Universidad Nacional Autónoma de México

Facultad de Contaduría y Administración

Construcción de un clúster prototipo para el estudio comparativo de la funcionalidad de las herramientas de monitoreo CFEngine y Nagios en la Coordinación de Supercómputo de la DGTIC.

Diseño de un Sistema o Proyecto

Eduardo Iván Ortega Alarcón

México, D.F. **2015**

Universidad Nacional Autónoma de México

Facultad de Contaduría y Administración

Construcción de un clúster prototipo para el estudio comparativo de la funcionalidad de las herramientas de monitoreo CFEngine y Nagios en la Coordinación de Supercómputo de la DGTIC.

Diseño de un Sistema o Proyecto

Que para obtener el título de:
Licenciado en Informática

Presenta:
Eduardo Iván Ortega Alarcón

Asesor:
M.I. Lourdes Yolanda Flores Salgado

México, D.F. **2015**

Dedicatorias

Dedicado a mi familia, a mi padre, madre, hermano, hermana y sobrina.

Agradecimientos

A mi familia, a mi padre, mi madre y mis hermanos que siempre me han apoyado a lo largo de mi vida. Gracias, siempre serán un ejemplo para mi.

A la Universidad Nacional Autónoma de México y la Facultad de Contaduría y Administración, que me dieron mi formación profesional.

A mis profesores de la Facultad, gracias por sus enseñanzas y experiencias.

A la Maestra Lourdes Yolanda Flores Salgado, por el apoyo en la realización de éste trabajo, gracias por sus enseñanzas y experiencias.

A la Coordinación de Supercómputo y al Maestro José Luis Gordillo Ruiz.

A mis amigos que me apoyaron e impulsaron a continuar con éste trabajo. Gracias Arlette, Pedro, Jorge, Yesenia, Gabriela, Francisco, Fernando...

Índice de contenido

Introducción

Problema de investigación

Antecedentes

El supercómputo, también llamado cómputo numérico intensivo, es el procesamiento de datos a partir de computadoras con grandes recursos informáticos, utilizadas principalmente para la investigación y el desarrollo tecnológico.

Actualmente el supercómputo se compone de computadoras individuales conectadas entre sí, que deben tener la integridad suficiente para funcionar como si fuera sólo una, por lo cual la administración de éste tipo de máquinas no es tarea sencilla, pues hablamos de cientos (y a veces miles) de equipos que deben de ser administrados con el fin de conseguir integridad, seguridad y disponibilidad en cada uno de ellos.

La administración de sistemas es el área de la informática que se encarga de diseñar, implementar y mantener un sistema en óptimas condiciones, tratando de llegar al estado deseado de éste. Sin embargo, no es lo mismo administrar un sistema que cien sistemas, por lo que se presentan varios problemas, algunos de ellos son:

- Tiempo de administración
    - La administración de muchos sistemas trae consigo un incremento en el tiempo de administración de éstos, perjudicando la integridad, seguridad y disponibilidad.
- Probabilidad de error
    - Teniendo el factor humano siempre existe una probabilidad de error y al tener varios sistemas, la probabilidad de que exista un error aumenta.

Las tareas de administración se pueden realizar de tres formas distintas; administración manual, automatización por *scripts* y herramientas especializadas en la automatización.

Administración manual

Por su naturaleza, ésta solución no es la óptima, pues como se mencionó antes, el factor humano hace que se tenga una alta probabilidad de error, además de que consumiría mucho tiempo el monitoreo manual, siendo así una tarea cansada e ineficiente de la administración.

Automatización por scripts

Una de las tareas de un administrador de sistemas, es documentar el comportamiento de su(s) sistema(s). Esto puede ser a través de documentos oficiales de la organización, de manera personal o incluso en *scripts*.

Al crear *scripts* para la administración y el monitoreo, se resuelve en su mayor parte el peligro del error humano, pues el tenerlo hace que la tarea se automatice y siempre se realicen las mismas acciones al ejecutarlo.

Es una solución sencilla, pero que en infraestructuras de muchos sistemas la tarea se complica, pues hay que ejecutar los *scripts* en todos y cada uno de los sistemas, además que (aunque eso se pretende) no todos los sistemas son iguales y se pueden presentar inconsistencias o complicaciones que, al final, deben solucionarse manualmente.

Otro problema es cuando los *scripts* crecen debido a su complejidad y comienzan a perder claridad, a pesar de la documentación que se tenga.

Herramientas especializadas en la automatización

Existen muchas herramientas de automatización (también llamadas herramientas de administración de la configuración) que sirven precisamente para automatizar las tareas de un administrador de sistemas. Éstas herramientas llevan a cabo muchas tareas, como lo son el diagnóstico, monitoreo, configuración, entre otros. La ventaja principal, es que pueden utilizarse con un sistema o con miles de ellos, teniendo que configurar sólo uno para obtener su réplica en los restantes.

