GAMALIEL MARTÍNEZ IBARRA

ARQUITECTURA DE ORQUESTACIÓN PARA LA ADMINISTRACIÓN DE RECURSOS EN REDES PRIVADAS Y PÚBLICAS: CASO DE ESTUDIO ITCV.

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO
Instituto Tecnológico de Cd. Victoria

Instituto Tecnológico de Ciudad Victoria

División de Estudios de Posgrado e Investigación

TESIS

Arquitectura de orquestación para la administración de recursos en redes privadas y públicas: Caso de estudio ITCV.

Presentada por

Gamaliel Martínez Ibarra

como requisito para la obtención del grado de
Maestro en Sistemas Computacionales

Director de tesis
Dr. Claudio Castellanos Sánchez

Codirectora de tesis
Dra. Adriana Mexicano Santoyo

Ciudad Victoria, Tamaulipas, México. Marzo de 2019.

“Es importante aprender método y técnicas de investigación,
Pero sin caer en un fetichismo metodológico.
Un método no es una receta mágica. Más bien
Es como una caja de herramientas, en la que se toma
Lo que sirve para cada caso y para cada momento.”

— Ander-Egg

RESUMEN

En la actualidad en el Tecnológico Nacional de México/Instituto Tecnológico de Ciudad Victoria (TNM/ITCV) se sigue el sistema tradicional de administrar a las tecnologías de información y comunicación (TIC), existiendo la posibilidad de utilizar nuevos sistemas que brindan mejoras en los tiempos de despliegue de las tecnologías.

El presente trabajo se enfoca en el equipo de cómputo distribuido en todo el TNM/ITCV que pueda ser utilizado como servidor y que debido a su naturaleza distribuida no pueda ser utilizado como uno solo, ocasionando que el máximo rendimiento sea individual. El uso de las nuevas tecnologías pueden mejorar la administración de los recursos de cómputo actuales para lograr un rendimiento en conjunto.

Se propone una arquitectura para la administración de los recursos de cómputo utilizando redes privadas y públicas que sea escalable mediante la automatización y orquestación. Dicha propuesta se implementó en un conjunto limitado de servidores con la justa capacidad para los elementos de la propuesta.

La arquitectura funcionó como se esperaba, pero existen adecuaciones que deben ser aplicadas para el correcto funcionamiento para casos específicos como el tipo se sistema operativo utilizado, tecnologías de red disponibles, hardware de servidor.

El presente trabajo está dedicado a mi esposa Angélica, por haber sido mi apoyo a lo largo de toda mi carrera universitaria y en especial de esta maestría

AGRADECIMIENTOS

De manera especial a mi director de tesis, por haberme guiado, no solo en la elaboración de este trabajo, sino a lo largo del proceso de la maestría y haberme brindado el apoyo para desarrollarme profesionalmente y seguir cultivando mis valores.

A mis padres y a todas las personas especiales que me acompañaron en esta etapa, aportando a mi formación tanto profesional como humana.

También quiero agradecer al Instituto Tecnológico de Ciudad Victoria, directivos y profesores por la organización del programa de Maestría en Sistemas Computacionales.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE LISTADOS

ACRÓNIMOS

API Application Programming Interface

DHCP Dynamic Host Configuration Protocol

IaaS Infrastructure as a Service

IPSec Internet protocol security

PXE Pre-Boot Execution Environment

RIP Routing Information Protocol

RPV Red Privada Virtual

VLAN Virtual Local Area Network

1 INTRODUCCIÓN

Hoy en día *Big data*, *Cloud computing* e *Internet of things* (Grandes datos, Cómputo en nube e Internet de las cosas, por su traducción literal al español, respectivamente) se refieren a las grandes tendencias computacionales, siendo el *Cloud computing* la base sobre la que se sostienen las demás, pues dicha tecnología permite un mejor aprovechamiento de los recursos computacionales y posibilita la creación de servicios al estilo de pago por uso[1].

El *Cloud computing* o Cómputo en nube (o simplemente "nube", en su manera coloquial) cambia la forma en que utilizamos las computadoras personales; se pasa de la instalación y mantenimiento de las aplicaciones a la utilización de aplicaciones que no precisan de instalación ni mantenimiento, esto debido a que dichas aplicaciones se encuentran alojadas en servidores informáticos externos a las que se acceden utilizando Internet, por tanto el trabajo de mantenimiento pasa a ser responsabilidad del proveedor. Sin embargo, el Cómputo en nube es considerado como una evolución de los servicios de Internet en donde juega un papel muy importante el ancho de banda y las nuevas tecnologías de la información y de la comunicación (TIC).

Las tecnologías basadas en el cómputo en nube son utilizadas por empresas como Google, Facebook, Apple, etc., a las cuales acceden muchas personas, desconociendo qué tipo de tecnología manejan y sin preocuparse de la infraestructura utilizada. Como ejemplo de servicios en la nube tenemos: el correo electrónico, procesador de textos, hoja de cálculo, presentaciones, almacenamiento, entre otras.

1.1 DESCRIPCIÓN DEL PROBLEMA

Existen diversos problemas que afectan a la calidad de los servicios informáticos que se pueden ofrecer en las empresas así como en instituciones educativas con recursos limitados. A pesar de ello se debe lograr la administración de dichos recursos informáticos. Entre los principales problemas se encuentran:

- Falta de capacidad de procesamiento y almacenamiento de información
- Conectividad ineficiente
- Trabajo colaborativo inexistente
- Infraestructura tecnológica heterogénea

A continuación se describe cada uno de ellos resaltando cómo se presentan en las instituciones y su grado de afectación.

1.1.1 *Falta de capacidad de procesamiento y almacenamiento de información*

El procesamiento y almacenamiento de la información representan elementos importantes en las actividades diarias tanto de empresas como de centros educativos, por lo que la eficiencia del equipo informático cobra importancia para que la realización del trabajo se haga sin contratiempo. Contrario a ello, si se carece de esta eficiencia informática, las tareas de procesamiento y almacenamiento se tornan lentas y la calidad de las actividades laborales empieza a mermarse. Por lo tanto, los usuarios de equipos informáticos se ven obligados a buscar apoyo con un colega que tenga un equipo de cómputo con mejores prestaciones, o subcontratar servicios externos, o cambiar su equipo de cómputo obsoleto.

Es importante señalar que los servidores informáticos también pueden llegar a ser obsoletos, al igual que el equipo del usuario, pero estos servidores informáticos se encuentran diseñados con capacidades muy superiores a las del equipo informático del usuario.

En la actualidad, el desarrollo de software y hardware se realiza cada vez más rápido, sin embargo, las instituciones educativas con recursos limitados no pueden seguir dicho ritmo por su alto costo y la única opción es ajustarse a sus prioridades financieras.

Otro de los problemas al que se enfrentan los usuarios de equipo informático es que este no cuenta con las capacidades necesarias que le permitan trabajar adecuadamente, pues al no contar con medios de almacenamiento para respaldar la información de su trabajo tendrá que utilizar el almacenamiento con el que cuenta su equipo de cómputo, ya que si solicita ayuda al departamento encargado del equipo, el tiempo para solucionar dicho problema va desde un día hasta meses o más si es que se soluciona.

1.1.2 *Conectividad ineficiente*

Una herramienta que se utiliza en el medio laboral es el Internet, el cual es utilizado para intercambiar información de forma ágil y en mayor cantidad, situación que lo ha convertido en indispensable, pues el contar con conectividad permanente trae grandes beneficios al usuario. Sin embargo, las fallas en la conectividad interrumpen el trabajo del usuario frustrándolo cuando tiene límite de tiempo para realizar su trabajo.

La gestión básica de los recursos de una red informática, ha sido sobrepasada por los nuevos escenarios que llegan con la implementación de tecnologías actuales, puesto que requieren de configuraciones

específicas que deben de ser compatibles con ellas, por ejemplo: la telefonía VoIP, en donde uno de sus requerimientos es la entrega rápida de los paquetes de voz para asegurar una comunicación de calidad. Asimismo, el sistema de video vigilancia IP, en donde uno de sus requerimientos es su aislamiento para que no cualquier usuario acceda a ellos, por mencionar algunos de ellos.

1.1.3 *Trabajo colaborativo inexistente*

En la mayoría de las instituciones el usuario ya tiene definida una forma de trabajar por lo que para realizar sus tareas sigue un esquema definido que no ha cambiado o difícilmente cambia. Pero es natural que un proceso evolucione y con ello las tareas, por lo que las necesidades actuales no son las mismas que las de hace décadas.

Las actividades actuales están cambiando, antes una tarea era realizada por una persona, ahora más de una persona son necesarias para realizar dicha tarea, pero la solución no es el fruto de la colaboración entre las diferentes personas, es decir, no interactúan en tiempo real para resolver juntas dicha tarea.

Realizar la tarea en serie como es la forma actual, es posible pero toma tiempo; por ejemplo, una actividad en donde dos personas participan, la primera persona realiza su parte del trabajo y al terminar pasa el trabajo a la segunda persona para que lo termine. De esta forma el trabajo se concluye pero toma tiempo y se tiene el defecto que solo una persona a la vez trabaja por lo que se podría decir que la otra está ociosa a la espera de trabajo.

1.1.4 *Infraestructura tecnológica heterogénea*

Es común que exista heterogeneidad de infraestructura en las empresas o instituciones educativas, lo cual puede presentarse por diversos factores como: ofertas en la adquisición del equipo, equipo donado, equipo que en su momento solo se encontraba disponible de solo un tipo, se contaba con recurso limitado, etc. Esto trae como consecuencia una infraestructura que no fue pensada para trabajar de manera integrada.

Para aprovechar al máximo las bondades ofrecidas por el hardware existente, se podría uniformar toda la infraestructura a un solo fabricante (pero implica una fuerte inversión) para aprovechar todo su potencial. Pero esto implica un reto, ya que la infraestructura de red en la mayoría de las organizaciones se ha adquirido de diferentes fabricantes (modernos como antiguos) lo que dificulta una integración que satisfaga todas las necesidades de los usuarios.

1.2 JUSTIFICACIÓN

Hoy en día, cada vez más los departamentos de TIC en las instituciones públicas o privadas utilizan el Cómputo en nube. Se espera que estos departamentos pronto tengan al menos una porción de su tecnología en el Cómputo en nube [14].

Los beneficios de una solución de Cómputo en nube se encuentran en la optimización de la infraestructura con la que se cuenta, mejorando hasta cinco veces más el rendimiento de los servidores informáticos, lo que genera beneficios económicos y ambientales, pues se reduce el costo energético [25].

El implementar el Cómputo en nube con respecto a otras tecnologías, tiene un menor costo en cuanto a su mantenimiento. La automatización de tareas repetitivas disminuye considerablemente la carga de trabajo del administrador, pero este requiere algo que le ayude a vigilar que dicha automatización se ejecute correctamente. Es aquí en donde entra la orquestación[1] pues aporta los siguientes beneficios: despliegue ágil, gestión de múltiples aplicaciones y manejo de cargas de trabajo. Estos beneficios son utilizados para aligerar el trabajo del administrador en las tareas de mantenimiento de la nube, y a su vez, a los servicios que se ejecutarán dentro de la nube.

Los entornos de cómputo virtuales en nube habilitan la implementación de nuevos servicios de manera más rápida y aislada. Gracias a los mecanismos de seguridad que proporciona la nube se pueden realizar trabajos de instalación, configuración, pruebas y eliminación de entornos sin afectar a los demás usuarios.

Por ejemplo, se podrá virtualizar el sistema de telefonía IP en un entorno aislado, después el sistema de vídeo vigilancia en otro, los dos trabajarían de manera independiente. Cuando se prescinda de alguno de ellos la eliminación no afecta al funcionamiento del otro y se realiza desde un solo lugar. Contrario a si se realiza de la forma tradicional en donde la utilización de redes virtuales (*VLAN*, mecanismo para el aislamiento de las redes), se debe de eliminar la red *VLAN* en cada uno de los dispositivos de red en donde se utilizó[2].

Por otra parte, el software como servicio es una solución que ayuda a mitigar los problemas de los usuarios en las instituciones públicas y privadas. Por ejemplo, tendrán a disposición aplicaciones que no necesitarán actualizar, pues el trabajo de actualización será orquestado y el almacenamiento de datos brindado como servicio. Esto permitirá a los usuarios tener un control centralizado de sus datos.

Será más sencilla la escalabilidad y la implementación aislada del software colaborativo mediante el uso de los entornos virtuales en nube.

---

1 Orquestación se definirá en el capitulo 2

2 Para el caso de un entorno heterogéneo en donde las tecnologías de administración son nulas o no son compatibles entre los distintos dispositivos

Por lo anterior, la implementación del Cómputo en nube al interior de una institución resolverá los problemas antes mencionados y permitirá una alta disponibilidad de los datos. Además, por ser una implementación privada al interior de la institución, la ausencia de Internet no afectará su funcionamiento y se aseguraría que los usuarios puedan trabajar si se encuentran dentro de la institución.

1.3 HIPÓTESIS

El cómputo en nube es una tecnología cada vez más utilizada en los centros informáticos de las instituciones educativas. En algunas de ellas existe hardware disperso que no es aprovechado todo el tiempo por lo cual sus recursos son desperdiciados, la utilización de la orquestación como apoyo a los centros informáticos para la administración de la nube, es una alternativa para lograr una mejor administración de los recursos.

1.4 OBJETIVOS

Diseñar una arquitectura de orquestación para la administración de recursos en redes privadas y públicas que sea escalable basada en Cómputo en nube.

Objetivos específicos:

- Seleccionar las tecnologías *Software* y *Hardware* que permitan la orquestación de servicios mediante comunicaciones seguras con el fin de proveer entornos colaborativos de trabajo.
- Diseñar un modelo de orquestación de servicios para la administración de recursos hardware y software del cómputo en nube que permita mejorar el desempeño en entornos colaborativos.
- Orquestar servicios en la nube mediante la adaptación de la implementación del modelo de orquestación para la optimización de recursos de software y hardware que permita el trabajo colaborativo.
- Implementar la orquestación de servicios en la nube propuesta para optimizar los recursos hardware y software del TNM-Victoria.

1.5 ALCANCES Y LIMITACIONES

1.5.1 *Alcances*

La presente tesis tiene como alcance la implementación orquestada del Cómputo en nube como base para la optimización de recursos

computacionales considerando tanto el hardware como el software disponible.

Dado que la orquestación del Cómputo en nube se realiza a través de orquestadores (*Juju*, *Cheff*, *Puppet*), gestores de red (*StrongSwan*, *FreeSWAN*, *Zebra*), gestores de automatización (*Metal as a Service* MaaS, *Full Automatic Installation* FAI), gestores de La nube (*OpenStack*, *CloudStack*) y el gestor para los servicios en La nube (*Juju*, *Cheff*, *Puppet*) solo se seleccionará un quinteto factible de integrarse para la administración y mantenimiento a nivel hardware, que soporta la configuración tanto de dispositivos de almacenamiento, procesadores y memoria, logrando un crecimiento horizontal.

Se propone una arquitectura escalable de dos formas: local y remota. La primera es donde el uso de redes lógicas (VLAN) separa el tráfico de los servidores informáticos internos del tráfico de los usuarios de la institución pública o privada. La segunda forma es el uso de redes privadas virtuales (RPV) para establecer la comunicación segura entre los servidores informáticos externos.

El modelo de orquestación del Cómputo en nube seleccionado se adecuará para la orquestación de servicios que se integran como son: sistemas de gestión de cursos web (*Moodle*[3]), gestión de *big data* (*Hadoop*[4]) y aquellos que permitan almacenar y ofrecer plataformas colaborativas de trabajo para los usuarios finales.

1.5.2 *Limitaciones*

El presente trabajo se aplica en el TNM-Victoria a través de la División de Educación Superior Tecnológica a Distancia (DESTD), la División de Estudios de Posgrado e Investigación (DEPI) y el Laboratorio Interinstitucional de Cómputo Biomédico (LICB).

Para la implementación de prueba se contó con 2 estaciones de trabajo con prestaciones medianas de la DEPI, 2 servidores de la DESTD con prestaciones bajas, 1 estación de trabajo del LICB con prestaciones altas y con disponibilidad condicionada.

Por otro lado, se utiliza de forma compartida dispositivos facilitados por el TNM-Victoria y la DESTD. Sin embargo, dicha infraestructura no es exclusiva para la libre manipulación y configuración, por estar brindando servicio a diferentes usuarios tanto internos como externos.

Por lo anterior, el consumo del ancho de banda está limitado por las configuraciones por defecto con las que cuentan algunos dispositivos de red del TNM-Victoria, como es el caso de los *switch* de red, pero que no pueden ser modificados para el presente trabajo ya que la

3 Entorno de Aprendizaje Dinámico Orientado a Objetos y Modular (Modular Object-Oriented Dynamic Learning Environment). https://moodle.org/

4 Hadoop es un software de código abierto que permite el procesamiento distribuido de grandes conjuntos de datos mediante *cluster* de computadoras utilizando modelos de programación simples. https://hadoop.apache.org/

conectividad de la red no se puede resolver si los responsables de la red no realizan los ajustes a la red para asegurar la conectividad.

1.6 ESTRUCTURA DEL DOCUMENTO

A continuación se describen los capítulos que constituyen la presente tesis.

El capítulo 1 se realiza una introducción general, se describe el problema, se presenta la hipótesis y se definen los objetivos, alcances y limitaciones.

El capítulo 2 se da a conocer la nube, su definición dada por dos organizaciones de estandarización. Además se define la automatización y orquestación, se describen las comunicaciones seguras y sus respectivas características y se da a conocer el software para crear nubes, realizar orquestación y crear redes seguras.

El capítulo 3 se presenta el modelo propuesto como una solución al problema planteado, se describen sus capas y los elementos que participan y se define su funcionamiento.

