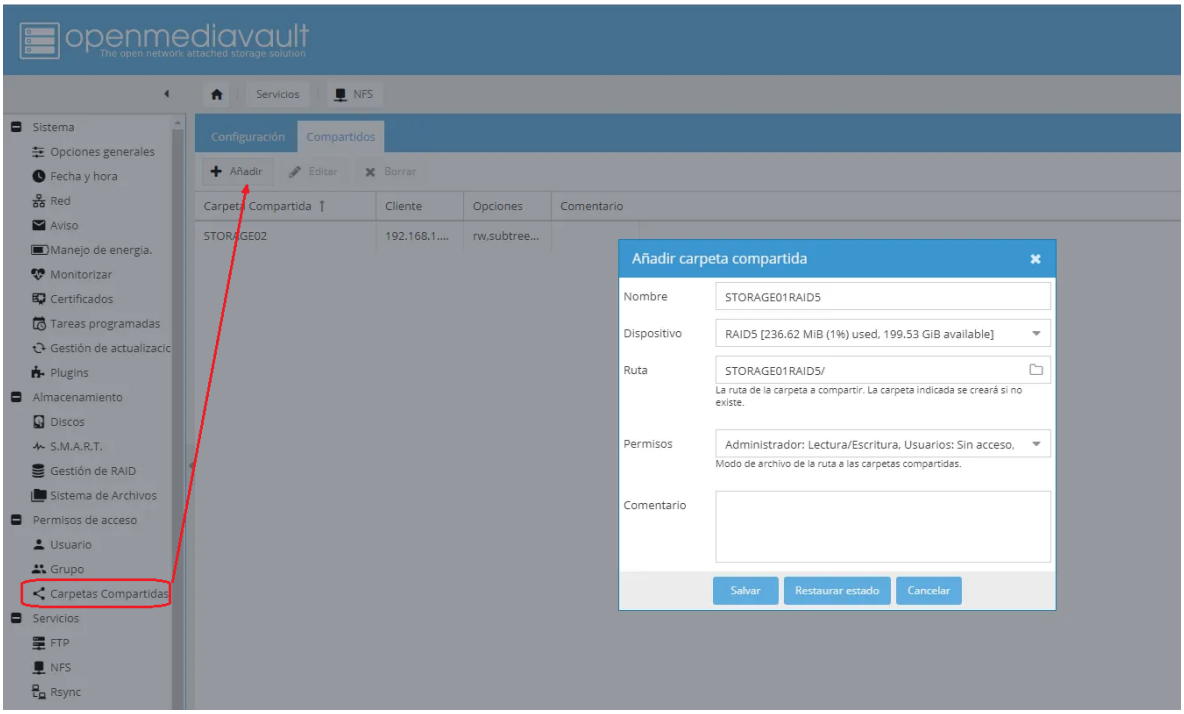

Figura 81 Se agrega carpeta compartida.

Finalmente, en la figura 82, en la opción NFS, configuramos la carpeta compartida, la red de cliente en cual queremos compartir, aquí se puede especificar algún cliente especifico o en general un grupo de clientes que estén conectados a una red, se define los permisos sean de lectura o escritura y clic en salvar para finalizar la configuración.