Este tipo de herramientas son muy útiles por varias razones:

- Reduce errores.
- Mejora el tiempo de administración.
- Monitorea constantemente.
- Sólo envía advertencias si es que ocurre algo fuera de lo común.
- Obtiene información relevante acerca de el estado del sistema (o sistemas).

Estado deseado

El trabajo de un administrador de sistemas es, entre otras cosas, automatizar un sistema para que se encuentre en el “estado deseado”.

Se conoce como estado deseado cuando un sistema funciona como se espera que funcione, es decir, que su comportamiento esté dentro de lo previsto por su administrador.

Cuando se administra un sistema utilizado por más de un usuario, se obtienen comportamientos inesperados, regularmente causados por fallas en software o hardware. Estas fallas pueden ser accidentales o intencionales, pero de cualquier manera, representan problemas para el sistema y puede verse comprometida la integridad, disponibilidad o seguridad.

Justificación

Actualmente en la Coordinación de Supercómputo se utiliza la herramienta Nagios para el monitoreo de su infraestructura de cálculo numérico intensivo. El problema reside en que genera mucha información innecesaria para los administradores. Se debe instalar y configurar la herramienta CFEngine 3.5 para poder realizar la comparación entre las herramientas. A partir de ello, se podrá conocer qué herramienta realiza el monitoreo de manera más eficiente, además de saber qué herramienta reportará las incidencias realmente importantes.

Objetivo General

Realizar un análisis comparativo entre de la funcionalidad de las herramientas de monitoreo Nagios y CFEngine con el fin de determinar qué herramienta es la más óptima para la administración de un clúster de supercómputo.

Objetivos particulares:

- Crear un clúster prototipo con 47 nodos de cálculo.
- Instalar un sistema operativo en el clúster.
- Instalar la herramienta Nagios en el clúster.
- Instalar la herramienta CFEngine en el clúster.
- Analizar el funcionamiento de cada herramienta instalada.
- Comparar los resultados de los análisis previos de las herramientas.

Alcance

Se planea obtener un clúster prototipo funcionando con la distribución GNU/Linux CentOS 6.5 con CFEngine 3.5 instalado, aplicando las configuraciones necesarias y realizando algunas tareas de administración para la comparación de su funcionalidad con la herramienta utilizada actualmente.

Aportaciones

El proyecto se desarrolló tomando en cuenta las necesidades de rendimiento de la supercomputadora Miztli en la Coordinación de Supercómputo. Se espera que la implementación de CFEngine optimice el rendimiento al realizar el monitoreo de los nodos de cálculo numérico intensivo, así como mejorar la generación de información que concierne a los incidentes que deben ser reportados al administrador del sistema.

Se pretende que el trabajo escrito sirva como fuente de información sobre el supercómputo y sea una fuente de consulta sobre la administración de clústeres para los estudiantes de la Universidad Nacional Autónoma de México y en general para los administradores de sistemas que trabajan con clústeres de alto rendimiento.

El trabajo escrito también pretende ser una fuente de información en español de la herramienta CFEngine, ya que actualmente no existe mucha información sobre ésta herramienta en el idioma español.

Estructura capitular

Capítulo I

Se detalla brevemente la historia de la Universidad Nacional Autónoma de México, así como la historia de lo que actualmente es la Dirección General de Cómputo y de Tecnologías de Información y Comunicación.

Organigramas, de DGTIC y de la Coordinación de Supercómputo.

Capítulo II

Se define el marco teórico que apoyara al resto del trabajo escrito. Se intenta explicar el problema que se tiene para la administración de sistemas y el porqué es necesaria una herramienta de automatización.

Capítulo III

Planteamiento de la solución al problema, especificando las medidas necesarias de acuerdo a las actividades de la Coordinación de Supercómputo.

Capítulo IV

Resultados y conclusiones.

Anexos

Definición de algunos conceptos utilizados en el trabajo, así como material que apoya al mismo.

Bibliografía

Capítulo I

Universidad Nacional Autónoma de México

Historia

La Universidad Nacional Autónoma de México fue fundada el 21 de septiembre de 1551 con el nombre de Real Universidad de México, que en 1955 cambió el nombre a Real y Pontificia Universidad de México, creada por cédula real de Carlos V y firmada por su hijo el Príncipe de Asturias (futuro Felipe II), inaugurando sus cursos el 25 de enero de 1553, siendo virrey don Luis de Velasco y Ruiz de Alarcón.

La Universidad Nacional fue inaugurada el 22 de septiembre de 1910 como parte del programa de festejos del Centenario de la Independencia de México. La ceremonia se organizó en la Escuela Nacional Preparatoria, en el anfiteatro construido por el arquitecto Samuel Chávez para tan digno evento. La concurrencia fue numerosa, entre profesores, alumnos, integrantes de las universidades invitadas, directores de institutos y varios miembros de la élite porfiriana que se dieron cita a la fiesta inaugural que dio comienzo a las 10:30, justo a la hora que llegó el presidente Porfirio Díaz y el primer rector de la Universidad Nacional, Joaquín Eguía Lis.