El capítulo 4 se muestra una implementación del modelo propuesto y las configuraciones obtenidas.

Y por último, en el capítulo 5 se dan a conocer las conclusiones obtenidas y trabajos futuros.

2 ESTADO DEL ARTE

En el presente capítulo se abordan los conceptos para comprender: el Cómputo en nube, la automatización, la orquestación y las comunicaciones seguras en redes informáticas, así como las posibles herramientas utilizadas. Al final del capitulo se muestra una selección de herramientas para dar solución al planteamiento del problema.

La solución propuesta en el presente trabajo permite una mejor administración de los recursos de cómputo disponibles en la institución para lograr el mayor rendimiento cuya configuración propuesta pueda ser administrada de forma automatizada y orquestada. De dicha manera se pretende aprovechar al máximo los recursos de cómputo.

2.1 PRELIMINARES

A continuación se introduce la definición del Cómputo en nube, una breve reseña de las características que permiten identificar si un servicio puede ser considerado como Cómputo en nube o no, y los diferentes tipos de modelos para la implementación del Cómputo en nube. Para ello se toma en consideración las dos organizaciones internacionales que hasta el momento han estandarizado su definición: NIST e ISO/IEC.

Por otra parte, no se ha tomado en cuenta al Instituto de Ingenieros Eléctricos y Electrónicos (*Institute of Electrical and Electronics Engineers, IEEE*) pues hasta el momento solo cuenta con los grupos de trabajo P2301 y P2302. El primer grupo intenta crear perfiles para las propuestas de estándares de Cómputo en nube existentes, en forma de hoja de ruta (*roadmap*). Además, está destinado a ayudar a los usuarios en la adquisición, desarrollo, construcción y uso de productos y servicios basados en nube. El grupo P2302 trabaja en un estándar sobre la definición de una norma para topología, protocolos, funcionalidad y una administración comercial confiable de interoperabilidad e intercambio de datos de nube a nube.[31]

2.1.1 *Antecedentes*

El concepto de Cómputo en nube como lo conocemos hoy no es nuevo, ha ido evolucionando con el tiempo y con las tecnologías. Aparece por primera vez el Cómputo como servicio en 1961 con las ideas de John McCarthy y Douglas Parkhill. Después, en la década de 1990, aparece cómputo en malla en diversas instituciones de educación su-

perior en los Estados Unidos de América hasta convertirse en lo que hoy llamamos Cómputo en nube a partir de 2006.

A continuación un breve desglose de estos tres momentos.

2.1.1.1 *Cómputo como servicio público*

Uno de los principales conceptos del Cómputo en nube tiene sus raíces en la colaboración y distribución de recursos entre una comunidad de usuarios [49]. Esta idea fue introducida en 1961 por el profesor John McCarthy del *Massachusetts Institute of Technology* (MIT) en un discurso para celebrar el centenario del MIT. En dicho discurso se considera que en un futuro la computación podría ser categorizada como servicio público (*Computing utility*, Cómputo como servicio público, por su interpretación en español)[1][6].

La principal idea del cómputo como servicio público es ofrecer los recursos computacionales tal como lo harían las empresas de servicios públicos, es decir, solo pagar por los recursos ocupados tal y como se hace con los servicios de agua, electricidad y gas.[13]

Más adelante, en 1966, el tecnólogo Douglas F. Parkhill publica un libro llamado *The Challenge of the Computer Utility*. En él explora a fondo las características modernas del Cómputo en nube y su comparación con la industria eléctrica por su forma de uso de manera pública, privada, gubernamental y comunitaria [16][37].

2.1.1.2 *Cómputo en malla*

Tiempo después en la década de 1990, las universidades analizaban la posibilidad de resolver problemas complejos, utilizando grandes cantidades de equipos sumamente económicos y con software de código abierto. A esto se le conoció como computación en malla [1].

La computación en malla es una metáfora para acceder al poder de cómputo con la misma facilidad con la que accedemos a la potencia proporcionada por la red eléctrica. En la metáfora, utilizar un dispositivo es simplemente conectarlo al tomacorriente de pared, el cual enlaza a una gran infraestructura de recursos eléctricos. Enfocándose a solo utilizar el dispositivo sin consideración o importancia de cómo o dónde es generada la electricidad.

En sus inicios la computación en malla no fue del todo eficiente. Inicialmente solo se podía resolver una tarea específica a la vez, pues toda la infraestructura estaba exclusivamente dispuesta para resolver ese tipo de problema. Sin embargo, existía mucho tiempo muerto cada vez que se quería dedicar el equipo a la resolución de otra nueva tarea [1]. Para reducir los tiempos muertos se creó software conocido como *middleware* y así facilitar el acceso a los servicios de la red [37].

1 La traducción literal es Utilidad informática, pero no hace referencia a lo que fue pensado en su idioma original.

En general, la computación en malla es la integración de recursos computacionales que virtualmente pueden ser vistos como una simple y poderosa computadora, es decir, un gran colección de recursos informáticos y sistemas heterogéneos dispuestos de tal forma que se pueda resolver un grupo de tareas específicas. [7]

2.1.1.3 *Cómputo en nube*

Tiempo después, el *boom* del Cómputo en nube inicia en el 2006 cuando la empresa *Amazon* lanza *EC2*, un servicio de pago por uso sin la necesidad de un contrato: todo lo que el consumidor necesita es una tarjeta de crédito para acceder a una capacidad de cómputo escalable en nube. Después se une *Google* con *AppEngine*, una plataforma administrada en la que el usuario solo se preocupa por el código de la aplicación y no de la infraestructura que lo soporta. Se enfoca principalmente en servicios web y para dispositivos móviles. Por otra parte, *Microsoft*, con *Azure*, ofrece un completo conjunto de servicios en nube (por ejemplo: Máquinas virtuales, Aplicaciones WEB, Aprendizaje automático, *Backend* móvil y Aplicaciones remotas) para crear e implementar aplicaciones a través de los centro de datos de *Microsoft*. [29]

2.1.2 *¿Qué es "Cómputo en nube" o "La nube"?*

En el ámbito de las Tecnologías de la Información y de la Comunicación (TIC), las revistas y la publicidad han creado mucho contenido acerca del Cómputo en Nube, pero esto difiere ampliamente de lo que constituye en realidad el Cómputo en nube [12]. Esto se debe a que los proveedores de servicios en la Nube mezclan características de la Nube con características propias, dificultando su diferenciación. Esto conlleva a ver la Nube como una sola cosa. Para aclarar esto se examinará el origen.

El término "nube" proviene de una metáfora utilizada comúnmente en el mundo de las telecomunicaciones. En este ámbito se utiliza el símbolo de una nube para representar un medio de transmisión que se presume disponible para transferir información sin tener que preocuparse de la infraestructura que lo soporte [35]. La idea de utilizar el símbolo de la nube en el mundo de las telecomunicaciones se extendió para ser utilizada en el Cómputo en nube, con la idea de facilitar el concepto a las personas no apegadas a dicha tecnología.

En primera instancia se puede relacionar al Cómputo en nube como un conjunto de tecnologías y servicios. Dicho conjunto, permite trabajar los recursos informáticos de una forma más eficiente, con altos niveles de automatización, y que desde un enfoque empresarial, se le utiliza como de servicio público [1] [14].

Dos organizaciones han dado una definición formal para el Cómputo en nube, el *National Institute of Standards and Technology* (NIST), y el

International Organization for Standardization (ISO) junto con el *International Electrotechnical Commission* (IEC). La primera lo cristaliza en el estándar NIST SP 800-145 que define Cómputo en nube como:

> Un modelo para habilitar un acceso conveniente, bajo demanda y a través de la red, a un conjunto de recursos computacionales configurables (por ejemplo, redes, servidores, almacenamiento, aplicaciones y servicios), los que pueden ser aprovisionados y liberados rápidamente con un mínimo esfuerzo administrativo o con poca interacción con el proveedor de servicios [38].

Por otro lado, ISO/IEC 17788:2014 define Cómputo en nube como:

> Un paradigma para habilitar el acceso de red a un conjunto escalable y elástico de recursos que se pueden compartir, físicos o virtuales con el aprovisionamiento de autoservicio y administración a demanda [27].

Para comprender las definiciones anteriores, entiéndase por **escalabilidad** a la disposición para incrementar o decrementar los recursos bajo demanda para satisfacer las necesidades computacionales. [9], [48].

Existen dos tipos de escalabilidad, la vertical (*scale up*) y la horizontal (*scale out*). La primera involucra la adición de más recursos informáticos como procesador, memoria, disco duro, etc. La segunda se logra mediante la adición de sistemas completos. Estos reparten la carga de trabajo, logrando trabajar, en forma general, como un sistema único. Cabe aclarar que el servicio a utilizar debe soportar el trabajo distribuido [4].

La **elasticidad** es un concepto que tiene que ver con la escalabilidad. La elasticidad hace uso de la escalabilidad para incrementar o decrementar los recursos de un sistema de acuerdo a la demanda de uso, realizándose de forma automatizada [8].

El presente trabajo utiliza la definición del NIST por tener una definición más simple y a fin al presente trabajo.

2.1.2.1 *Características del Cómputo en nube*

Existen características que permiten identificar cuándo un servicio puede considerarse como Cómputo en nube. Además de esta características, cada proveedor puede ofrecer características únicas que pueden facilitar el trabajo y no ser características del Cómputo en nube.

En este punto, ambas organizaciones (NIST e ISO/IEC), reconocen las siguientes características principales que permiten identificar al Cómputo en nube:

AUTOSERVICIO A LA DEMANDA Característica que permite a un consumidor abastecerse de las capacidades del Cómputo en nube

(por ejemplo, procesamiento, almacenamiento, etc) cuando las necesite o con una interacción mínima con el proveedor. En este sentido se le habilita al consumidor la posibilidad de hacer lo que necesita y cuando lo necesita, sin requerir las interacciones de usuarios humanos adicionales.

ACCESO A RED DE BANDA ANCHA Característica que permite a los recursos del Cómputo en nube, estar disponibles sobre la red y puedan ser accesados utilizando mecanismos estándares. Esto promueve el uso de plataformas heterogéneas (por ejemplo, teléfonos móviles, tablets, laptops y estaciones de trabajo), permitiendo una mayor comodidad para acceder a los recursos, desde donde se trabaje, siempre que sea accesible desde la red.

RECURSOS COMPARTIDOS Característica que permite al proveedor del Cómputo en nube agregar recursos físicos o virtuales. Los recursos puedan ser asignados y reasignados de acuerdo a la demanda de consumo (modelo multi-arrendatario). En este sentido, el consumidor no tiene el control o conocimiento de la posición exacta de los recursos, pero puede ser capaz de especificar la ubicación en un nivel de abstracción (por ejemplo, país, estado o centro de datos). Todo lo que el consumidor sabe es que el servicio funciona. El trabajo del mantenimiento de los datos es responsabilidad del proveedor.

SERVICIO MEDIDO Característica que permite que los recursos del Cómputo en nube puedan ser supervisados, controlados, reportados y facturados (por ejemplo, procesamiento, almacenamiento, ancho de banda y cuentas de usuarios). En este sentido el cliente solo paga por los recursos que utiliza proporcionando transparencia tanto para el proveedor como para el consumidor.

ELASTICIDAD Y ESCALABILIDAD RÁPIDA Característica que permite ajustar de manera rápida y elástica los recursos del Cómputo en nube, y en algunos casos hasta de forma automática, de acuerdo a la demanda del consumidor. En este sentido, para el consumidor los recursos parecen ser ilimitados y se pueden obtener en cualquier momento ya no se preocupe por la planificación de la capacidad del sistema.

El NIST no considera la siguiente característica, pero el ISO/IEC sí:

MULTI-INTERMEDIARIO Es una característica en la que se asignan los recursos físicos o virtuales de tal manera que múltiples intermediarios, así como sus procesamientos y datos están aislados de forma inaccesible de un intermediario a otro. Por lo general y dentro del contexto de multi-arrendamiento, el grupo de intermediarios de los servicios de Cómputo en nube pertenecen a

la misma organización que demanda servicios de Cómputo en nube. Puede haber casos en los que el grupo de intermediarios implique a múltiples intermediarios de diferentes servicios de Cómputo en nube, sobre todo en los casos de una implementación pública o comunitaria. Sin embargo, dada una organización de intermediarios se puede dar el caso de tener diferentes intermediarios con un único proveedor que representa a diferentes grupos dentro de la organización.

Autoservicio a demanda	ISO/IEC	NIST
Acceso a red de banda ancha	ISO/IEC	NIST
Recursos compartidos	ISO/IEC	NIST
Servicio medido	ISO/IEC	NIST
Elasticidad rápida	ISO/IEC	NIST
Multiarrendamiento	ISO/IEC	

Tabla 2.1: Características para que un servicio o conjunto de servicios se consideren como Cómputo en nube.

La Tabla 2.1 muestra todas las características y cual estándar lo soporta, se puede concluir que las cinco primeras características en esencia son la misma idea.

La última característica de la ISO/IEC se puede entender como soporte para multi-usuario, pues se describe cómo se pueden comportar los usuarios dentro de un modelo multi-arrendamiento. Pero la NIST no define esta característica y solo se hace una mención en la característica de recursos compartidos utilizado como un modelo multi-arrendamiento.

Las características que identifican a los servicios como Cómputo en nube, pueden presentarse en diversos escenarios que son llamados modelos de servicios en los cuales los proveedores de servicios en Cómputo en nube delimitan sus responsabilidades.

2.1.2.2 *Modelos de servicios*

Según los modelos de servicios del NIST[38] y del ISO/IEC[27] se encuentran los siguientes:

SOFTWARE COMO SERVICIO (del inglés *Software as a Service*, SaaS)

Es un modelo de servicio en donde el consumidor tiene la capacidad de **utilizar** las aplicaciones que el proveedor ofrece sobre una infraestructura de Cómputo en nube. Se puede acceder a estas aplicaciones desde diferentes dispositivos mediante interfaces de cliente como pueden ser navegadores web o interfaces

de programa. El consumidor no tiene acceso o control de las capas inferiores de la capa de aplicación como pueden ser la red, servidores, sistemas operativos o almacenamiento, con la excepción de configuraciones limitadas a usuarios específicos.

PLATAFORMA COMO SERVICIO (del inglés *Plataform as a Service*, PaaS)

Es un modelo de servicio en donde el consumidor tiene la capacidad de **crear y ejecutar** aplicaciones que son creadas usando lenguajes de programación, librerías, servicios e instrumentos proporcionados por el proveedor. El consumidor no tiene acceso o control de las capas inferiores de la capa de aplicación como pueden ser la red, servidores, sistemas operativos o almacenamiento, pero tiene control sobre las aplicaciones que ejecuta y posiblemente ajustes de configuración para el ambiente de ejecución, con la excepción de configuraciones limitadas a usuarios específicos.

INFRAESTRUCTURA COMO SERVICIO (del inglés *Infraestructure as a service*, IaaS)

Es un modelo de servicio en donde el consumidor tiene la capacidad de *controlar* el procesamiento, almacenaje, redes y otros recursos de cómputo para ejecutar y controlar software arbitrario. El consumidor no tiene acceso o control a la capa más profunda que es la que habilita al cómputo en nube, pero tiene control de sistemas operativos, almacenamiento, aplicaciones y el control posiblemente limitado de componentes de red seleccionados (Ej., cortafuegos).

La ISO/IEC[27] no se limita a tres modelos de servicio, como en el caso del NIST[38] y provee una lista más amplia de modelos de servicio como los siguientes:

RED COMO SERVICIO (del inglés *Network as a Service* NaaS)

Es una categoría de servicio de Cómputo en nube en la cual se ofrece la conectividad de transporte y las capacidades de red relacionadas.

ALMACENAMIENTO DE DATOS COMO SERVICIO (del inglés *Data Storage as a Service* DSaaS)

Es una categoría de servicio de Cómputo en nube en la cual se ofrece el aprovisionamiento y uso de almacenaje de datos.

CÓMPUTO COMO SERVICIO (del inglés *Compute as a Service* CompaaS)

Es una categoría de servicio de Cómputo en nube en la cual se ofrece el aprovisionamiento y uso de los recursos de procesamiento requeridos para desplegar y correr un determinado software.

COMUNICACIONES COMO SERVICIO (del inglés *Comunications as a Service* CaaS)

Es una categoría de servicio de Cómputo en nube en la cual se ofrece la interacción y colaboración en tiempo real.

En la figura 2.1 se pueden observar los modelos de servicios primarios. Los demás servicios son una combinación o extensión de los primarios.

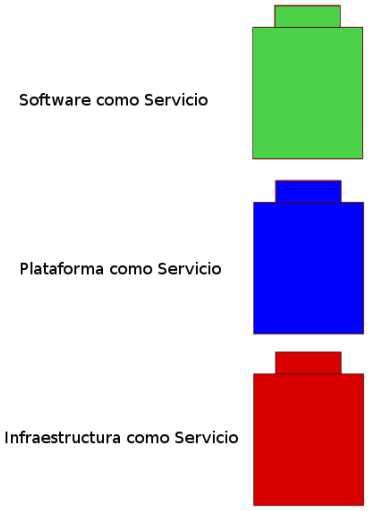

Figura 2.1: Modelos de servicios primarios.

2.1.2.3 *Modelos de implementación*

Los modelos de implementación del Cómputo en nube propuestos por la NIST[38] y el ISO/IEC[27] son los siguientes:

PÚBLICA En este modelo de implementación se prepara la infraestructura para ser utilizada por el público en general. Los recursos de infraestructura disponibles están localizados en las instalaciones del proveedor de servicios y son controlados por él mismo. La disponibilidad para los usuarios puede estar sujeta a restricciones jurisdiccionales.