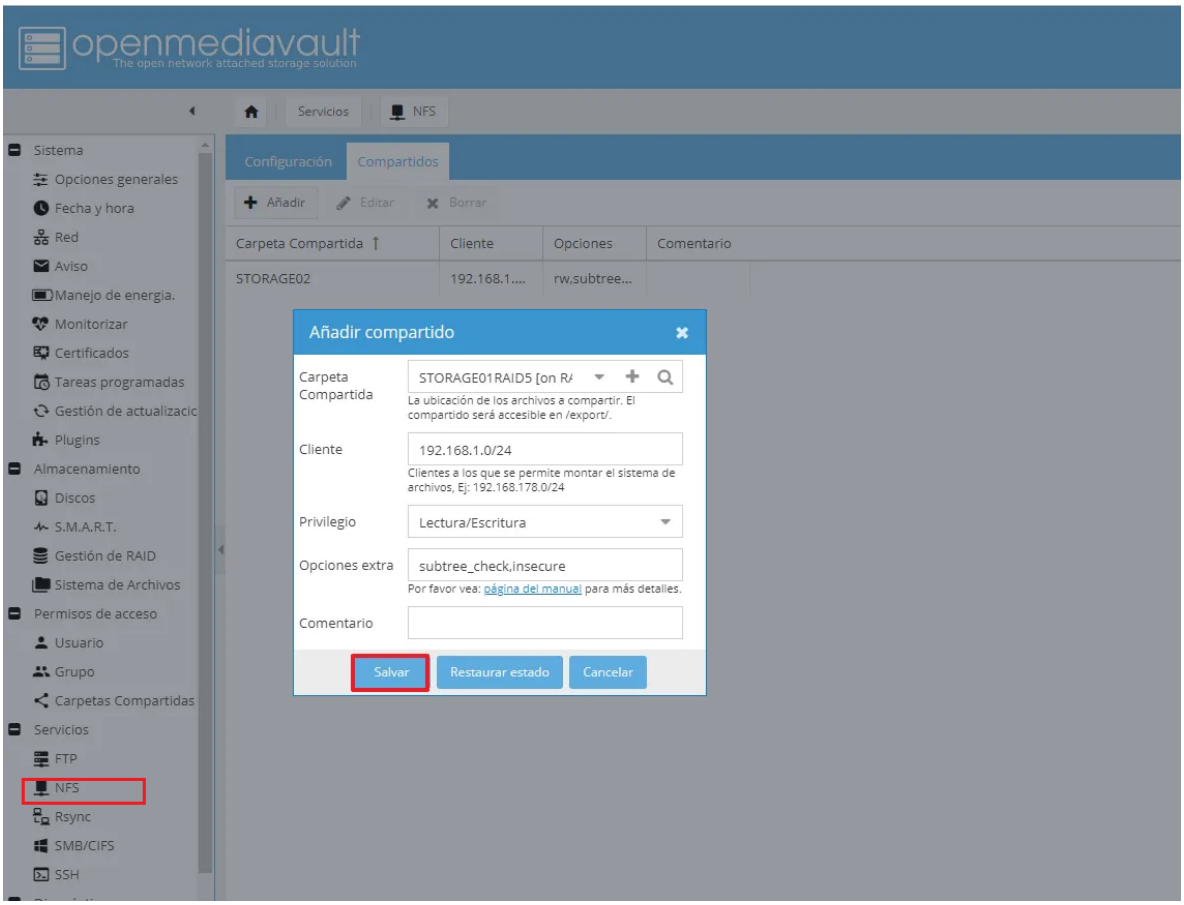

Figura 82 Definición de clientes que utilizaran el RAID mediante la carpeta compartida.

Creación de NAS mediante Samba

Para el uso de Samba y un disco duro en una distribución de Linux, se debe instalar paquetes y controladores que permitan soportar la lectura de extensiones NTFS y EXFAT que son extensión de Windows y Linux respectivamente, para ello se ejecuta los siguientes comandos:

- sudo apt-get install ntfs-3g /paquete para NTFS

- sudo apt-get install exfat-utils exfat-fuse / paquete para EXFAT

Ahora se procede a crear un directorio donde se montará la imagen del disco duro, esta carpeta se llamará PiServer2:

-sudo mkdir /PiServer2/ Se crea el directorio

-sudo chmod 777 /PiServer2 / le otorgamos todos los permisos a la carpeta.

Ahora con el siguiente comando se monta la unidad al directorio
-sudo mount /dev/sda1 /PiServer2

Para poder tener el acceso al disco desde la red utilizamos el servicio Samba y procedemos a instalarlo con el siguiente comando:
-sudo apt-get install samba samba-common-bin

Se edita el archive smb.conf y agregamos la siguiente información para identificar el nombre de la carpeta compartida en la que estará montada la imagen del disco duro y se agrega la siguiente información al final del archivo:
- sudo nano /etc/samba/smb.conf
[jCloud Pi4 Server] comment = "anyname"
path = /PiServer2
browseable = yes
read only = no
writeable= yes
create mask = 0777
directory mask = 0777
public = no
force user = root

El siguiente paso es la creación del usuario y clave para el servicio samba para acceder a la carpeta mediante el servicio Samba:
-sudo adduser jcloudusr /creación del usuario
-sudo smbpasswd -a jcloudusr /asignación de la contraseña

Finalmente reiniciamos el servicio samba para que se apliquen los cambios.
-sudo /etc/init.d/smbd restart

Uno de los problemas más comunes que puede ocurrir es que cuando la Raspberry se reinicie se pierda la conexión con el disco y sea necesario cargarlo manualmente de nuevo, para evitar esto se debe ejecutar el archivo fstab con el siguiente comando:
-sudo nano /etc/fstab

Y se copia la siguiente línea, con la cual se le indica como acción primaria que debe montar la unidad de disco duro en la carpeta PiServer2 cada que el sistema se reinicie.
-/dev/sda1 /PiServer auto defaults, user 0 2

Se guarda el archivo y se puede reiniciar y cada que se reinicie la imagen del disco será montada automáticatiamente.