El discurso inaugural fue pronunciado por Justo Sierra, ministro de Instrucción Pública, a quien debemos reconocer el importante legado reflejado en la creación de la Universidad Nacional y cuyo principal objetivo dijo, se sustentaría en la idea de lograr una educación que emane de la acción científica y cultural en pro del porvenir y unificación del país. La solemne inauguración culminó a la una de la tarde con la Procesión Universitaria, que partía del anfiteatro hacia las oficinas de la nueva institución.

La autonomía de la Universidad se obtuvo el 28 de mayo de 1929 cuando el Presidente Emilio Portes Gil se la otorgó y autorizó la construcción de Ciudad Universitaria. A partir de esto, quedó establecida como Universidad Nacional Autónoma de México (UNAM).

En el año de 1920, el Rector José Vasconcelos crea una de las iniciativas más importantes, la ley que establece el escudo y el lema de la institución, "Por mi raza hablará el espíritu", junto con la imagen del águila y el cóndor que rodean el mapa que representa a la América Latina, desde la frontera norte de México hasta el Cabo de Hornos.

Con el paso de los años nuevas instalaciones fueron construidas para la UNAM en su Ciudad Universitaria, entre ellas el Estadio Olímpico Universitario que empezó a construirse el 7 de agosto de 1950 y fue inaugurado el 20 de noviembre de 1952 , y la Biblioteca Central, inaugurada el 5 de abril de 1956. Para la década de los setenta, se llevó a cabo un gran programa de expansión, en el cual se crearon las cinco sedes del Colegio de Ciencias y Humanidades, así como las cinco unidades multidisciplinarias, actualmente todas con el nombre Facultad de Estudios Superiores: FES Acatlán, FES Aragón, FES Cuautitlán, FES Iztacala y FES Zaragoza; adicionalmente, se crearon en el año 2011 y 2012, campus universitarios en Guanajuato y Michoacán. ENES León y ENES Morelia.

A finales del 2005, la UNAM ha sido reconocida internacionalmente como una de las mejores universidades de Hispanoamérica (América Latina, España y Portugal) por el diario inglés The Times) y como la número 45 en el ranking mundial, que coincide también con la clasificación internacional llevada a cabo por la Universidad de Shanghái. Esta posición mejoró en el año 2006, cuando la UNAM alcanzó el lugar 54 gracias a las áreas de Humanidades y Artes, y actualmente está en la posición número 38 de las mejores universidades del mundo, por lo cual es una de las mejor ubicadas de América Latina, España y Portugal. Otras clasificaciones internacionales la ubican en la posición 1 en Latinoamérica.

Dirección General de Cómputo y de Tecnologías de Información y Comunicación

Historia

El 14 de octubre de 1981 se inauguró el Programa Universitario de Cómputo (PUC), por el Rector de la Universidad Nacional Autónoma de México, el Dr. Octavio Rivero Serrano.

En ese entonces, lo que hoy se conoce como DGTIC, sólo contaba con cuatro áreas; Cómputo para la Docencia, Cómputo para la Administración Académica, Cómputo para la Administración y Cómputo para la Investigación.

Debido al gran crecimiento y actualización constante de las Tecnologías de Información y Comunicación, el 14 de mayo de 1985 el Programa Universitario de Cómputo se convirtió en la Dirección General de Cómputo Académico, siendo parte de la Secretaría de Servicios Académicos (que tiempo después, se integró a la estructura de la Secretaría General), para poder tener una mejor administración de las Tecnologías de Información de la UNAM.

La DGSCA contaba con las áreas; Dirección de Cómputo para la Docencia, Dirección de Cómputo para la Investigación, Dirección de Sistemas y Dirección de Telecomunicaciones, además de una Coordinación de Servicios Educativos en RedUNAM – SERUNAM.

La Dirección General de Cómputo y de Tecnologías de Información y Comunicación nació, gracias a un acuerdo realizado por el Dr. José Narro Robles, Rector de la Universidad Nacional Autónoma de México. El acuerdo se debió al crecimiento tecnológico institucional que comprende lo relativo a cómputo y a la convergencia de tecnologías digitales que permiten la comunicación entre los estudiantes de la Universidad. Se planteó que el cambio de denominación se debe a que es más adecuado a lo que actualmente se maneja como Tecnologías de Información, además de que deja una gama amplia de servicios a añadir con el tiempo para los productos educativos e innovaciones en diversos ámbitos que no sólo involucran al cómputo para poder integrar esa convergencia al ámbito universitario.