PRIVADA En este modelo de implementación la infraestructura solo es utilizada por una única organización. Los recursos de infraestructura pueden estar instalados dentro de las instalaciones de la organización o fuera de ellas; además, son controlados por dicha organización. La organización puede autorizar el acceso a otras organizaciones para su beneficio.

COMUNITARIA En este modelo de implementación la infraestructura es para uso exclusivo de una comunidad de organizaciones que comparten las mismas preocupaciones (por ejemplo misión,

políticas, requerimientos de seguridad, etc). Los recursos de esta infraestructura son controlados por al menos un miembro de esta comunidad y puede existir dentro de las instalaciones de alguna de las organizaciones o fuera de ellas.

HÍBRIDA En este modelo de implementación se combina al menos dos o más modelos de implementación diferentes (pública, privada, comunitaria) que permanecen siendo modelos únicos, pero están unidos por una tecnología estandarizada o propietaria que permita la interoperabilidad y la portabilidad de los datos y de aplicaciones. Las limitaciones de la implementación híbrida refleja las limitaciones de las implementaciones anteriores.

Cada modelo de implementación del Cómputo en nube pueden ser de propiedad, administrada u operada por una empresa, una organización académica, gubernamental o una combinación de ellas.

En la Tabla 2.2 se muestran los diferentes modelos de implementación de los dos estándares.

Nube privada	ISO/IEC	NIST
Nube pública	ISO/IEC	NIST
Nube híbrida	ISO/IEC	NIST
Nube comunitaria	ISO/IEC	NIST

Tabla 2.2: Modelos de implementación del Cómputo en nube.

2.1.3 *¿Qué se entiende por automatización y qué por orquestación?*

En primera instancia, se puede pensar que la **automatización** es algo que las fábricas de automóviles utilizan en sus líneas de ensamblaje, pero no es algo que sea exclusivo de ellas. La automatización se puede aplicar a muchas cosas más, solo debemos entender como se aplica.

Desde hace tiempo se ha tratado de realizar de forma automática, con la más mínima intervención humana, los procesos repetitivos. Por ejemplo, el registro de entrada y salida del personal en una oficina. Todos los días el personal registra en un libro la hora de entrada y la hora de salida, después estos datos son analizados por una persona designada para realizar un cálculo de las horas laboradas y realizar un reporte. Con la automatización, este proceso se puede realizar con la ayuda de las computadoras, dispositivos auxiliares y el *software* que permita la comunicación entre los dispositivos y su integración. Todo un trabajo en conjunto para colectar la información, realizar el cálculo y presentar informes de manera automática[45].

Actualmente se habla de automatización de las Tecnologías de la Información (TI). Pero, ¿qué se quiere decir con automatización de las TI? Se puede entender como utilizar *software* para realizar la instalación de otros tipos *software* (sistema operativo, de aplicación, etc.), con la mínima interacción humana y reduciendo al mínimo la carga del responsable de la automatización. Esto trae como ventaja estar menos sujetos a errores humanos en las tareas repetitivas y se puede utilizar el tiempo para resolver otro tipo de problemas[10].

Pero si la automatización es mal empleada, puede llevar a la ejecución de instrucciones incorrectas y, por consiguiente, al caos: desde el mal funcionamiento del servicio, hasta la modificación de los archivos, sistemas de archivos o la propia aplicación[42].

Por otro lado, cada vez se habla más de la **orquestación**. Esta palabra primordialmente se encuentra relacionada con la música, pero actualmente está adquiriendo nuevos significados que dependen del contexto en el que se este utilizando. Aquí lo veremos en el ámbito computacional. Se utiliza la palabra orquestación para definir una forma de trabajar o un sistema que permita gestionar despliegues de *software* en una infraestructura de modo automático o semiautomático a partir de configuraciones [15].

La orquestación puede ser vista como la definición y ejecución de procesos o flujos de trabajo para gestionar la finalización de una tarea. También se puede considerar como la habilidad para diseñar y modificar gráficamente un proceso de flujo de trabajo. Esto es la clave para diferenciar el orquestado de procesos del conjunto compilado estándar de procedimientos.[51]

La orquestación se apoya en la automatización para ejecutar las tareas y se centra en la administración de procesos. La capa de orquestación permite la automatización planeada y el aprovisionamiento de tareas dentro de un ambiente de Cómputo en nube. Normalmente se enfoca en la configuración, cambios, administración de cambios e implementación de software[42].

En la figura 2.2 se muestra de forma genérica cómo es el trabajo de un orquestador. El proceso inicia en la flecha de color verde, en donde se envía un servicio a desplegar. Dentro de este servicio enviado, se incluyen los requerimientos de *hardware*, *software*, intercomunicación y parámetros de configuración. El servicio es analizado por el orquestador y de acuerdo a los requerimientos. Después (flecha amarilla) contacta a los nodos e inicia el proceso de instalación y configuración de acuerdo a los requerimientos del servicio. Una vez terminado el proceso de instalación y configuración el servicio trabaja de forma independiente utilizando la intercomunicación (flecha roja).

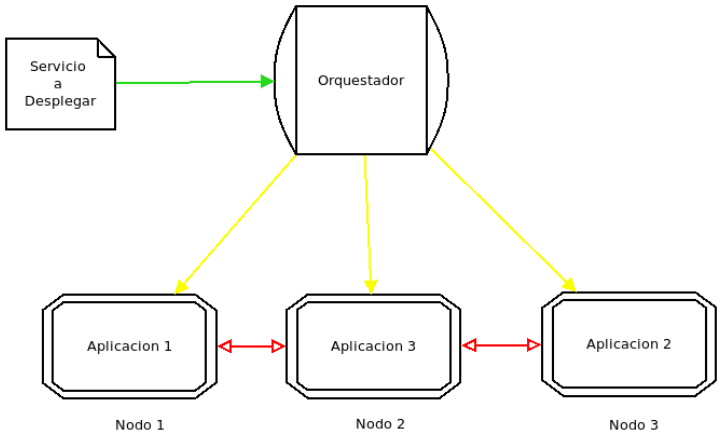

Figura 2.2: Diagrama genérico de un orquestador.

2.1.4 *Seguridad en las comunicaciones*

Hoy en día la seguridad es un factor importante que se encuentra en muchos campos de la vida moderna.

El NIST indica que la seguridad es "Una condición que resulta del establecimiento y mantenimiento de medidas de protección que permiten a una empresa cumplir con su misión o funciones críticas a pesar de los riesgos que plantean las amenazas en el uso de los sistemas de información. Las medidas de protección pueden incluir una combinación de disuasión, evitación, prevención, detección, recuperación y corrección que deberían formar parte del enfoque de gestión de riesgos de la empresa"[30].

El ISO/IEC la define como "todos los aspectos relacionados con la definición, el logro y el mantenimiento de la confidencialidad, la integridad, la disponibilidad, el no repudio, la responsabilidad, la autenticidad y la confiabilidad" [26].

Así como la NIST y la ISO/IEC han definido la seguridad, existen más organizaciones que han adaptado los conceptos y percepciones de la seguridad a sus necesidades, pero esto se escapa al alcance del presente trabajo.

En general, se puede definir seguridad en las comunicaciones como: propiedad de estar protegido frente a accesos no intencionados o no autorizados, cambios o destrucción de la información, garantizando la disponibilidad, integridad y confidencialidad [24].

Actualmente más que hablar de seguridad en las comunicaciones se hace referencia a ciberseguridad.

La aplicación de seguridad en las comunicaciones se realiza en los servidores locales y remotos, para asegurar la comunicación entre los mismos. Son dos situaciones diferentes con necesidades específicas. En el caso de los servidores locales se requiere separar el tráfico de los usuarios del tráfico de control y administración de los servidores. En el caso de los servidores remotos es necesaria una comunicación

segura utilizando redes de dominio público en las cuales no necesariamente se cuenta con la administración de ellas.

Las tecnologías a utilizar son : Red privada virtual (del Inglés *Virtual Private Network, VPN*), como seguridad del canal de comunicación y la Red virtual de área local (del Inglés *Virtual Local Area Network, VLAN*), como seguridad Interna.

2.1.4.1 *¿Qué es una Red privada virtual?*

La Red Privada Virtual (RPV) comúnmente conocida como VPN son redes privadas que utilizan la infraestructura de una red pública (para este caso Internet) para transmitir información [19].

En ocasiones también se les refiere como túneles, ya que transmiten la información por un medio público, pero aislado del resto de la información que atraviesa el mismo medio. La separación se logra mediante el cifrado de la información privada [18].

2.1.4.2 *Y ¿qué se puede esperar de la RPV?*

Según Fernández y colaboradores[18] lo que esperamos de una RPV o VPN es:

- Confidencialidad o privacidad: Los datos transferidos solo pueden ser vistos por las personas autorizadas.
- Confiabilidad o integridad: Los datos durante la transferencia no pueden cambiar.
- Disponibilidad: Los datos transferidos deben de estar disponibles cuando sea necesario.
- No repudio: Para impedir que una vez firmada la información por el origen, este se retracte o niegue de haberla enviado.

Las RPV se puede implementar de diversas formas, por *software* ó *hardware*, cada una presenta ventajas y desventajas, el optar por alguna queda fuera del presente trabajo, pero debido a las restricciones iniciales, se implementará RPV por software.

2.1.4.3 *¿Qué es una Red virtual?*

Es una tecnología que permite crear redes independientes dentro del mismo medio físico. Esto se logra mediante la división a nivel lógico, lo que permite tener múltiples redes. Esta tecnología está estandarizada en el IEEE 802.1q en la cual se define un sistema de etiquetado para identificar las VLAN y trabaja en la capa 2 del modelo OSI (del Inglés, *Open System Interconnection*) [44].

Cuando se desarrolla una red segura los siguientes aspectos deben ser tomados en cuenta:

- Acceso - Solo los usuarios autorizados pueden comunicarse.
- Confidencialidad - La información en la red permanece privada.
- Autenticación - Se asegura que los usuarios de la red sean quien dicen ser.
- Integridad - Se asegura que los mensajes no han sido modificados durante el viaje.
- No repudio - Un mensaje tiene que ir firmado, y el que lo firma no puede negar que el mensaje lo envió él.

Normalmente cada VLAN se asocia con una subred IP (del Inglés, *Internet Protocol*). Para comunicar dos VLAN diferentes hace falta un dispositivo de capa 3 llamado enrutador (del Inglés, *router*) el cual enruta el tráfico entre dos subredes IP. En la figura 2.3 se muestran tres VLAN distribuidas en un edificio de tres pisos. En dicha red, los equipos que pertenecen a la VLAN en ingeniería y no cuentan con comunicaciones con las otras VLAN.

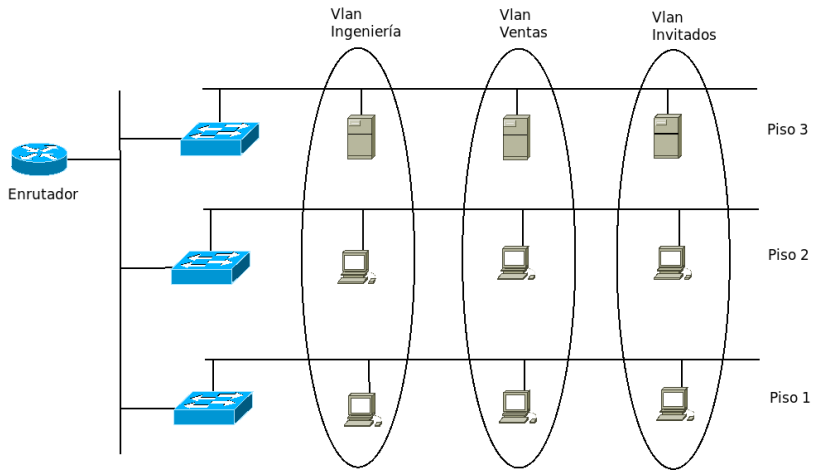

Figura 2.3: Ejemplo de VLAN.

Las VLAN trae como ventaja la movilidad, es decir, un equipo del tercer piso puede moverse al primero, mientras este siga perteneciendo a la misma VLAN no perderá la comunicación con el resto de los dispositivos.

2.2 TECNOLOGÍAS QUE IMPULSAN EL CÓMPUTO EN NUBE

A continuación se exponen las herramientas disponibles para cada tema que compone a la solución, iniciando por las que permiten crear Cómputo en nube.

2.2.1 *Tecnologías para crear La(s) nube(s)*

En el estado del arte se han encontrado los siguientes proyectos de software:

- Apache CloudStack
- Eucalyptus
- OpenStack

La lista no es exhaustiva y se ha colocado por orden alfabético y no por importancia. A continuación se describirá a grandes rasgos las características de cada uno.

2.2.1.1 *Apache CloudStack*

Es un software de código abierto basado en java para construir nubes públicas, privadas o híbridas. Se enfoca en el modelo de servicio de infraestructura como servicio (*IaaS*)[46]. Es capaz de utilizar múltiples hipervisores (*hypervisor*) como KVM, vSphere, XenServer y VMware. Soporta una arquitectura escalable y multi-nodo así como también el balanceo de cargas para la alta disponibilidad. Incluye soporte para el API (*Application Programming Interface*) de Amazon AWS, además de contar con su propia API [33].

Apache CloudStack es una solución llave en mano que administra las capas de red, almacenamiento y cómputo así como también un conjunto de características para ofrecer los servicios de infraestructura y red. Se enfoca principalmente en resolver los problemas de negocios, automatizar y desplegar los servicios de TI rápidamente para reducir los costos de operación. [33]

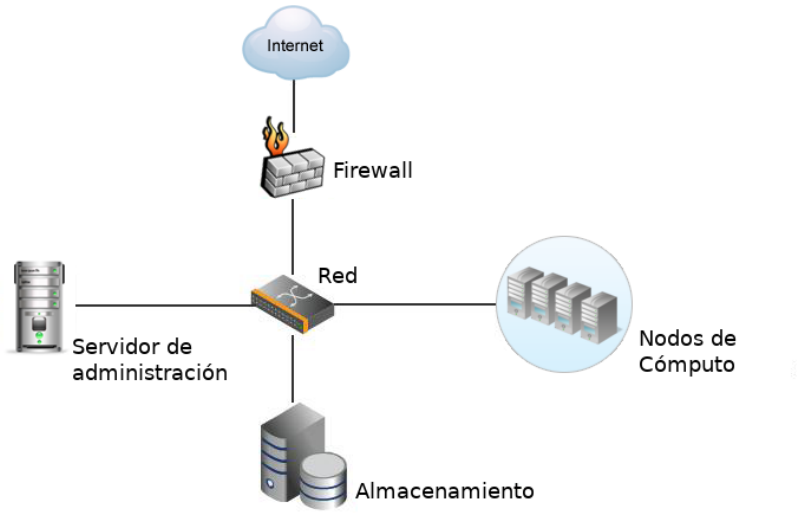

Figura 2.4: Diagrama simple de *Apache CloudStack* (tomado de [46]).

En la Figura 2.4 se muestra la arquitectura simple de implementación de *Apache CloudStack*. En ella se muestra la capa de red, almacenamiento y cómputo así como las conexiones entre ellos y hacia Internet.

Como los beneficios se puede listar[33]:

- Se enfoca principalmente en resolver problemas de negocios.
- Automatiza la entrega de servicios rápidamente y ayuda a reducir los costos de operaciones.
- Permite proporcionar cargas de trabajo estandarizadas.
- Mejora la asignación de recursos.

Como desventajas se puede listar[33]:

- No es tan personalizable su Flexibilidad y modularidad.
- Solo es compatible con Fibre Channel como almacenamiento primario a través del hipervisor.
- Las copias de seguridad y su restauración no siempre se resuelven suficientemente.

Cloud.com y Shapeblue.com son empresas que ofrecen servicios de cómputo en nube utilizando CloudStack.

2.2.1.2 *Eucalyptus*

Es un software de código abierto para construir nubes privadas o híbridas compatibles con *Amazon WEB Services*. Se enfoca en la nube híbrida la cual aprovecha la infraestructura de TI existente para construir una nube detrás del *firewall*. La fuerte compatibilidad de la API con *Amazon WEB Services* permite que el balanceo de cargas pueda ser intercambiado entre la nube pública de *Amazon WEB Services* y *Eucalyptus* [28].

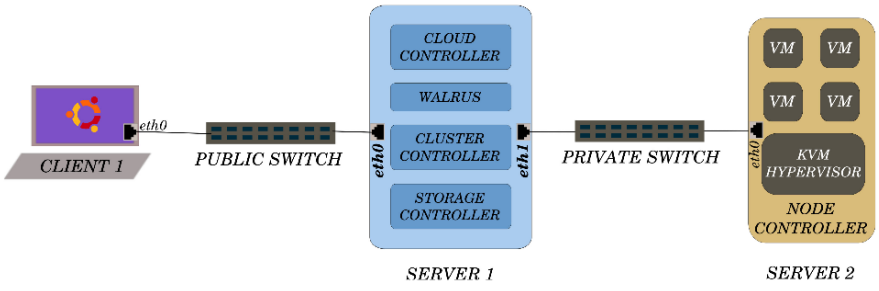

Figura 2.5: Diagrama simple de *Eucalyptus* utilizando dos servidores y un cliente (tomado de [28]).

En la Figura 2.5 se muestra un diagrama simple para implementar Eucalyptus en dos servidores, en donde en el primer servidor se encuentra todo el software de administración de la nube y en el segundo servidor se encuentran las máquinas virtuales a brindar.

Como los beneficios se puede listar[36][22]:

- Proporciona reducción en los costos de prueba.
- Proceso simplificado para realizar actualizaciones de software.
- Permite crear infraestructura de nube en sitio de forma rápida y sencilla, emulando a Amazon AWS.
- Compatibilidad con la API EC2 de Amazon.