Objetivos de la organización

- Proporcionar nuevos programas de capacitación y actualización permanente, considerando la convergencia digital.
- Transformar RedUNAM en una red multimedia integral que integre más a los universitarios, con servicios tales como telefonía, voz sobre IP, videoconferencia y audioconferencia.
- Consolidar la información histórica, presente y futura de la institución, en diversos formatos, dentro de una red de acervos digitales (RedUNAM), sustentada por la más avanzada tecnología de seguridad y resguardo de la información, de tal forma que todo contenido educativo, todo resultado de alguna investigación o todo recurso

cultural esté al alcance de los universitarios desde cualquier dispositivo de información digital.

- Posicionar a la UNAM a la vanguardia como una institución digital, lo que requerirá desde la formación de su comunidad en todo este abanico de tecnologías hasta la constante innovación en cómo esos recursos se integrarán a la misión y objetivos de la Universidad.
- Apoyar a la institución en la mejora permanente de sus procesos administrativos con la ayuda de tecnologías de información y comunicación que den certidumbre, pertinencia y eficiencia a tales procesos a un costo más reducido.
- Impulsar la participación de la UNAM en la protección al medio ambiente, reduciendo el consumo de papel, impresos, mensajería y transportes, siendo sustituida esta información por recursos digitales.
- Ampliar el impacto de la UNAM en beneficio de la sociedad en general por medio de la integración de tecnologías que faciliten un mayor acceso de todo tipo de personas a los acervos públicos de la institución, sin importar su ubicación geográfica o capacidades físicas.

Misión

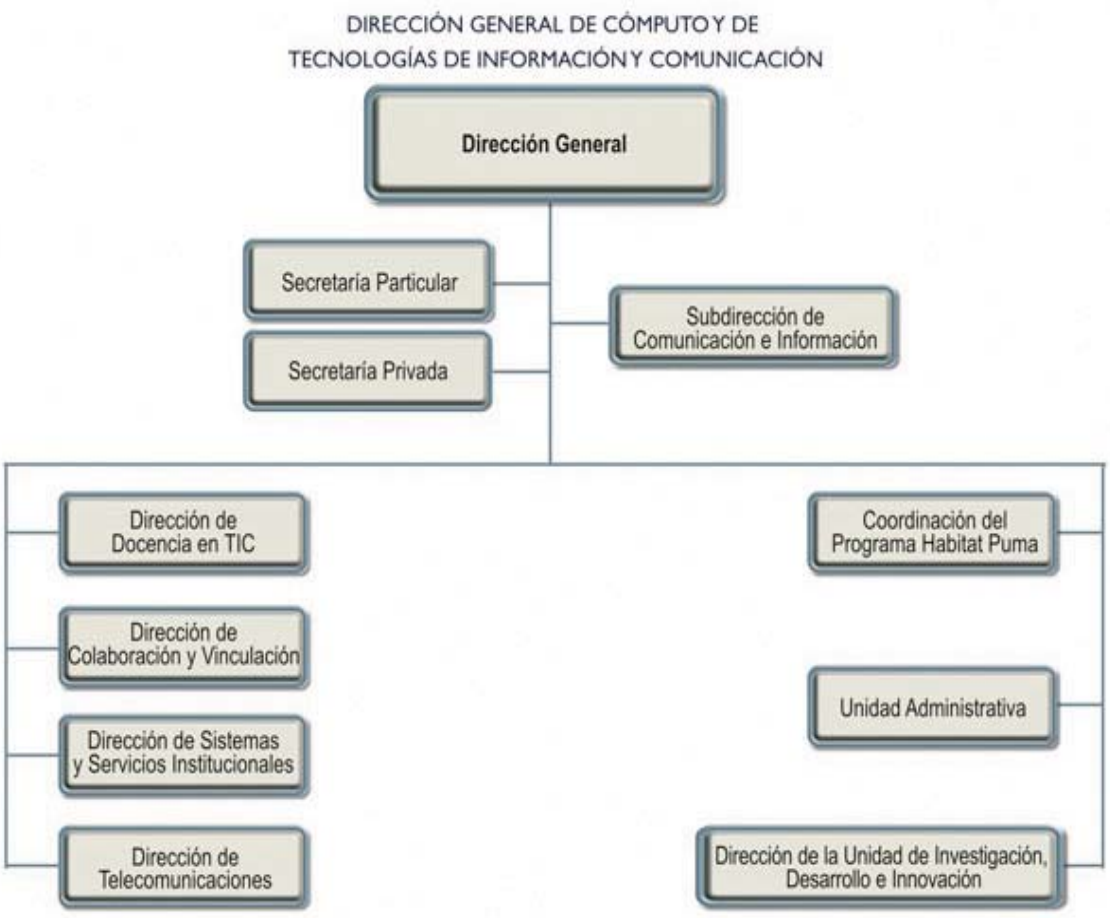
La Dirección General de Cómputo y de Tecnologías de Información y Comunicación (DGTIC), contribuye al logro de los objetivos de la UNAM como punto de unión de la comunidad universitaria para aprovechar los beneficios que las tecnologías de la información y las comunicaciones pueden aportar a la docencia, la investigación, la difusión de la cultura y la administración universitaria.

Funciones

- Mejora de procesos administrativos con la ayuda de tecnologías de información y comunicación
- Proporcionar una red informática para los universitarios (RedUNAM).

- Conservar la información histórica, presente y futura de la institución dentro de los acervos digitales, para que esté al alcance de los universitarios por medio de diversos dispositivos.
- Facilitar el acceso de los acervos públicos de la institución por medio de tecnologías de información.
- Implementar programas de capacitación en cómputo científico para los universitarios.

Organigrama de la institución



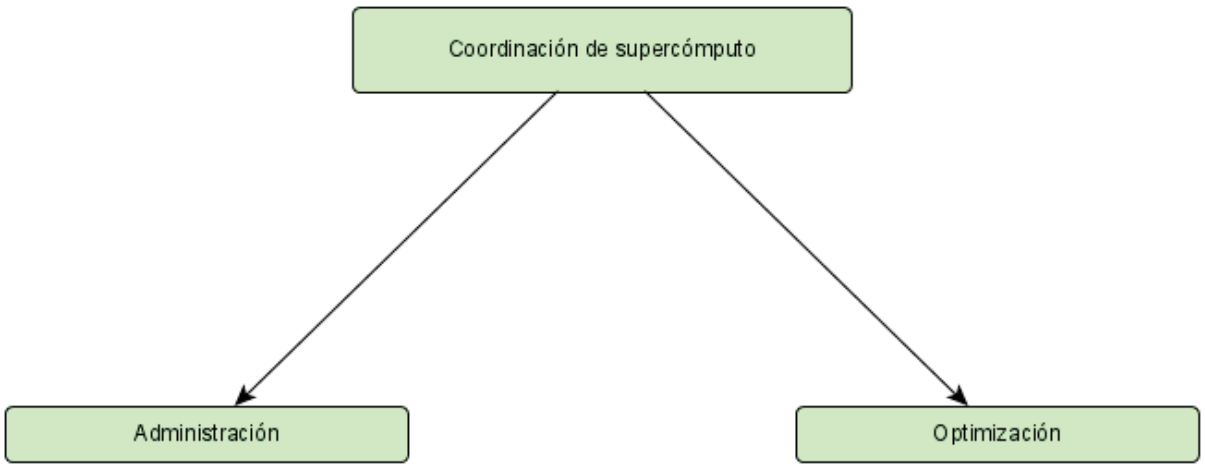
Coordinación de supercómputo

El supercómputo en la UNAM se practica en diversos institutos, tales como el Instituto de Astronomía, Ciencias Nucleares, Geociencias, Geofísica y Biotecnología. De igual manera la coordinación de supercómputo administra una supercomputadora utilizada principalmente para la investigación.

El supercómputo es la utilización de computadoras con capacidades extraordinarias (supercomputadoras) para la realización de investigación en diversas áreas del conocimiento, que van desde el estudio de la estructura del universo hasta el comportamiento de partículas subatómicas (Dirección General de Cómputo y de Tecnologías de Información y Comunicación, 2014).

La coordinación de supercómputo actualmente cuenta con dos áreas; administración y optimización.

Organigrama de la Coordinación



Capítulo II

La Coordinación de Supercómputo, perteneciente a la Dirección General de Cómputo, Tecnologías de Información y Comunicación (DGTIC), de la Universidad Nacional Autónoma de México, cuenta con una infraestructura de supercómputo, la cual conlleva a tener la necesidad de contar con una herramienta de monitoreo de sistemas que cumpla con características específicas que se adecuen a la infraestructura y que permita realizar las labores de monitoreo y administración centralizada de la forma más sencilla posible.

Informática y Tecnologías de Información

La informática, que es la disciplina encargada de el manejo, procesamiento y automatización de la información a partir de datos sin significado propio, puede resolver la necesidad del monitoreo de sistemas e infraestructuras computacionales, ya que una de las áreas de ésta disciplina, es precisamente la administración de sistemas.

Una infraestructura de supercómputo no sólo consta en administrar un sistema (por muy grande que pueda ser), sino también de la manutención de una red informática y su correcta configuración. Por lo tanto, no basta con procesar y automatizar la información, sino también se debe de mantener un medio por el cual esa información pueda transmitirse y almacenarse de manera íntegra. Aquí es donde entran las TIC.