Como desventajas se puede listar[22][23]:

- Está dirigido principalmente a las nubes privadas.
- Si se desea obtener acceso al conjunto completo de funciones, se requiere que compre una licencia comercial.
- Los algoritmos de colocación de máquinas virtuales son menos potentes ya que no permiten ningún tipo de expresión de la prioridad para la máquinas virtuales.

HP es una empresa que ofreció un servicio para crear nubes utilizando Eucalyptus, su servicio fue llamado HP Helio Eucalyptus ofrecido hasta el 31 de enero del 2016.

2.2.1.3 *OpenStack*

OpenStack es un proyecto de código abierto, principalmente programado en Python y soportado por diversas compañías del mundo (principalmente de los Estados Unidos de América). Ejemplo de ellas son Intel, Dell, AMD, Yahoo[54]. Implementa dos tipos de API: la EC2 API compatible con Amazon y la de RackSpace[46]. Además, es un proyecto modular en donde se pueden utilizar los módulos de manera independiente brindando la posibilidad que puedan ser intercambiables. Su arquitectura está basada en tres componentes principales: OpenStack *compute*, *image* y *object* que se muestran en la Figura 2.6. El primero controla las máquinas virtuales, por ejemplo la creación, inicio y apagado; el segundo brinda el *software* que la máquina virtual utilizará, es decir, el sistema operativo y sus métodos para comunicar a La nube su estado, y el tercero, se refiere al almacenamiento tanto de objetos como de bloques [47].

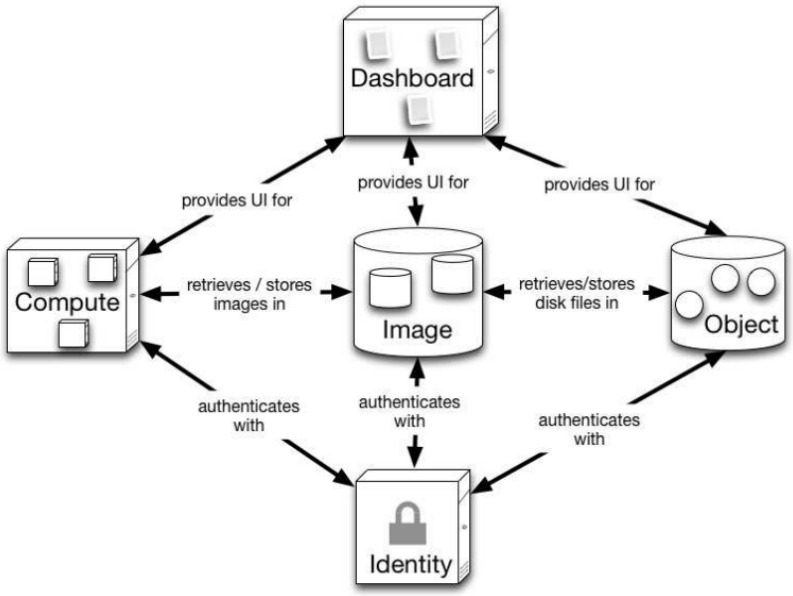

Figura 2.6: Arquitectura de OpenStack (tomado de [47]).

Como los beneficios se puede listar[34][21]:

- APIs adecuadas.
- Recomendado para grandes despliegues.
- Por tener una arquitectura fragmentada y distribuida.
- Comunidad en crecimiento.

Como desventajas se puede listar[2][34]:

- El texto claro se utiliza en la API de la red. Los puntos finales de la API de OpenStack fomentan el uso de texto claro y no existe soporte SSL/TLS disponible al momento.
- La interfaz web es sencilla y limitada.
- Instalación compleja y controlada por múltiples CLI.

Las tecnologías para crear La nube más comparadas han sido: *OpenStack, CloudStack, Eucalyptus* [50].

CloudStack y OpenStack se encuentran muy parejos en cuanto a las funciones que pueden proveer. La única diferencia es que *CloudStack* está desarrollado en *Java* y *OpenStack* en *Python y Scripts*.

2.2.2 *Tecnologías para automatización y orquestación*

Por otro lado, existen proyectos de software para orquestar:

- CFEngine

- Puppet
- JuJu

Solo por mencionar lo más utilizados.

2.2.2.1 *CFEngine*

Creado en 1993 como un software privado, después se transforma en software abierto con soporte comercial. CFEngine fue la primera herramienta en su tiempo que ofrecía implementaciones basadas en el contexto de estado deseado. Usa el lenguaje declarativo para escribir reglas que alcanzarán el estado deseado. Las reglas son definidas como promesas y de acuerdo a ellas se determina el estado final del sistema [40]. Si por algún motivo la promesa no puede ser alcanzada (por ejemplo, un error en los permisos de los archivos), CFEngine intentará arreglar los problemas de forma predefinida mediante la ejecución de subrutinas o métodos hasta alcanzar el estado ideal. [39]

Su forma de trabajar puede ser de dos formas, cliente o estructura multi-nodo. En el modo cliente se trabaja de forma independiente sin la necesidad de un servidor. En cambio, en el modo estructura multi-nodo, algunos clientes son configurados como servidores de políticas, los cuales mantienen un directorio con los archivos de configuración a utilizar. Cuando un servidor publica un cambio, los clientes se mantienen actualizados mediante el uso de una función de extracción. Cuando el cliente contacta al servidor, verifica el directorio de archivos, si encuentra una diferencia más reciente en el servidor, descarga los archivos y aplica los cambios en el sistema [40], [54]. En la Figura 2.7 se muestra un ejemplo de las comunicaciones en multi-nodo.

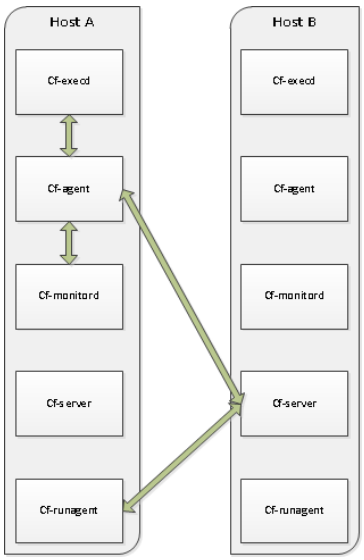

Figura 2.7: Comunicaciones en multi-nodo (tomado de [54]).

Como los beneficios se puede listar[43]:

- Altamente escalable.
- Estable.
- Herramienta flexible puede facilitar la configuración y el cambio de las especificaciones de acuerdo con las demandas del cliente y del sistema.

Como desventajas se puede listar[43]:

- Instalación compleja.
- Difícil de configurar.

2.2.2.2 *Puppet*

Puppet fue introducido en el 2003 por *Reductive Labs*. Utiliza un lenguaje de dominio específico para los archivos de configuración, por lo cual se enfoca en las tareas a realizar a través de la capa de abstracción del sistema operativo. Esta herramienta fue escrita como alternativa a *CFEngine 2* y trata de enfocarse en la usabilidad de lo que el usuario desea hacer con la herramienta en lugar de como debería de usarse[40]. Los archivos de configuración son referenciados como manifiestos y se encuentran escritos en el lenguaje *Ruby* o en una versión simplificada. Muchos de ellos se encuentran en la *Puppet Forge* y son de código abierto.[39]

La forma de trabajar puede ser en dos entornos: sin servidor o cliente-servidor. En el entorno cliente-servidor el servidor es llamado *Puppet master* y el cliente *Puppet agent* (también llamados **Nodos**). El servidor compila los manifiestos en catálogos que después son aplicados a los clientes [40].

En el entorno sin servidor el cliente compila los manifiestos y los aplica a sí mismo[54]. En la Figura 2.8 se muestra el flujo de trabajo.

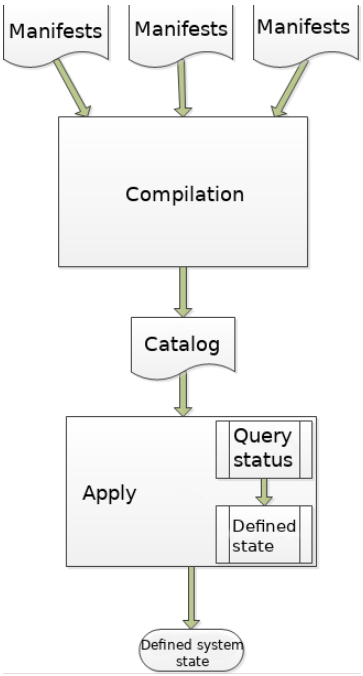

Figura 2.8: Flujo de trabajo de *Puppet* (tomado de [54]).

Como los beneficios se puede listar[43]:

- Fácil instalación.
- Altamente escalable.
- Estable.
- Herramienta flexible puede facilitar la configuración y el cambio de las especificaciones de acuerdo con las demandas del cliente y del sistema.

Como desventajas se puede listar[43]:

- Falta de soporte.
- El lenguaje Ruby puede ser difícil de comprender.
- Puede no ser adecuado para implementaciones pequeñas.

2.2.2.3 *JuJu*

JuJu, antes conocido como *Ensemble*, es un software de código abierto para la configuración de servicios y como herramienta de despliegue desarrollado por *Canonical*. Principalmente se encuentra enfocado en sistemas que ejecutan *Ubuntu Linux* y tiene compatibilidad con máquinas físicas, contenedores *Linux* y entornos en nube como *Amazon, Eucalyptus y OpenStack*. Los archivos de configuración utilizados por *JuJu* son llamados *charms* y son compartidos a través de un catálogo público llamado *Charm Store*. Estos archivos pueden ser escritos

en cualquier lenguaje que pueda ser interpretado por *Ubuntu*. Esto posibilita su combinación con *CFEngine o Puppet* [39].

Su forma de trabajo es del tipo cliente-servidor. El cliente opera en línea de comandos o interfaz gráfica (CLI ó GUI) para configurar y envíar el *charm* al servidor, llamado Controlador *JuJu*. Este controlador inicializa los nodos necesarios de acuerdo al *charm* enviado e instala y configura el software indicado por él. Un nodo puede ser un contenedor o una máquina virtual que depende del entorno en donde se encuentre.

Al finalizar el proceso el servidor notifica al cliente que el *charm* ha sido desplegado. En la Figura 2.9 se puede observar al cliente, el servidor y los nodos, en donde el servidor, es decir, el controlador *JuJu*, se encuentra en el mismo entorno que los nodos. Para este caso se encuentra dentro de La nube, y el cliente se encuentra fuera de él[5].

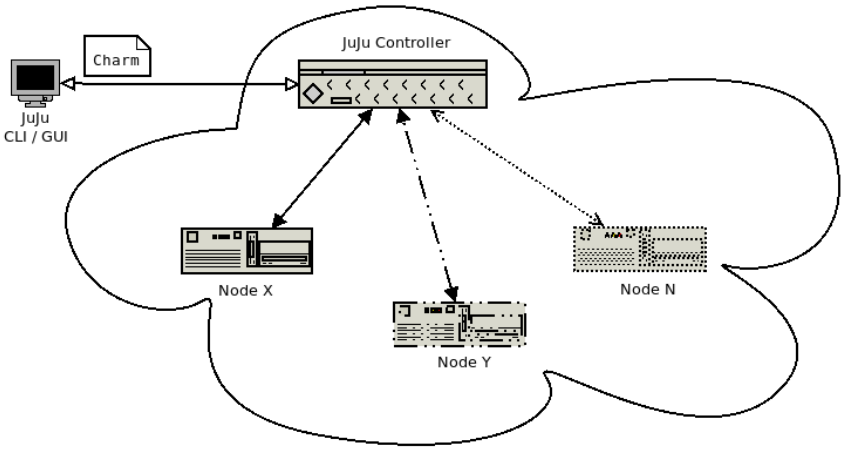

Figura 2.9: Diagrama de *JuJu*.

Como los beneficios se puede listar[41][3]:

- Los *scripts* se pueden escribir en cualquier idioma ejecutable por ubuntu.
- Está orientado a la integración y despliegue de servicios.
- Funciona sobre servicios en nube o sobre servidores físicos mediante la comunicación con otro software llamado MaaS.

Como desventajas se puede listar[3]:

- En caso de falla en la implementación de la receta el administrador debe de resolver los problemas para poder continuar.
- Juju carece de varias de las características más avanzadas de los sistemas de aprovisionamiento de servidores como Chef, Puppet, Ansible, etc.

2.2.3 *Tecnologías para comunicaciones seguras*

Los proyectos de software más conocidos para comunicaciones seguras son:

- OpenVPN
- OpenSwan
- StrongSwan

2.2.3.1 *OpenVPN*

OpenVPN es un proyecto de código abierto para RPV del tipo *Secure Layer Sockets*. Este utiliza ampliamente el protocolo SSL/TLS *(Transport Layer Security* para manejar la creación de túneles y elementos criptográficos para crear la RPV que son del mismo tipo que las creadas por IPSec. *OpenVPN* no opera a nivel de *kernel*, sino en el espacio de usuario. Esto es mediante la creación una interfaz virtual a la que puede acceder sin dependencia del *kernel* lo que permite ser un poco más seguro y estar menor propenso a errores de diseño [32].

Como los beneficios se puede listar[17][20]:

- Se puede utilizar una lista de revocación de certificados para evitar que una conexión se realice para el caso en el que un dispositivo sea comprometido.
- Puede operar a través del *firewall* con tan solo un puerto abierto.
- Puede utilizar interfaces virtuales que permiten reglas de *firewall* y enrutamiento muy específicas.
- Es compatible con sistemas operativos Linux, Mac y Windows.

Como desventajas se puede listar[17]:

- No es compatible con IPSec, e IPSec es un estándar.
- La mayoría de los dispositivos de red poseen compatibilidad con IPSec.
- Existe poca documentación para escenarios complejos.

2.2.3.2 *OpenSwan*

OpenSwan es un *fork* de la primera implementación libre de *IPSec* llamado *FreeSwan*, proyecto que finalizó en el 2004. *OpenSwan* inicia en noviembre del 2003 con un grupo de voluntarios y algunos miembros de *FreeSwan*. Su misión es más de carácter comercial ya que preserva los ideales de *FreeSwan*.

La *IPSec* se definió en el RFC2401 como una colección de estándares que tratan, mediante el uso del cifrado, garantizar la autenticidad y en casi todos los casos también para garantizar la confidencialidad del contenido de los paquetes IP [52].

Como los beneficios se puede listar[20]:

- Soporta Nat-transversal en modo transporte y en modo túnel.
- Soporta configuración cliente/servidor con IP dinámica.
- Interoperabilidad con varios estándares de hardware y software.

Como desventajas se puede listar[20]:

- OpenSwan no soporta IKEv2.
- No dispone de interfaz de cliente a nivel de usuario, pero es compatible con clientes VPN genéricos

2.2.3.3 *StrongSwan*

StronSwan se basó originalmente en el proyecto *FreeSwan*, pero se reescribió por completo y, por lo tanto, no comparte ningún código con su antecesor. *StrongSwan* soporta los protocolos de intercambio de llaves *IKEv1 y IKEv2*. El protocolo *IKE* es responsable de la fase de establecimiento de la clave y de la negociación de los algoritmos criptográficos entre los puntos finales de comunicación. *IKEv2* fue diseñado para con nuevas funciones y corregir algunos problemas encontrados en la versión anterior. Este proyecto se dirige exclusivamente al nuevo protocolo *IKEv2* sin consider a *IKEv1* [53]

Como los beneficios se puede listar[20]:

- Se conecta con productos de Alcatel-Lucent, Cisco, Juniper, Nokia, SonicWall entre otros.
- Soporte para "Online Certificate Status Protocol" (OCSP, RFC 2560).
- Soporta IPSec con NAT-trasversal (RFC 3947).
- Soporta el uso de IP virtuales vía configuración estática y modo de configuración IKE.

Como desventajas se puede listar[20]:

- Puede ser bloqueado por *firewall*.
- La instalación y configuración cliente/servidor es compleja.

2.3 CONCLUSIÓN

A pesar de que existen estándares para el Cómputo en nube, sus principales proveedores como *Amazon, Google y Microsoft* siguen estándares propios con lo que se pueden crear dependencias con un solo proveedor [11]. Se han presentado dos instituciones el *NIST* y el *ISO/IEC* las cuales han brindado una definición para el cómputo en nube, pero la *IEEE* sigue desarrollando el estándar, tal vez éste actualice elementos que no se tomaron en cuenta o que han surgido con el paso del tiempo pero primero se deberá esperar a que sea liberado. El uso de la automatización y orquestación se vuelve una actividad importante para optimizar el tiempo que se dedicaba a las tareas repetitivas, como la instalación de la nube y parte de su administración.

Se presento software para crear computo en nube, en donde cada uno se especializa en diferentes enfoques, siendo *OpenStack* una alternativa viable para su implementación por su modularidad. Para la automatización y orquestación se perfilan *JuJu* y *MaaS* por su trabajo en conjunto, el cual logran un mayor alcance desde el encendido del equipo hasta la instalación del software orquestado. Y para el establecimiento de comunicaciones seguras, se identifica a *StrongSwan* por su soporte para IKEv2 y su compatibilidad con soluciones en hardware.

3

MODELO DE ORQUESTACIÓN DE SERVICIOS EN LA NUBE

De manera global, la implementación del modelo de orquestación de servicios en La nube tiene la siguiente secuencia:

1. El desarrollo se inicia estableciendo y asegurando las comunicaciones, esto se hace utilizando las tecnologías de VLAN y RPV que se aplican en dependencia del contexto, si fuesen local o remoto.

2. Enseguida se instalan y configuran las tecnologías de automatización sobre las comunicaciones seguras. Con ésto se logra el encendido, la instalación y configuración del sistema operativo de los servidores que se utilizarán para La nube.

3. Inmediatamente después se instala y configura el orquestador para La nube, éste se encargará de instalarla en el *hardware* disponible y de escalarla de acuerdo a las necesidades del administrador. En este punto el orquestador trabaja en conjunto con el automatizador para realizar las tareas de instalación de La nube.

4. Una vez instalada La nube, el administrador configura las cuentas para los usuarios y los recursos en La nube que podrán utilizar.

5. Al final, el usuario podrá instalar el mismo orquestador que utilizó el administrador para orquestar los servicios que necesite para sus proyectos.

En los siguientes secciones se detallará cada uno de los incisos.

3.1 ELEMENTOS DEL MODELO DE ORQUESTACIÓN

El modelo se encuentra conformado por los bloques que se muestran en la figura 3.1.

En A) se encuentran las redes, sin ellas no sería posible establecer una comunicación entre los diferentes componentes del modelo.

En B) se localiza todo el *hardware* y el conjunto de tecnologías que permiten manejarlo de una manera automatizada, de otra forma la mayor parte del trabajo se realizaría de una forma manual.

En C) se encuentra el orquestador quien es el encargado de realizar el trabajo de instalación de La nube en el *hardware* disponible además de gestionar su escalabilidad.

En D) se sitúa la nube. Esta ofrecerá servicios al estilo IaaS y mediante esta se encuentra la optimización de recursos de *hardware*.

Y en E), se vuelve a colocar un orquestador el cual, además de ayudar al administrador a automatizar La nube, se utiliza para ayudar a los usuarios de La nube. Dicho de otra manera, el software que crea el administrador para el orquestador puede ser utilizado por el usuario, beneficiando a ambos.

Figura 3.1: Bloques del modelo propuesto.

3.1.1 *Redes*

En el presente trabajo se llama *host* al equipo de cómputo físico disponible, este puede ser una computadora personal, estación de trabajo o servidor. Se llama puerta de enlace (del Inglés *gateway*) al equipo que se encarga de proveer el servicio de internet y puede ser un enrutador físico (del Inglés *router*), un equipo de cómputo físico o virtual con software con funciones de enrutador.

En la sección anterior se hablaba de una dependencia del contexto, que pueden ser local o remota, a continuación se explicará la configuración local.

3.1.1.1 *Configuración local*

La nube a crear consta de dos partes: el núcleo y la parte extensible.

El núcleo es el *hardware* y *software* mínimo para mantener La nube funcionando en sus operaciones básicas, es decir, existen componentes de La nube que pueden fallar (como el despliegue de una máquina virtual o asignación de nuevo almacenamiento por mencionar algunos) y no afectan al funcionamiento de La nube, y otros componentes

(como la base de datos de La nube, el *Message Broker*, por mencionar algunos) que son de vital importancia y que si alguno de ellos falla, La nube deja de funcionar.

Por tal motivo todos los componentes necesarios son agrupados en el núcleo para trabajar de forma local. Una de las restricciones del proyecto es que se trabaja de manera compartida, es decir, el recurso de red es utilizada por usuarios ajenos al proyecto y se debe garantizar que estos no puedan acceder a los servidores o sus comunicaciones internas.

Para realizar todo lo descrito en el párrafo anterior se utiliza la tecnología 802.11q, es decir, las VLAN. Esta tecnología permite dividir de manera lógica un *switch* de red, y permite la separación del tráfico de los usuarios ajenos del tráfico de red de los servidores.

3.1.1.2 *Configuración remota*

Para este proyecto se usa el esquema de sitio a sitio (del Inglés *site to site*), por ser el que realiza la conexión de dos redes de forma segura. Esto permite que la comunicación entre dos *host* se realice de forma transparente, es decir, sin realizar modificaciones en los *host*.

En la figura 3.2 la "Red A" se comunica a Internet utilizando el canal de comunicación indicado en color rojo. Primero la red contacta a su puerta de enlace (*gateway*) y este a Internet. Esta comunicación se realiza sin cifrado.

De la misma forma sucede con la "Red B". Hasta este punto existe una forma de comunicar la "Red A" con la "Red B" pero esta comunicación es insegura, para corregir esto se necesita una RPV. Esta red se instala en la puerta de enlace de la red y su trabajo es establecer un túnel seguro a través de internet permitiendo a las redes una comunicación segura, dicha comunicación se encuentra indicada de color azul.

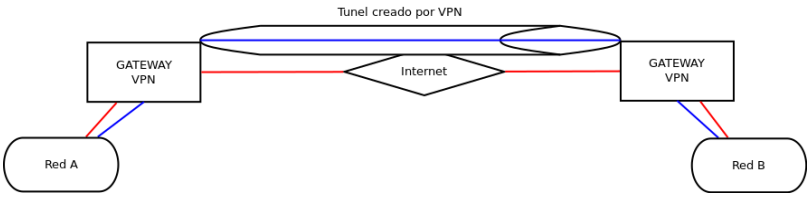

Figura 3.2: Modelo de RPV.

La seguridad en las comunicaciones se logra mediante la instalación de la RPV en la puerta de enlace en cada una de la redes. Se debe de realizar de esta forma ya que cuando un *host* intenta enviar un paquete a una red, primero verifica si se encuentra en alguna de sus interfaces de red, en caso de no encontrar ninguna coincidencia, envía el paquete a su puerta de enlace. Es aquí donde la puerta de enlace inspecciona el paquete y determina si va dirigido a Internet o

a la RPV. En caso de dirigirse a la RPV lo procesa cifrándolo y realizando los ajustes necesarios para que pueda ser enviado por Internet a su destino, un ejemplo de implementación se puede ver en la figura 3.3.

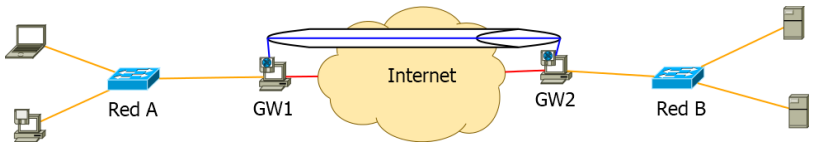

Figura 3.3: Ejemplo de implementación de RPV.

3.1.2 *Automatización en hardware*

MaaS se encarga de realizar la automatización, esto es la etapa de instalación del sistema operativo, su configuración y preparación para que el siguiente nivel pueda iniciar la instalación del software requerido. En la Figura 3.4 se muestran los elementos que participan. El nodo es el hardware objetivo en donde se instalará el sistema operativo, es decir, el equipo que se utilizará para La nube. El nodo es manipulado por el Controlador de rack, este utiliza el arranque PXE (por sus siglas en inglés PXE Pre-Boot Execution Environment) junto con el protocolo de configuración dinámica de *host* (por sus siglas en inglés DHCP Dynamic Host Configuration Protocol) para instalar el sistema operativo al nodo y establecer la configuración.

Dicha configuración es proporcionada por el controlador de región, la cual es suministrada por el administrador utilizando la interfaz WEB o la API. Además el controlador de región es el encargado de llevar el control de los nodos registrados y su estado.

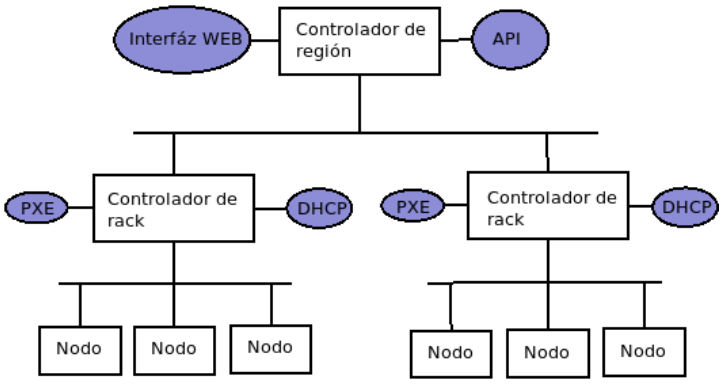

Figura 3.4: Funcionamiento de MaaS.

3.1.3 *Orquestación en hardware*

El principal trabajo se encuentra en el trabajo colaborativo entre MaaS y JuJu, los cuales realizan la automatización y orquestación necesaria para construir La nube sobre la infraestructura existente.

El nivel anterior termina cuando finaliza la configuración del sistema operativo en el nodo, para continuar en el presente nivel en donde JuJu se encarga de obtener, instalar y configurar el software de acuerdo a los requerimientos establecidos en el *bundle*. Un *bundle* esta compuesto por un conjunto de *charms* que han sido combinados para trabajar en conjunto. Y un *charm* contiene todas las instrucciones necesarias para el despliegue y configuración de la aplicación.

En esta etapa se tomó el *bundle* de instalación de *OpenStack* existente del repositorio de JuJu y se le realizaron los ajustes para el modelo propuesto. En la Figura 3.5 se muestran los elementos que participan. El cliente JuJu es donde el administrador debe de configurar JuJu con la API de MaaS, específicamente con el controlador de región. Una vez configurada la API, y antes de desplegar cualquier *charm* es necesario que desde el cliente JuJu se inicie un controlador, este es el encargado de llevar el control de los *charms* desplegados y a la vez, sirve como primera prueba para verificar que el nivel de automatización funciona. En la Figura 3.5 se muestra que el controlador JuJu pertenece a uno de los nodos disponibles en MaaS. Y que los próximos nodos que sean utilizados, son para los *charms* a desplegar.

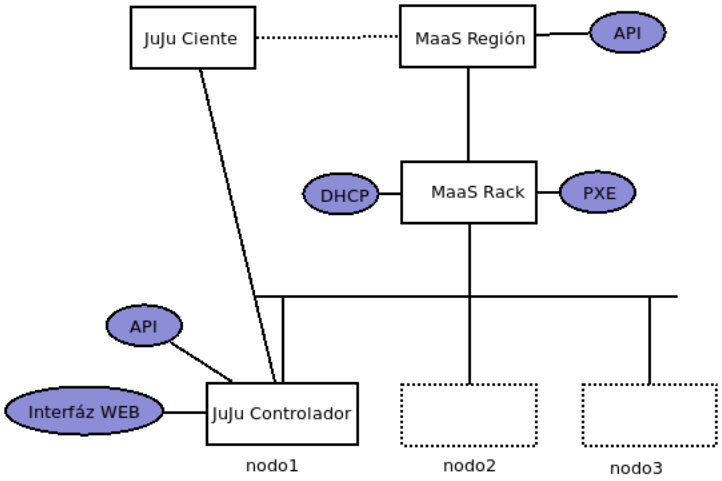

Figura 3.5: Funcionamiento de MaaS y JuJu.

3.1.4 *Nube*

El principal objetivo de este nivel es la configuración propia de La nube. Esta configuración es dependiente del software utilizado, siendo para el presente trabajo *OpenStack*. El mayor trabajo es del admi-

nistrador, quien es el encargado de configurar y planear la capacidad de La nube.

En el presente nivel se limitará a dar una lista de las actividades a realizar:

- Crear cuentas de usuarios.
- Crear proyectos.
- Configurar puerta de enlace hacia internet.
- Configurar los encaminadores internos.
- Configurar los sistemas operativos disponibles para los usuarios.
- Configurar el almacenamiento disponible.

La lista anterior contiene los puntos básicos para que un usuario de La nube, pueda utilizarla, pero hasta el presente punto, solo podrá utilizar automatización en sus proyectos, y para poder utilizar orquestación se debe de avanzar al siguiente nivel.

3.1.5 *Orquestación en Nube*

En el presente nivel se utilizará el mismo concepto del nivel Orquestación en hardware, pero ahora dentro de *OpenStack*, se instalará otro controlador de JuJu el cual administrará los servicios que se ocupen en los proyectos como pueden ser: servidor web, correo, *hadoop*, utilizando los *charms* aportados por la comunidad o bien creando las propios. En la Figura 3.6 se muestra los elementos participantes en el nivel, el cliente JuJu se configura con el API de openstack y de igual forma que sucede con el nivel de Orquestación en hardware se debe de crear el controlador JuJu.

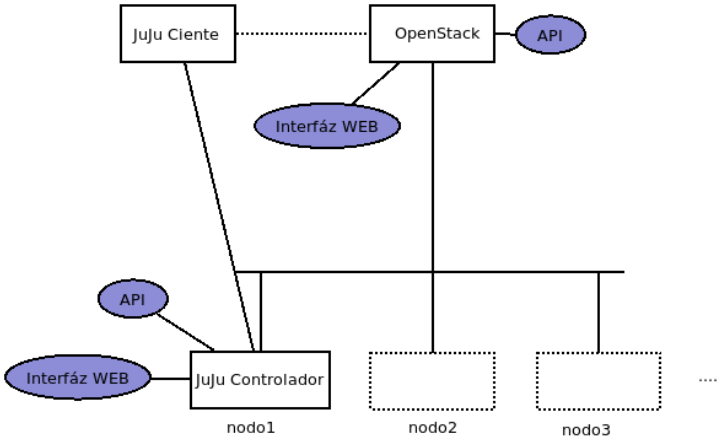

Figura 3.6: Funcionamiento de JuJu y OpenStack.

3.2 ELEMENTOS PARA LA ARQUITECTURA DE ORQUESTACIÓN

El modelo se divide en capas, en donde cada capa es transparente a la capa superior.

3.2.1 *Protocolo TCP/IP*

En el nivel más bajo se encuentra el protocolo TCP/IP.

Por la forma en que está desarrollado el *OpenStack*, dividido en módulos, estos módulos se comunican a través de direcciones IP lo que permite colocar los módulos en diferentes equipos, incluso en equipos separados geográficamente. Estos módulos se comunicarán utilizando Internet, el cual es una red insegura, por lo que se necesita asegurar las comunicaciones.

3.2.2 *RPV*

El siguiente nivel es la RPV.

Esta capa se encargará de transmitir mensajes seguros a través de Internet, lo que permitirá la comunicación de los módulos del *OpenStack* de forma segura. Esta comunicación se realiza de forma transparente para el *OpenStack*.

3.2.3 *MaaS*

En el siguiente nivel se encuentra MaaS.

Esta capa se encargará de realizar la instalación del sistema operativo en el hardware. Sus funciones son: encender el equipo, instalar el sistema operativo y dejarlo listo para la siguiente capa.

3.2.4 *Juju*

En el siguiente nivel se encuentra JuJu.

Esta capa se encargará de instalar los módulos del *OpenStack* en los equipos preparados por la capa anterior. Sus funciones son: descargar, instalar y configurar los módulos de *OpenStack* de acuerdo a la receta.

3.2.5 *OpenStack*

En el siguiente nivel se encuentra *OpenStack*.

Esta capa se encargará de ofrecer los servicios de *IaaS* a los usuarios de las nubes.

3.3 NECESIDADES DE HARDWARE

El primer problema que se enfrenta al implementar el Cómputo en nube en una institución es contar con el requerimiento mínimo de equipo de cómputo de acuerdo a la documentación de cada software utilizado. Cuando se construyó de forma teórica el funcionamiento de La nube se requirió lo siguiente:

Se requieren en total 12 equipos de cómputo distribuidos de la siguiente manera:

- 4 equipos de cómputo para OpenStack.
- 2 equipos de cómputo para MaaS.
- 2 equipos de cómputo para Juju.
- 1 equipo de cómputo para el enrutador y la RPV.
- 3 equipos de cómputo para la extensiones de enrutador/RPV, Maas y nodo de cómputo.

A continuación se especifican las características de cada equipo de cómputo según los niveles de utilización:

PARA OPENSTACK 4 equipos de cómputo con las siguientes características:

- 8 GB en RAM.
- 2 Discos duros: uno para el sistema operativo y el software, y el otro para el almacenamiento de la nube.
- 2 tarjetas de red: una para las comunicaciones internas de la nube, y la otra para la red externa de la nube.
- Núcleos de procesador suficientes para ser utilizados por las máquinas virtuales.

Los módulos del *OpenStack* serán distribuidos en los 4 equipos.

PARA MAAS 2 equipos de cómputo con la siguientes características (uno para el controlador de región y el otro para el de rack):

- 4 GB en RAM.
- 1 Disco duro para el sistema operativo y el software (mínimo 160GB).
- 1 y 2 tarjetas de red (un equipo con una sola tarjeta y el otro con dos).
- 1 Procesador de cuatro núcleos.

PARA JUJU 2 equipos de cómputo con las siguientes características (uno como el controlador y el otro como el cliente):

- 4 GB en RAM.
- 1 Disco duro (mínimo 160GB).
- 1 Tarjeta de red.
- 1 Procesador de cuatro núcleos.

PARA EL ENRUTADOR Y RPV 1 equipo de cómputo con las siguientes características (para la comunicación sitio a sitio):

- 4 GB en RAM.
- 1 Disco duro (mínimo 80GB).
- 2 Tarjetas de red.
- 1 Procesador de cuatro núcleos.

PARA CADA EXTENSIÓN SE OCUPA

- Para el Router y RPV 1 equipo con las siguientes características (para la comunicación sitio a sitio).
  - 4 GB en RAM.
  - 1 Disco duro (mínimo 80GB).
  - 2 tarjetas de red.
  - 1 Procesador de doble núcleo.
- Para MaaS, 1 equipo de cómputo con las siguientes características (para el controlador de rack).
  - 4 GB en RAM.
  - 1 Disco duro (mínimo 160GB).
  - 2 tarjetas de red: una para la RPV y la otra para los nodos de cómputo.
  - 1 Procesador de cuatro núcleos.
- Para el nodo de cómputo, 1 equipo de cómputo con las siguientes características:
  - 8 GB en RAM.
  - 1 Disco duro (mínimo 320GB).
  - 1 tarjeta de red.
  - Núcleos de procesador suficientes para ser utilizados por las máquinas virtuales.

Todos los equipos anteriores deberán estar cableados y con conexión a Internet en una red ethernet a 1 Gbps y con un *switch* dedicado. Si se está compartiendo, configurar una VLAN diferente a la del tráfico normal de la red.