El concepto de Tecnologías de la Información y Comunicación (TIC) está relacionado directamente con la informática, sin embargo no son lo mismo. Como su nombre lo indica, las TIC son tecnologías que se utilizan para los servicios que pretenden mejorar la productividad o incluso la calidad de vida a partir del uso de la informática, que como se mencionó anteriormente, se encarga del manejo y procesamiento de la información directamente, así como de su estudio para la toma de decisiones.

Informática y cómputo científico

El cómputo científico se enfoca a la construcción de modelos matemáticos y métodos numéricos para la resolución de problemas científicos, de ingeniería o de ciencias sociales. Generalmente requiere de grandes cantidades de cálculo puesto que los

problemas pueden pertenecer a áreas como la astronomía o la física computacional, por lo tanto es necesario procesar todos esos cálculos en una computadora con mayor poder de procesamiento, es decir, una supercomputadora.

Para que los cálculos científicos puedan realizarse con integridad y lo más rápido posible, se debe tener una infraestructura de supercómputo en óptimas condiciones; que los cálculos estén distribuidos, el almacenamiento disponible, los respaldos actualizados, entre otras.

Se recurre a la informática para poder automatizar las soluciones a todos estos problemas, puesto que al administrar una infraestructura de supercómputo, la necesidad de automatizar procesos es inminente.

Supercómputo

Hasta este momento, se ha hablado sobre sistemas de gran tamaño, gran cantidad de sistemas y cómputo científico. Todos estos términos son parte de lo que se conoce como supercómputo.

El supercómputo es la disciplina en donde se utilizan computadoras con grandes recursos informáticos para poder resolver problemas como los que presenta el cómputo científico. Generalmente, éstas computadoras se dividen en dos tipos:

- Supercomputadora.
- Clúster.

Se le llama **supercomputadora** cuando es un dispositivo o máquina con grandes recursos, pero que forman parte de un sólo dispositivo, es decir, que todos sus componentes pertenecen a la misma computadora, ya sea procesadores, memoria, almacenamiento, dispositivos de entrada o salida, etc. Comúnmente éste tipo de computadoras tienen varios procesadores interconectados entre sí, además de tener una gran cantidad de memoria a disposición de los procesos que ejecuta.

Por el contrario, un **clúster** se compone de un grupo de varias computadoras completamente independientes en cuanto a hardware (también llamados “nodos”), ya que cada una de ellas tiene su(s) procesador(es), memoria, almacenamiento, etc. Sin

embargo, los nodos de un clúster están interconectados por medio de una o más redes informáticas, y debe dar la impresión de ser sólo una computadora con grandes capacidades. Al estar conectados entre sí, los nodos de un clúster pueden compartir recursos por medio de la red, así como repartir las tareas de administración entre ellos.

La problemática comienza al administrar todos esos sistemas para que funcionen como uno, porque no sólo se deben tener funcionando individualmente, sino también en conjunto con los demás sistemas. Deben de comunicarse para diversas tareas como lo son:

- Comunicación entre procesos.
- Configuraciones uniformes.
- Comunicación para el almacenamiento.
- Compartición de memoria.
- Balance de carga.

Si un sistema no tiene la eficiencia suficiente para realizar este tipo de tareas, puede resultar en una falla de sistema, de usuario, la ejecución de los procesos puede demorar demasiado o se pueden obtener resultados erróneos.

La eficiencia es una de las cualidades más importantes que un sistema debe tener, no sólo en supercómputo, sino en cualquier computadora que se utilice regularmente. La necesidad de tener eficiencia nace cuando se ejecutan varios procesos en un sistema y el procesador debe dar a cada proceso un intervalo de tiempo (de acuerdo a su prioridad) para que se ejecute. Si se cuenta con eficiencia, este procedimiento se realizará sin ningún problema además de que el hecho de que se ejecuten muchos procesos a la vez, no debe perjudicar a los usuarios.

Arquitectura de sistemas de supercómputo.

Actualmente la mayoría de las computadoras, incluso las computadoras personales, son sistemas paralelos, es decir, tienen más de un procesador, más de un núcleo, o varios procesadores con varios núcleos. Para poder aprovecharlos de manera eficiente, se

deben tener programas que se ejecuten en todos los procesadores (o núcleos) simultaneamente. Regularmente se utilizan programación en paralelo para el cómputo numérico intensivo realizado por una supercomputadora, porque al tener grandes cantidades de datos y de procesamiento, se tiene que minimizar el tiempo de ejecución de los procesos.

Cuando ejecutamos varios programas en un solo procesador, el sistema operativo le da intervalos de tiempo de ejecución a cada proceso creado, de manera que sólo se ejecuta un proceso a la vez. Los intervalos de tiempo son muy pequeños para que una persona pueda notarlos y nos da la impresión que se ejecutan todos los procesos al mismo tiempo aunque no sea así. A pesar de que esos intervalos son en microsegundos, el tener muchos procesos para un solo procesador hace que los procesos tarden más en terminar. Cosa que es muy diferente cuando se tienen dos o más procesadores. En ese caso, cada procesador ejecuta un proceso de manera que se tienen el mismo número de procesos ejecutándose que el número de procesadores, además de que cada procesador también da intervalos de tiempo si es que tiene más procesos pendientes que ejecutar.