Los escenarios en los que se puede realizar la implementación son los siguientes:

- Implementación física (Se cuenta con cada uno de los equipos de cómputo físico).
- Implementación virtual (Se cuenta con equipo de cómputo físico que reuna la suma de los requerimientos de todos los equipos separados).
- Implementación mixta (Se cuenta con equipo de cómputo físico que puede agrupar algunos requerimientos y con equipos separados para los faltantes).

Como un detalle extra, se puede reemplazar los equipos para el RPV con hardware de red, pero deberá soportar IPSec y RIP. El presente trabajo se realizó utilizando una implementación mixta.

3.4 CONCLUSIÓN

En el capítulo se mostró la propuesta del modelo de orquestación de servicios en la nube, en donde se describe su funcionamiento de forma general, para posteriormente continuar por describir cada uno de los niveles que lo conforman. Se aborda de forma teórica los requerimientos para ser implementada la propuesta y los escenarios en donde puede ser implementada, siendo la implementación mixta la utilizada en el presente trabajo por el equipo de cómputo disponible.

4

IMPLEMENTACIÓN Y RESULTADOS

En este capítulo se muestra la implementación mixta del modelo de orquestación para la administración de recursos en redes privadas y públicas.

La implementación del modelo de orquestación propuesto se muestra en la Figura 4.1. En ella se ejemplifican los tres escenarios descritos en el capítulo 2, a saber: VLAN en el mismo edificio, RPV dentro de la institución y RPV hacia fuera de la institución. En dichos escenarios se muestran las conexiones de red y los dispositivos físicos utilizados.

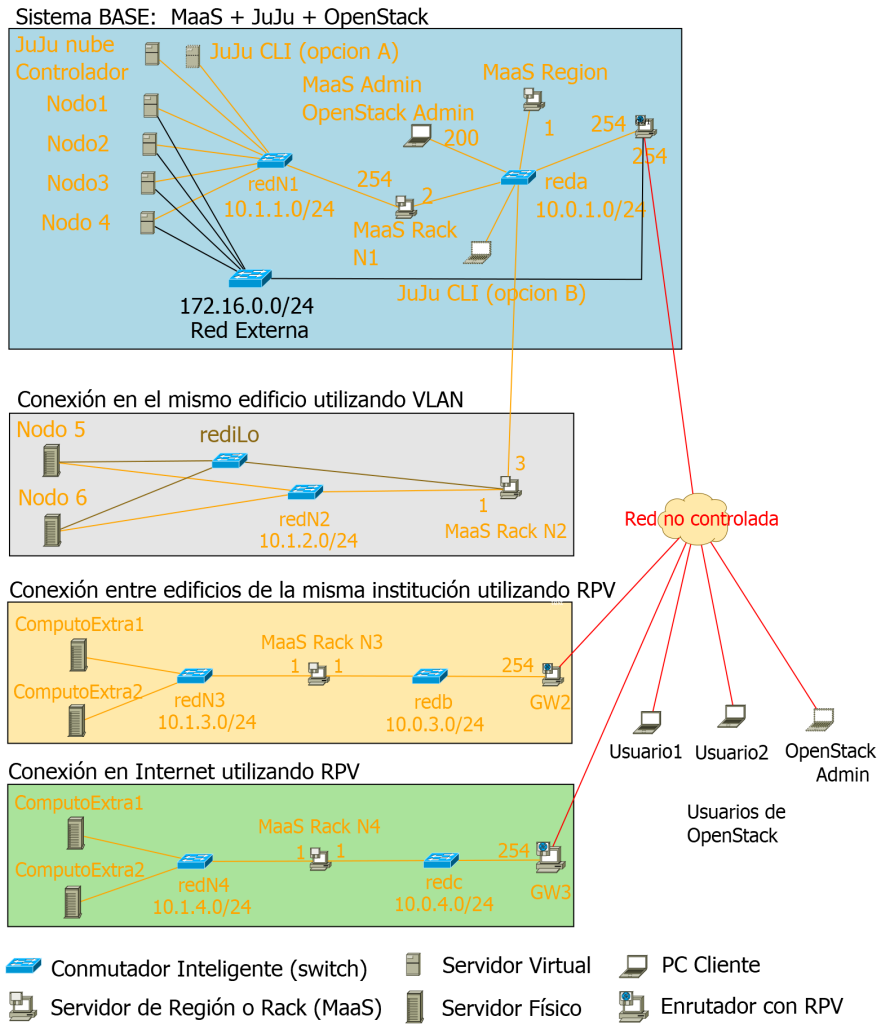

Figura 4.1: Diagrama general del modelo de orquestación implementado.

De igual forma se puede observar recuadros en colores celeste, gris, amarillo y verde los cuales agrupan a dispositivos que cumplen un mismo objetivo que se describirá a continuación.

RECUADRO: SISTEMA BASE CONSTITUIDO POR MAAS + JUJU + OPENSTACK En la Figura 4.2 se encuentran los componentes esenciales para el correcto funcionamiento de la arquitectura propuesta. En la Figura 4.3 se muestra las conexiones entre los diversos componentes. Es esencial mencionar que los componentes dentro de este grupo deben funcionar correctamente. De lo contrario, una falla en algunos de ellos haría que el funcionamiento en general se vea afectado ó incluso averiado. En la Tabla 4.1 se muestran los sistemas utilizados.

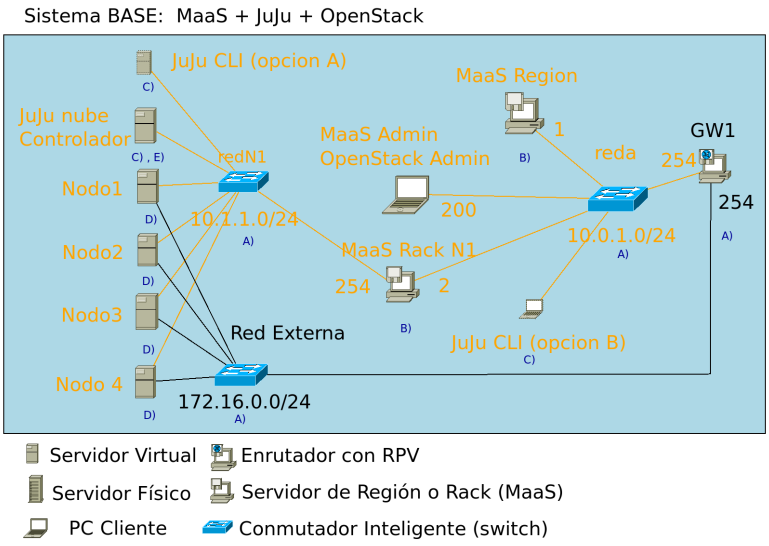

Figura 4.2: Diagrama del sistema base.

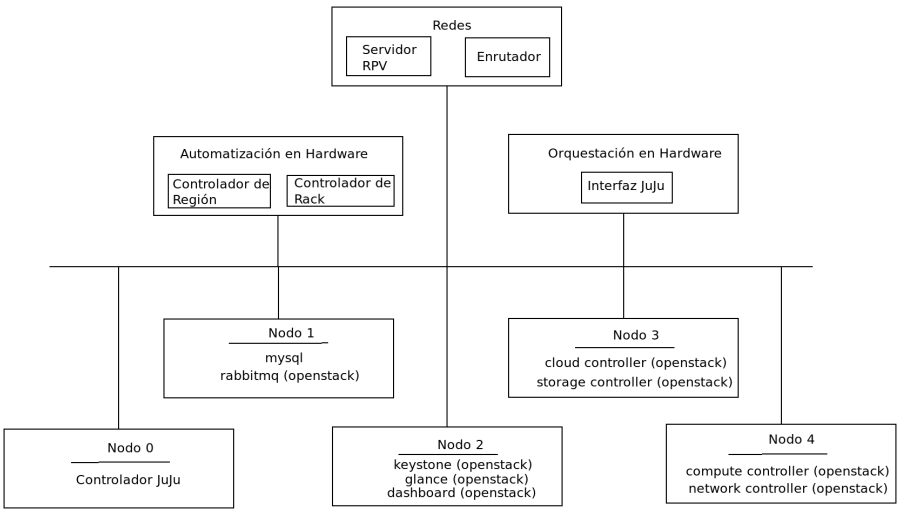

Figura 4.3: Diagrama lógico del sistema base.

Bloque del modelo	Equipos del modelo
A) Redes	Puerta de enlace y Servidor RPV
B) Automatización en hardware	Controlador de Región y de Rack
C) Orquestación en hardware	Interfaz de JuJu y Controlador JuJu
D) Nube	Módulos base de OpenStack
E) Orquestación en nube	Controlador JuJu

Tabla 4.1: Elementos del Sistema Base.

RECUADRO: CONEXIÓN EN EL MISMO EDIFICIO UTILIZANDO VLAN

En la Figura 4.4 se agrupan los componentes que serán utilizados cuando se desee escalar La nube, en la Figura 4.5 se muestra su diagrama lógico. Para el escenario en donde el sistema base se encuentre en un cuarto de telecomunicaciones sin capacidad para agregar nuevo equipo de cómputo pero en ese mismo edificio existe otro cuarto de telecomunicaciones en el cual puede ser instalado al nuevo equipo. Al contrario que el sistema base, una falla en esta agrupación no afecta al funcionamiento general de la propuesta. En la Tabla 4.2 se muestran los sistemas utilizados.

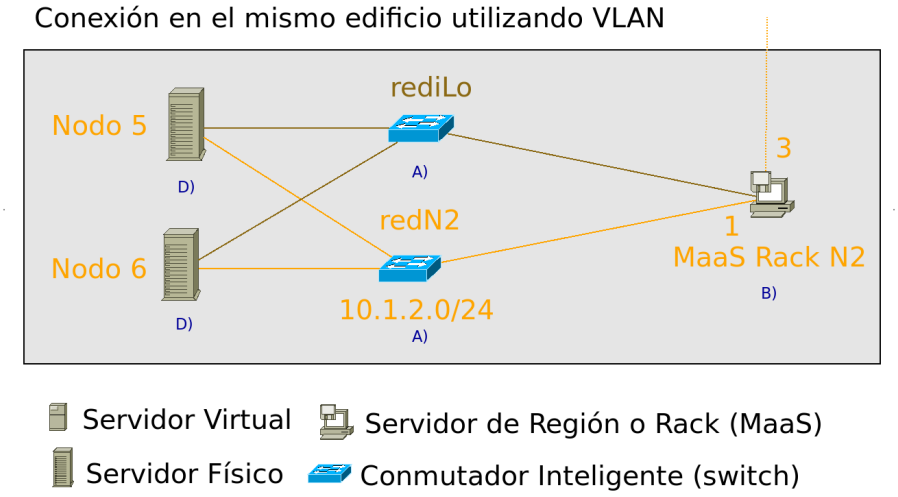

Figura 4.4: Diagrama de conexión en el mismo edificio utilizando VLAN.

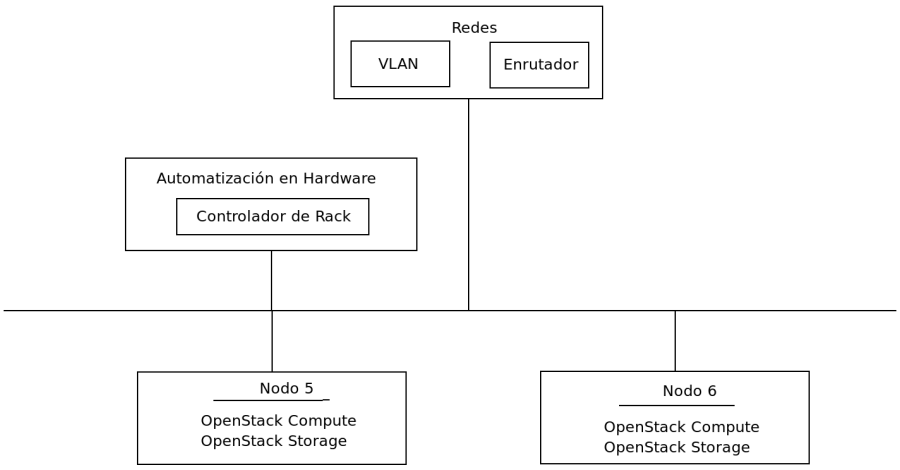

Figura 4.5: Diagrama lógico de conexión en el mismo edificio utilizando VLAN.

Bloque del modelo	Equipos del modelo
A) Redes	Configuración VLAN
B) Automatización en hardware	Controlador de Rack
C) Orquestación en hardware	-
D) Nube	Módulo Compute ó Storage de OpenStack
E) Orquestación en nube	-

Tabla 4.2: Elementos de conexión en el mismo edificio utilizando VLAN.

RECUADRO: CONEXIÓN ENTRE EDIFICIOS DE LA MISMA INSTITUCIÓN UTILIZANDO RPV. En la Figura 4.6 agrupan los componentes que serán utilizados cuando se desee escalar la nube, en la Figura 4.7 se muestra su diagrama lógico. Para el escenario en donde el sistema base se encuentre en un cuarto de telecomunicaciones sin capacidad para agregar nuevo equipo de cómputo pero en otro edificio de la misma institución existe un cuarto de telecomunicaciones en el cual pueda ser instalado nuevo equipo. Al contrario que el sistema base, una falla en esta agrupación no afecta al funcionamiento general de la propuesta. En la Tabla 4.3 se muestran los sistemas utilizados.

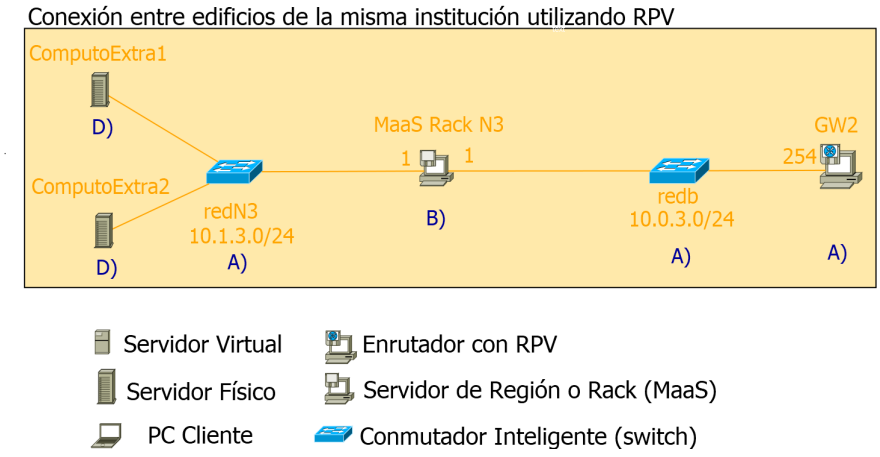

Figura 4.6: Diagrama de conexión entre edificios de la misma institución utilizando RPV.

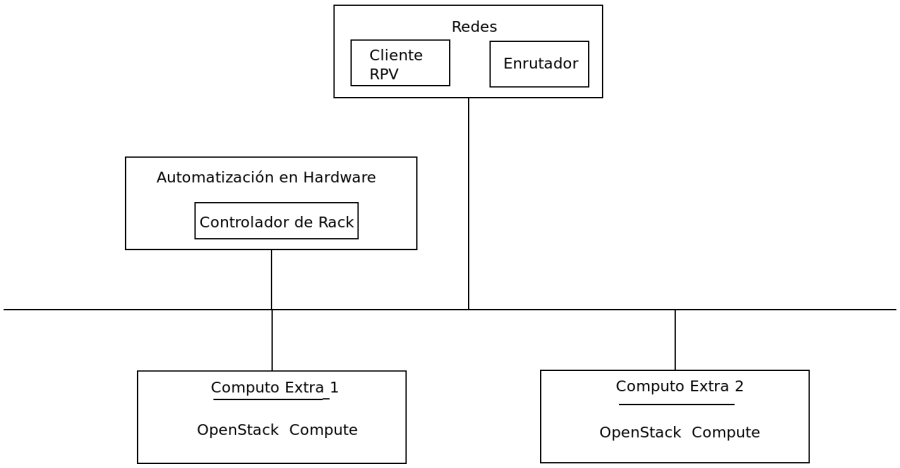

Figura 4.7: Diagrama lógico de conexión entre edificios de la misma institución utilizando RPV.

Bloque del modelo	Equipos del modelo
A) Redes	Servidor RPV
B) Automatización en hardware	Controlador de Rack
C) Orquestación en hardware	-
D) Nube	Modulo Compute de OpenStack
E) Orquestación en nube	-

Tabla 4.3: Elementos de conexión entre edificios de la misma institución utilizando RPV.

RECUADRO: CONEXIÓN A TRAVÉS DE INTERNET UTILIZANDO RPV

En la Figura 4.8 se agrupan los componentes que serán utilizados cuando se desee escalar la nube, en la Figura 4.7 se muestra su diagrama lógico. Para el escenario en donde el sistema base se encuentre

en un cuarto de telecomunicaciones sin capacidad para agregar nuevo equipo de cómputo pero en otro lugar con acceso a Internet existe un cuarto de telecomunicaciones en el cual pueda ser usado nuevo equipo. Al contrario que el sistema base, una falla en esta agrupación no afecta al funcionamiento general de la propuesta. En la Tabla 4.4 se muestran los sistemas utilizados.