La programación en paralelo se utiliza para que un solo proceso, se ejecute en varios procesadores o núcleos, asignando diferentes tareas del programa a diferentes procesadores de la computadora. Lo que sí se tiene es sólo una memoria para poder almacenar temporalmente los datos del programa. Existen dos tipos de esquemas de memoria para poder manipular la programación en paralelo; memoria compartida y memoria distribuida.

Memoria compartida

Se le conoce como memoria compartida al tipo de memoria que puede ser accedida por diferentes procesos o procesadores dentro de un mismo sistema informático. En la ilustración 1 se muestra un esquema ejemplificando la memoria compartida en un clúster.

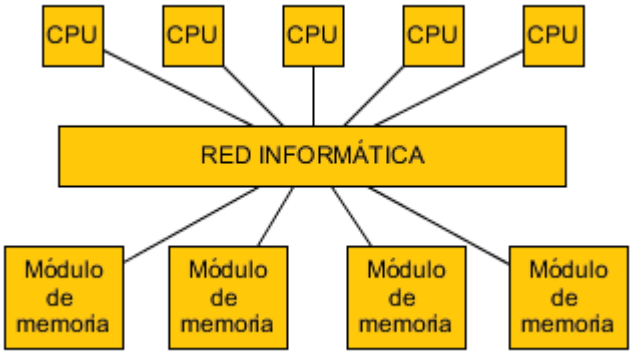


Ilustración 1: Esquema de memoria compartida en red.

Para las supercomputadoras reales, es decir, las que no son un clúster, la memoria es compartida por todos los procesadores dentro del mismo sistema sin tener comunicación de red con algún otro módulo de memoria, puesto que se cuenta con una gran cantidad de memoria local como de muestra en la ilustración 2.

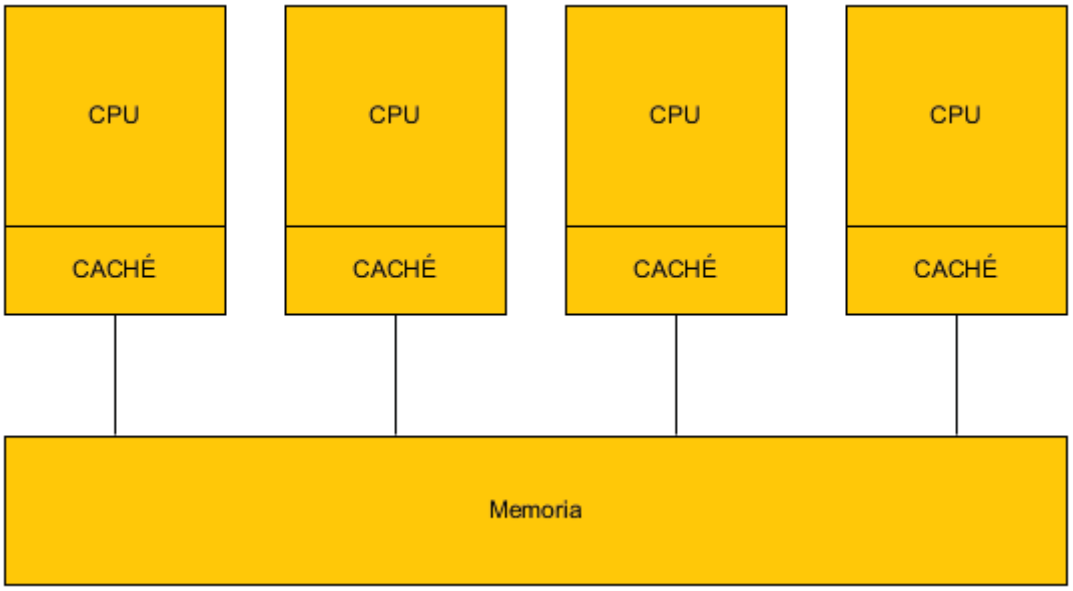


Ilustración 2: Memoria compartida en una supercomputadora.

Memoria distribuida

La memoria distribuida, también conocida como memoria compartida distribuida, es aquella donde la memoria se encuentra físicamente en lugares diferentes pero que para los procesos debe funcionar como una sola. Comúnmente éste paradigma se utiliza en los clústeres ya que cada nodo de ellos puede contar con su memoria propia y además compartirla para los demás nodos del clúster (ilustración 3).