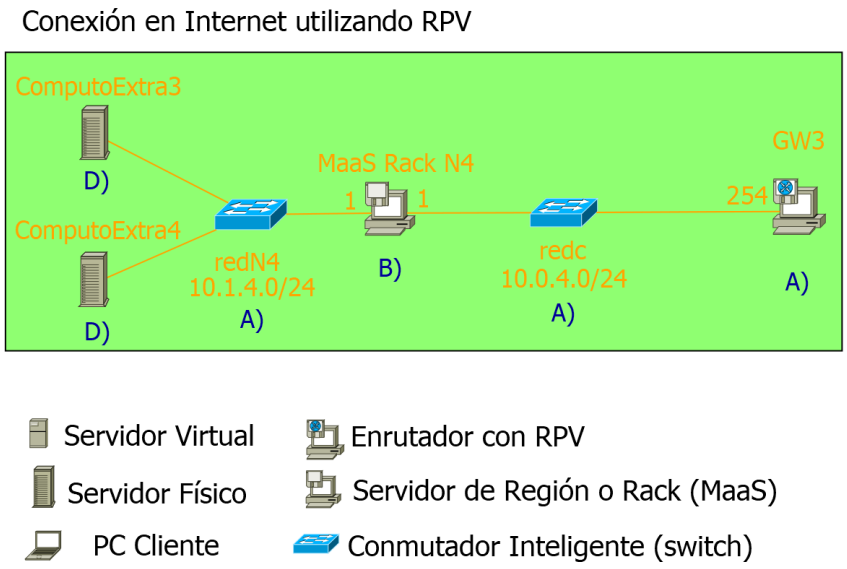

Figura 4.8: Diagrama de conexión en Internet utilizando RPV.

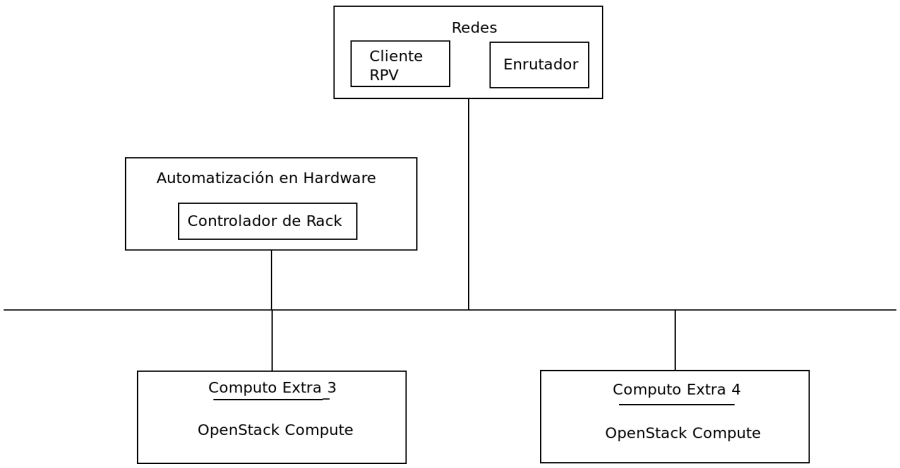

Figura 4.9: Diagrama lógico de conexión en Internet utilizando RPV.

Bloque del modelo	Equipos del modelo
A) Redes	Servidor RPV
B) Automatización en hardware	Controlador de Rack
C) Orquestación en hardware	-
D) Nube	Modulo Compute de OpenStack
E) Orquestación en nube	-

Tabla 4.4: Elementos de conexión en Internet utilizando RPV.

A continuación se describe el orden de instalación de los componentes de la arquitectura.

4.1 CONFIGURACIÓN DE LA RED

El primer paso para la implementación del modelo de orquestación propuesto es instalar y configurar la red. Para ello se deben configurar adecuadamente los equipos de cómputo o dispositivos de hardware que permitirán la comunicación entre toda la arquitectura propuesta. Un ejemplo de implementación del equipo que realizará la función de puerta de enlace se muestra en Listado 4.1 y de su configuración en Listado 4.2. Después se debe de configurar el servidor de RPV.

Listado 4.1: Instalación de quagga

```
Instalación de los paquetes con:

  sudo apt install quagga quagga-doc

Habilitar el reenvío IPv4

  echo "net.ipv4.conf.all.forwarding=1" | sudo tee -a /etc/sysctl
      .conf
  echo "net.ipv4.conf.default.forwarding=1" | sudo tee -a /etc/
      sysctl.conf
  sudo sysctl -p

Se crean los archivos de configuración:

  sudo touch /etc/quagga/zebra.conf
  sudo touch /etc/quagga/ripd.conf
  sudo touch /etc/quagga/vtysh.conf

Se establecen los permisos:

  sudo chown quagga:quagga /etc/quagga/zebra.conf && sudo chmod
      640 /etc/quagga/zebra.conf
  sudo chown quagga:quagga /etc/quagga/ripd.conf && sudo chmod
      640 /etc/quagga/ripd.conf
  sudo chown quagga:quaggavty /etc/quagga/vtysh.conf && sudo
      chmod 660 /etc/quagga/vtysh.conf

Se configuran los servicios a utilizar:

  sudoedit /etc/quagga/daemons

  ...
  zebra=yes
  ...
  ripd=yes
  ...
{}
```

Listado 4.2: Configuración de quagga

```
Se inicia el servicio con:

  sudo service quagga restart

Se ingresa al programa con:

  sudo VTYSH\_PAGER=more vtysh

Se inicia la configuración con:

  configure terminal
  hostname GW1
  password tecnologico
  enable password 1tcv3d
  ip forwarding
  router rip
  version 2
  network ens6
  redistribute connected
  end
  write
  exit

Se habilita el NAT con :

  sudo iptables -t nat -A POSTROUTING -o ens2 -j MASQUERADE
  sudo iptables -A FORWARD -i ens2 -o ens6 -m state --state
      RELATED,ESTABLISHED -j ACCEPT
  sudo iptables -A FORWARD -i ens6 -o ens2 -j ACCEPT

donde :

ens2 = eth0
ens6 = eth1
```

Una vez finalizado la configuración de la puerta de enlace, el servidor de RPV y los dispositivos de red de procede a configurar la automatización en hardware.

4.2 CONFIGURACIÓN DE LA AUTOMATIZACIÓN EN *hardware*

En primer lugar debe de instalarse el controlador de región como se indica en Listado 4.3. Después se instala el controlador de *rack* siguiendo Listado 4.4. El controlador de *rack* desempeña un papel importante: es la puerta de enlace para los nodos que dependan de él. En Listado 4.5 se muestra como realizar la configuración.

Listado 4.3: Configuración de controlador de región

```
Se realiza una instalación de Ubuntu server con idioma por
    defecto en Inglés y teclado latinoamericano, la red se
    configura de forma manual de acuerdo a la implementación.

Una vez instalado el sistema operativo se ejecuta en una terminal
    :

  sudo apt install maas-region-controller

Después se crea el usuario admin en donde se solicitará la
    contraseña del usuario y después se preguntará por la
    contraseña para la cuenta a crear:

  sudo maas createadmin --username=admin --email=gamamtz@region1

  [sudo] password for gamamtz:
  Password: tecnologico
  Again:

Seguido se crema el directorio .maas mediante :

  mkdir .maas

Enseguida se debe crear una variable de entorno con los
    siguientes datos:

  API\_KEY\_FILE=/home/gamamtz/.maas/apikey.admin

Con el siguiente comando se generá un api-key para acceder por
    consola

  sudo maas-region apikey --username=admin > \$API\_KEY\_FILE

Se puede iniciar sesión con:

  maas login admin http://localhost:5240/MAAS/api/2.0 - < \$API\_
      KEY\_FILE

y se mostrará lo siguiente mensaje, si se realizó correctamente:

  You are now logged in to the MAAS server at
  http://localhost:5240/MAAS/api/2.0/ with the profile name '
      admin'.

  For help with the available commands, try:

  maas admin --help

Se termina la sesión con:

  maas logout \$PROFILE
```

Listado 4.4: Configuración de controlador de rack

```
Se realiza la instalación de Ubuntu server en idioma por defecto
    en Inglés y teclado latinoamericano, la red se configura de
    forma manual de acuerdo a la implementación.

Una vez instalado el sistema operativo se ejecuta en una terminal
    :

  sudo apt install maas-rack-controller

Se realiza una conexión mediante el cliente ssh al servidor de
    región y se busca el archivo /var/lib/maas/secret el cual
    contiene la clave para el registro con el controlador de
    región.

%Nota: la clave también puede ser encontrada en el portal del
    MaaS en la sección de ajustes

la clave para este servidor es 84f82cd886e01c87b649c16a6680e79d

Conociendo la clave en el controlador de rack y se ejecuta:

  sudo maas-rack register \\
  --url http://10.0.1.1:5240/MAAS \\
  --secret  84f82cd886e01c87b649c16a6680e79d

el comando anterior no muestra ningún resultado, la forma de
    comprobar si la conexión se realizó es en el portal de MaaS,
    en la sección: lista de controladores.
```

Listado 4.5: Configuración de puerta de enlace en el controlador de rack

```
Se debe de instalar los siguientes paquetes:

  sudo apt install quagga quagga-doc

Se habilita el reenvío IPv4

  echo "net.ipv4.conf.all.forwarding=1" | \
  sudo tee -a /etc/sysctl.conf

  echo "net.ipv4.conf.default.forwarding=1" | \
  sudo tee -a /etc/sysctl.conf

  sudo sysctl -p

Se crean los archivos de configuración:

  sudo touch /etc/quagga/zebra.conf  && \

  sudo touch /etc/quagga/ripd.conf && \

  sudo touch /etc/quagga/vtysh.conf

Y se configuran los permisos:

  sudo chown quagga:quagga /etc/quagga/zebra.conf && \
  sudo chmod 640 /etc/quagga/zebra.conf

  sudo chown quagga:quagga /etc/quagga/ripd.conf && \
  sudo chmod 640 /etc/quagga/ripd.conf

  sudo chown quagga:quaggavty /etc/quagga/vtysh.conf && \
  sudo chmod 660 /etc/quagga/vtysh.conf

Se activan los servicios a ejecutar:

  sudoedit /etc/quagga/daemons

  ...
  zebra=yes
  ...
  ripd=yes
  ...

Se inicia el servicio con:

  sudo service quagga restart

Se ingresa mediante:

  sudo VTYSH\_PAGER=more vtysh

Se configura como se muestra a continuación:

  configure terminal
  hostname RackLAB
  password tecnologico
  enable password 1tcv3d
  ip forwarding
  router rip
```

Terminada la configuración de la automatización en *hardware*, se procede a registrar los nodos en donde se realizará la instalación de La nube. En la Figura 4.10 se muestra un ejemplo de los nodos registrados.

Después del registro de los nodos en La nube se inicia la configuración de la orquestación del hardware como se describe enseguida.

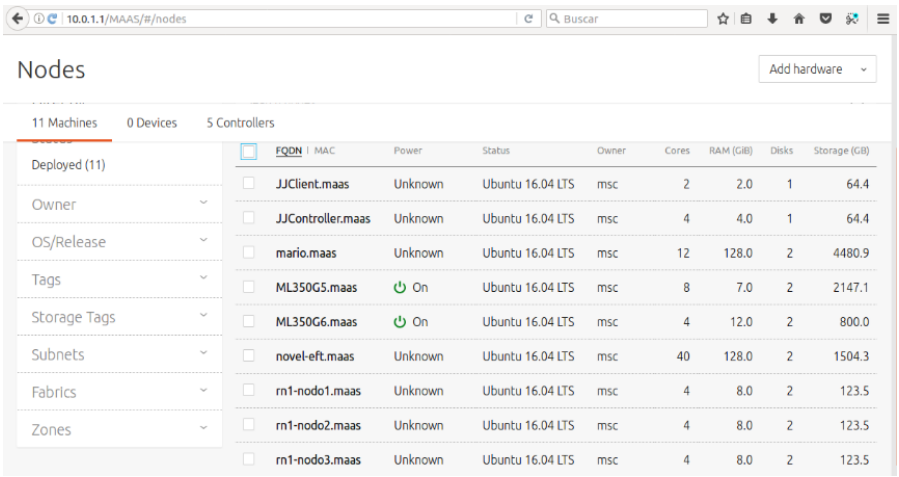

Figura 4.10: Nodos registrados en MaaS.

4.3 ORQUESTACIÓN EN *hardware*

Completada la configuración de Red y la configuración de automatización en *hardware* se debe de configurar la orquestación. Para ello se necesita del siguiente equipo:

1 Equipo que cuente con 2 núcleos, 2 Gb RAM, 60 GB disco duro para la interfaz de comandos de JuJu

1 Equipo que cuente con 4 núcleos, 4 Gb RAM, 60 GB disco duro para la creación del controlador de JuJu

El nodo que se utilizará para la creación del controlador de JuJu debe ser registrado dentro de MaaS, siendo una opción para el nodo de interfaz de comandos de JuJu el ser registrado ó utilizar una instalación tradicional del sistema operativo. Una vez realizado lo anterior se inicia por configurar la interfaz de comandos de JuJu como se describe enseguida.

Crear una cuenta de usuario que controlará la instalación de la nube. Para el presente ejemplo se utilizará el usuario MSC. Después de crear al usuario se instalará el software JuJu y se configura dando de alta la instalación del sistema de automatización de *hardware* como proveedor para el JuJu. Un ejemplo de la implementación de puede ver en Listado 4.6.

Listado 4.6: Configuración de orquestanción en hardware

```
Se instala la versión de JuJu 2.2.2 que se encuentra en snap
    store ejecutando:

  sudo snap install juju --classic

Se crea el archivo maas-clouds.yaml el cual contiene los
    siguientes datos:

  clouds:
    maas:
    type: maas
    auth-types: [oauth1]
    endpoint: http://10.0.1.1/MAAS

para dar de alta la nube se crea con:

  juju add-cloud maas maas-clouds.yaml

se agrega la credencial:

  juju add-credential maas

  Enter credential name: msc

  Using auth-type "oauth1".

  Enter maas-oauth:   (aquí pegar la api key del usuario generada
      del perfil en MaaS. No se ve lo que se escribe, así que
      pegar solo 1 vez. En caso de error, se puede repetir el
      comando para corregir)*Credentials added for cloud
      maas.Ahora creamos el controlador con:juju bootstrap maas
      msc-controllerCreating Juju controller "msc-controller.on
      maasLooking for packaged Juju agent version 2.2.2 for
      amd64Launching controller instance(s) on maas...- tsqw7f
      (arch=amd64 mem=4G cores=4)Fetching Juju GUI 2.8.0Waiting
      for addressAttempting to connect to 10.1.1.156:22Bootstrap
      agent now startedContacting Juju controller at 10.1.1.156
      to verify accessibility...Bootstrap complete,
      "msc-controllerçontroller now available.Controller machines
      are in the çontroller"model.Initial model "default.ªdded.Con
      el comando siguiente podemos ver el estado de la nube:juju
      statusCon esto se ha terminado de instalar JuJu.
```

El paso siguiente es instalar OpenStack mediante el *Charm* adecuado para JuJu, en la Figura 4.11 se muestra los módulos a instalar y las conexiones entre ellos. Dichos módulos serán instalados en los 4 nodos registrados en el MaaS y actuarán como el sistema base. En la Figura 4.12 se muestra el *Charm* de La nube implementada.

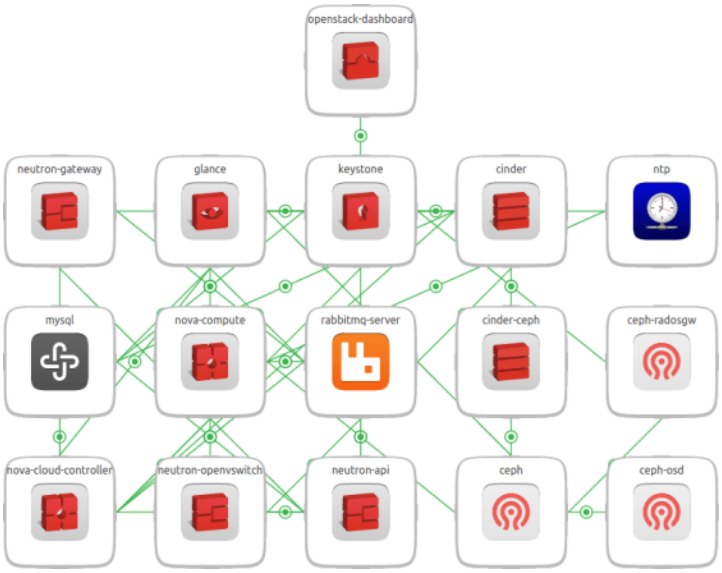

Figura 4.11: *Charm* que muestra las conexiones entre los diferentes módulos de OpenStack.

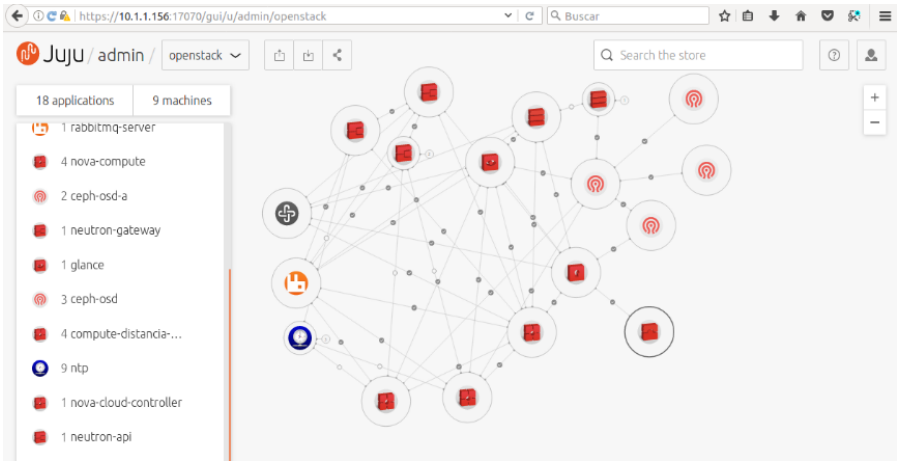

Figura 4.12: *Charm* de JuJu implementada.

Para desplegar el *Charm* es necesario colocarse en el nodo para la interfaz de comandos de JuJu y seguir las Listado 4.7.

Listado 4.7: Desplegar el *Charm* de JuJu para instalar OpenStack

```
En un terminal ejecutar:

  juju deploy openstack-base-bundle.yaml
```

4.4 NUBE

En esta sección el administrador de La nube, deberá de configurar los enrutadores que podrán ser utilizados por las máquinas virtuales, los diferentes tamaños para las máquinas virtuales, los proyectos y los usuarios. Para realizar dichas configuraciones se debe de consultar el manual oficial de OpenStack debido a que el funcionamiento de La nube no ha sido modificado y por lo tanto siendo transparente su funcionamiento. En la Figura 4.13 se muestra la capacidad total de la nube (en la parte superior), en la parte central los nodos de cómputo activos, así como la cantidad de procesadores virtuales utilizados, memoria *RAM* y disco duro.

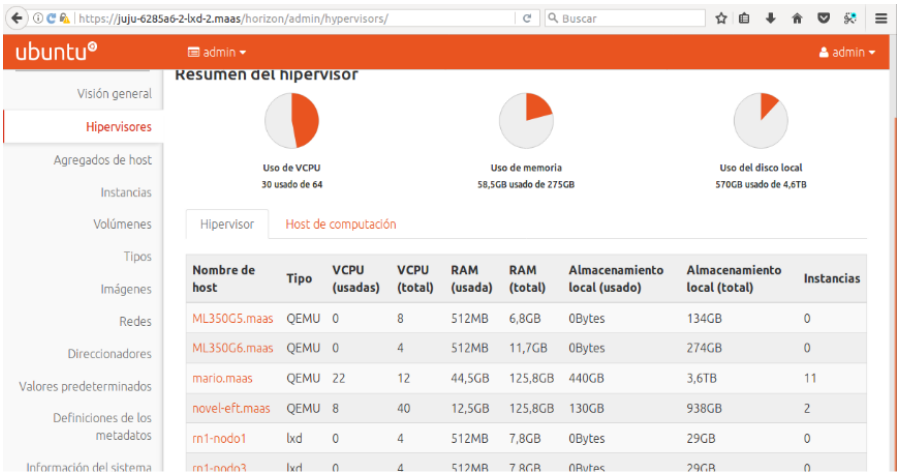

Figura 4.13: Resumen de los nodos activos en OpenStack.

4.5 ORQUESTACIÓN EN NUBE

El usuario de la nube será el responsable de implementar el orquestador para La nube, debido a que cada proyecto genera una sola API-key, la cual es utilizada para realizar la conexión entre OpenStack y JuJu.

Listado 4.8: Configuración de orquestación en nube

```
Se crea el archivo openstack-clouds.yaml el cual contiene los
    siguientes datos:

  clouds:
    mystack:
    type: openstack
    auth-types: APIkey
    endpoint: https://:35574/v3.0/

para dar de alta la nube se crea con:

  juju add-cloud mystack openstack-clouds.yaml

se agrega la credencial:

Select one or more auth types separated by commas: userpass

Ahora creamos el controlador con:

  juju bootstrap mystack oscontroller \
  --config network=<network_id> \
  --config external-network=<external_network_id> \
  --config use-floating-ip=true




Con el comando siguiente podemos ver el estado de la nube:

  juju status

Con esto se ha terminado de instalar JuJu.
```

4.6 CONCLUSIÓN

Existen elementos esenciales en cada uno de los niveles del modelo propuesto, los cuales si llegasen a fallar comprometerían el funcionamiento de La nube. Por tal motivo el modelo se divide en una base y la parte extensible. En la base se encuentran todos los elementos que permiten el correcto funcionamiento de La nube, debido a la arquitectura fragmentada del software utilizado para crear La nube es posible que la mayoría de ellos puedan ser utilizados en un sistema de alta disponibilidad como medida para evitar que la base de La nube fallase.

En la parte extensible se encuentran los elementos que no son requeridos para el correcto funcionamiento de La nube y los nodos de cómputo que aumentan la capacidad de la nube. Una falla en esta

parte no afecta al funcionamiento general de La nube, si no más bien a los recursos que se encuentren en la parte extensible con falla.

5 CONCLUSIONES Y PERSPECTIVAS

5.1 CONCLUSIONES

En la actualidad el equipo informático que se encuentra distribuido en el Tecnológico Nacional de México/Instituto Tecnológico de Ciudad Victoria, es utilizado la mayor parte del tiempo en un horario específico, quedándose inactivo la otra parte de este. En un entorno de Cómputo en nube se podría aprovechar el potencial del equipo en cualquier horario. Esto se logra mediante el desacoplo que existe entre el servicio (por ejemplo servidor web, base de datos, etc) que se ofrece y el equipo informático en el que opera. Lo que permite que el servicio pueda ser trasladado entre los diferentes servidores disponibles.

A la vez, se cambia el paradigma del usuario en el que normalmente este piensa que cuando posee un servidor, por ejemplo de 24 núcleos, 32 GB en RAM y 1 TB en disco duro, no pueda realizar una división de los recursos para utilizarlos de una mejor manera. En el entorno de Cómputo en nube se puede dividir los recursos para utilizarlos de la forma que más le convenga, sin requerir la intervención del administrador.

Por otro lado, el Cómputo en nube permite utilizar el equipo informático distribuido y administrarlo de una manera centralizada, sin importar que éste se encuentre en diferentes edificios. Además, la utilización de *hardware* heterogéneo permite la portabilidad a cambio de no poder utilizar las características únicas que pueda ofrecer un determinado *hardware*.

La aportación del presente trabajo es una alternativa al modelo de implementación híbrida, el cual necesita que en los sitos remotos exista toda una infraestructura en La nube disponible, es decir, necesita de dos infraestructura en Nube para funcionar, lo que es un problema para la situación a la que se enfrenta, ya que se cuenta con limitada cantidad de equipo de cómputo. La solución propuesta permite utilizar todo aquel equipo remoto para ser utilizado dentro de la nube.

Como logros se pueden mencionar los siguientes:

- Utilizar el equipo de cómputo disperso en el instituto de manera centralizada.
- Utilizar comunicaciones seguras para separar el tráfico de los usuarios del de los servidores.
- Encender y apagar el equipo de cómputo de forma remota de manera automatizada.

5.2 TRABAJO FUTURO

Al utilizar la RPV para conectar sitios remotos, se pierde la capacidad de utilizar VLAN, esto es debido a que la RPV no envía tráfico a nivel de VLAN. Este error no es reconocido por el MaaS pues no reconoce que existe una RPV. Aunque en la interfaz del MaaS aparece la opción para reconocer a la RPV, en la práctica no funciona. Las Redes Privadas Virtuales Extendidas (VXLAN) son una reciente alternativa para superar las limitaciones de la RPV.

Además, la protección de puntos vitales de La nube debido a un fallo en un punto crítico causaría que La nube deje de operar. Una forma de solucionar esto es implementando lo necesario para la alta disponibilidad.

Adicionalmente, se requiere brindar seguridad a los datos almacenados en La nube. Los mecanismos de seguridad existentes pueden ser violentados. Por lo tanto, es conveniente contar con otra capa de seguridad donde el cifrado sea una opción a ser implementada y analizar su impacto.

BIBLIOGRAFÍA

[1] H. Aalbers, *Introducción a Cloud Computing*. Aalbers, Huibert, dic. de 2013, págs. 32-36. dirección: http://www.huibert-aalbers.com/downloads/IntroduccionCloudComputing.pdf.

[2] H. Albaroodi, S. Manickman y P. Singh, «Critical Review of OpenStack Security Issues and Weaknesses.», *Journal of Computer Science*, 2014, ISSN: 1549-3636. DOI: 10.3844/jcssp.2014.23.33.

[3] A. Alonso, «Cloud Computing Enhancements and Private Cloud Management.», Tesis Licenciatura, Universitad Politecnica de Catalunya, mayo de 2018, págs. 33-43.

[4] R. Anandhi y K. Chitra, «A Challenge in Improving the Consistency of Transactions in Cloud Databases - Scalability», *Computer Applications*, vol. 52, n.º 2, ago. de 2012, ISSN: 0975-8887.

[5] A. Awasthi y R. Gupta, «Comparison of OpenStack Installers», *International Journal of Innovative Science, Engineering & Technology*, vol. 2, n.º 9, sep. de 2015, ISSN: 2348-7968. dirección: ijiset.com/vol2/v2s9/IJISET_V2_I9_92.pdf.

[6] K. Ayanlowo, O. Shoewu, S. O. Olatinwo, T. S. Fadiji y S. Adeyanju, «Conceptual Design and Implementation of a Cloud Computing Platform Paradigm», *Computer Engineering and Intelligent Systems*, vol. 3, n.º 12, 2012, ISSN: 2222-1719. dirección: https://iiste.org/Journals/index.php/CEIS/article/view/3838.

[7] H. Barkallah, M. Gzara y H. Ben Abdallah, «Evolution of the Distributed Computing Paradigms: a Brief Road Map», *International Journal of Computing and Digital Systems*, vol. 6, n.º 5, sep. de 2017, ISSN: 2210-142X. dirección: https://journal.uob.edu.bh/handle/123456789/292.

[8] L. Beernaert, M. Matos, R. Vilaça y R. Oliveira, «Automatic Elasticity in OpenStack», en *Proceedings of the Workshop on Secure and Dependable Middleware for Cloud Monitoring and Management*, ép. SDMCMM '12, Montreal, Quebec, Canada: ACM, 2012, 2:1-2:6, ISBN: 978-1-4503-1615-6. DOI: 10.1145/2405186.2405188. dirección: http://www.gsd.inesc-id.pt/~mm/papers/2012/sdmcmm-elastack.pdf.

[9] G. Brataas, N. Herbst, S. Ivansek y J. Polutnik, «Scalability Analysis of Cloud Software Services», en *2017 IEEE International Conference on Autonomic Computing, ICAC 2017, Columbus, OH, USA, July 17-21, 2017*, 2017, págs. 285-292. DOI: 10.1109/ICAC.2017.34. dirección: https://doi.org/10.1109/ICAC.2017.34.

[10] A. B. Brown y J. L. Hellerstein, «Reducing the Cost of IT Operations: Is Automation Always the Answer?», en *Proceedings of the 10th Conference on Hot Topics in Operating Systems - Volume 10*, ép. HOTOS'05, Santa Fe, NM: USENIX Association, 2005, págs. 12-12. dirección: http://dl.acm.org/citation.cfm?id=1251123.1251135.

[11] C. CANO GENOVÉS, «Automatización de la Reconfiguración Dinámica de Servicios Cloud», Tesis Licenciatura, Universidad Politécnica de Valéncia, 2016, págs. 9-15. dirección: https://riunet.upv.es/bitstream/handle/10251/69080/CANO%20-%20Automatizaci%c3%b3n%20de%20la%20Reconfiguraci%c3%b3n%20Din%c3%a1mica%20de%20Servicios%20Cloud.pdf?sequence=1&isAllowed=y.

[12] L. Coyne, T. Hajas, M. Hallback, M. Lindström y C. Vollmar, *IBM Private, Public, and Hybrid Cloud Storage Solutions*, IBM, abr. de 2016, págs. 1-19, ISBN: 0738456845. dirección: http://www.redbooks.ibm.com/redpapers/pdfs/redp4873.pdf.

[13] J. P. Degabriele y D. Pym, *Economic Aspects of a Utility Computing Service*, 2007.

[14] R. Derrick y C. Ileana, *The Basics of Cloud Computing: Understanding the Fundamentals of Cloud Computing in Theory and Practice*. Syngres is an imprint of Elsevier, 2014.

[15] P. Dillenbourg y P. Jermann, *Technology for classroom orchestration*, 2009.

[16] I. Faynberg, H.-L. Lu y D. Skuler, *Cloud Computing: Business Trends and Technologies*. Wiley, 2016.

[17] M. Feilner, *OpenVPN Building and Integrating Virtual Private Networks*, D. Chittar, ed. Packt Publiching, 2006, ISBN: 1-904811-85-X.

[18] J. Fernández, J. Alonso, C. Figuerola y A. Zazo, *Redes Privadas Virtuales*, 2006.

[19] A. González, *Redes Privadas Virtuales*, 2006.

[20] G. E. Guerrero Mejias, «Diseño y Análisis de soluciones seguras VPN basadas en software libre», Licenciatura, Universidad Central de Venezuela, ene. de 2019.

[21] K. Hasan, «Implementation of IaaS private cloud using openstack cloud management platform», Master, Daffodil International University, abr. de 2015.

[22] C. Hofer y G. Karagiannis, «Cloud computing services: taxonomy and comparison», *This article is published with open access at Springerlink.com*, 2011.

[23] ——, «Architectural Comparison and Implementation of Cloud Tools and Technologies», *International Journal of Future Computer and Communication*, vol. 3, n.º 3, 2014.

[24] IIC, *The Industrial Internet of Things Volume G8: Vocabulary*, 2015.

[25] IMC, *Cómputo en la nube: nuevo detonador para la competitividad de México*, 2012.

[26] ISO, *TR 15443-3: 2007-12-15 (1st ed.)* 2007.

[27] ——, *Information technology – Cloud computing – Overview and vocabulary ISO/IEC 17788:2014*, International Organization for Standardizatio e International Electrotechnical Commission, 2014.

[28] D. Johnson, M. Kiran, R. Murthy, R. Suseendran y G. Yogesh, *Eucalyptus Beginner's Guide – UEC Edition*. CSS Corp. Pvt. Ltd, mayo de 2010, págs. 2-16. dirección: http://www.csscorp.com/eucauecbook.

[29] L. Joyanes Aguilar, *La Computación en Nube (Cloud Computing): El nuevo paradigma tecnológico para empresas y organizaciones en la Sociedad del Conocimiento*, 2009.

[30] R. Kissel, *NISTIR 7298: Glossary of Key Information Security Terms*, 2013.

[31] A. Koschel y S. Hofman, *Standardization in Cloud Computing*, 2014.

[32] I. Kotuliak, P. Rybar y P. Trúchly, «Performance Comparison of IPSec and TLS Based VPN Technologies», 2011.

[33] R. Kumar, K. Jain, H. Maharwal, N. Jain y A. Dadhich, «Apache CloudStack: Open Source Infrastructure as a Service Cloud Computing Platform», *International Journal of Advancement in Engineering Technology, Management& Applied Science*, 2014.

[34] S. Kumar, S. Kumar, A. Kumar y A. Sumitra, «A Survey on Cloud Based Softwares (Eucalyptus vs OpenStack)», *International Journal of Computer & Mathematical Sciences*, vol. 5, n.º 5, 2016.

[35] D. López, «La "computación en la nube" o "cloud computing" examinada desde el ordenamiento jurídico español», *Revista de Derecho de la Pontificia Universidad Católica de Valparaíso*, 2013.

[36] A. Mansaf y A. S. Kashish, «Recent Developments in Cloud Based Systems: State of Art», ene. de 2015. dirección: https://arxiv.org/abs/1501.01323.

[37] D. C. Marinescu, *Cloud Computing: Theory and Practice*. Morgan Kaufmann, 2013.

[38] P. Mell y T. Grance, *Special Publication 800-145*, 2011.

[39] S. Meyer, P. Healy, T. Lynn y J. Morrison, «Quality Assurance for Open Source Software Configuration Management», en *2013 15th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*, sep. de 2013, págs. 454-461. DOI: 10.1109/SYNASC.2013.66.

[40] S. Pandey, *Investigating Community, Reliability and Usability of CFEngine, Chef and Puppet*, mayo de 2012.

[41] G. Pierantoni, T. Kiss y G. Terstyanszky, «Towards Cloud Application Description Templates Supporting Quality of Service», *WSG 2017, 9th International Workshop on Science Gateways*, 2017.

[42] M. Queiroz, P. P. Tallon, R. Sharma y T. Coltman, *The Role of IT Application Orchestration Capability in Improving Agility and Performance*, 2017.

[43] J. Rahman, «Investigating Configuration Management Tools Usage in Large Infrastructure», Master, University of OSLO, mayo de 2012.

[44] A. Raihan y M. Afroze, *SECURING A NETWORK BY USING VLAN, PORT SECURITY AND ACCESS CONTROL LIST*, 2016.

[45] SEREM, *Automatización de procesos,primer paso para la Transformación Digital*, 2017.

[46] N. Sabharwal y R. Shankar, *Apache CloudStack Cloud Computing*. Packt Publishing Ltd., 2013.

[47] O. Sefraoui, M. Aissaoui y M. Eleuldj, «OpenStack: Toward an Open-Source Solution for Cloud Computing», *International Journal of Computer Applications*, vol. 55, n.º 3, pág. 38, 2012.

[48] J. L. Shah, *Cloud Computing: The Technology for Next Generation*, 2014.

[49] J. Téllez Valdés y E. Ibarra, *Lex cloud computing: Estudio jurídico del cómputo en la nube en México*, 2013.

[50] X. Wen, G. Gu, Q. Li, Y. Gao y X. Zhang, «Comparison of Open-Source Cloud Management Platforms: OpenStack and OpenNebula», *Piscataway, NJ IEEE 2012*, mayo de 2012.

[51] «White Paper: Enhanced Control, Orchestration, Management & Policy (ECOMP) Architecture», AT&T Inc., inf. téc., nov. de 2016, págs. 7,17-19. dirección: https://about.att.com/content/dam/snrdocs/ecomp.pdf.

[52] P. Wouters y K. Bantoft, *Building And Integrating Virtual Private Networks With Openswan*. Packt Publishing, 2006, ISBN: 1-904811-25-6.

[53] B. eldhuizen, «Automated state machine learning of IPsec implementations», Bachelor, Radboud University, ago. de 2017.

[54] F. Önnberg, «Software Configuration Management: A comparison of Chef, CFEngine and Puppet», Master, University of Skövde, jun. de 2012.