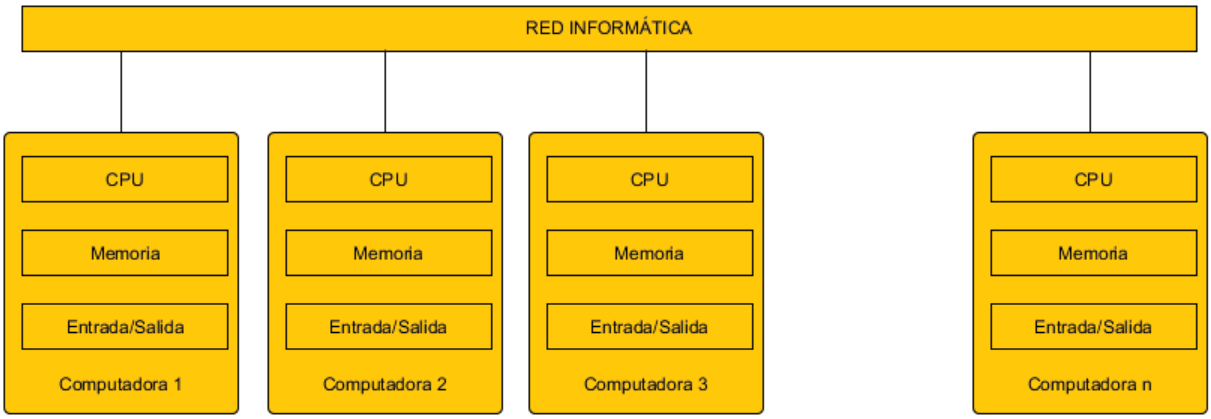


Ilustración 3: Memoria compartida distribuida

Es un paradigma muy eficiente, porque cualquier proceso que necesite memoria, principalmente la solicitará del mismo nodo donde está corriendo y sólo si la memoria local no fuera suficiente, es entonces cuando solicita recursos de los nodos vecinos.

A pesar de ser la solución más eficiente, es también la más compleja y difícil de programar.

Administración de sistemas y supercómputo

Cuando se tiene una infraestructura de supercómputo, la administración de sistemas se convierte en una tarea más compleja. La cantidad de computadoras aumenta, el número de procesos ejecutados (por el sistema y por usuarios) incrementa exponencialmente, la optimización del sistema debe ser más precisa, la disponibilidad debe ser casi del 100%, etc.

Políticas

Para la correcta administración y uso de un sistema (sin importar su tamaño), es necesaria la implementación de políticas, con el objetivo de que el sistema tenga un estado previsible y ordenado.

La real academia española (RAE), en su diccionario de la lengua española, define política como “orientaciones o directrices que rigen la actuación de una persona o entidad en un asunto o campo determinado”.[1]

Por lo tanto las políticas en administración de sistemas se podrían definir como directrices que tienen por objetivo establecer lineamientos que los usuarios deben seguir, para asegurar la integridad de los datos, sistemas, redes y procedimientos de un sistema de cómputo.

Las políticas pueden variar dependiendo del tipo de sistema que se tenga, pero existen una serie de principios de administración de sistemas que difícilmente cambian en cualquier tipo de sistema.

Principios de administración de sistemas

Cimentación

La administración de sistemas comienza con una política que decida el comportamiento deseado en relación con el comportamiento posible.

No siempre el estado deseado es posible, sin embargo se debe evaluar la posibilidad de tener la máxima aproximación a ese estado deseado.

Previsibilidad

La administración de sistemas busca trabajar siempre con un sistema previsible. Con esto se obtiene confiabilidad, seguridad e integridad.

1 Real Academia Española. (2001). *política*. Julio 2, 2014, de RAE Sitio web: http://lema.rae.es/drae/?val=pol%C3%ADtica

El tener un sistema previsible, hace que cualquier siniestro sea reparable, para que la integridad no se vea afectada.

Escalabilidad

Un sistema se considera escalable cuando crece apegándose a la política establecida y mantiene su previsibilidad aún después de aumentar su tamaño.

La previsibilidad juega un papel muy importante dentro de la escalabilidad, ya que si es un sistema previsible, será más fácil que sea escalable.

Interdependencia

Un sistema es más vulnerable cuando sus sistemas dependen los unos de los otros. Cuando se disminuyen las dependencias, la previsibilidad y confiabilidad aumentan.

Suponiendo que todos los nodos de un clúster dependen de un nodo maestro y sin éste no funcionen, cuando el nodo maestro deje de funcionar, teóricamente todos los demás también lo harán, muy diferente a si todos los sistemas son dependientes de ellos mismos.

Separación de datos

Dentro del sistema operativo, los diferentes tipos de datos deben tener una estructura diferente de directorios. Esto facilita la administración e incluso la eficiencia.

Si el sistema tiene los datos separados, las tareas de actualización, reinstalación y mantenimiento serán más sencillas.

Simplicidad

Los usuarios tolerarán las reglas siempre y cuando éstas sean fáciles de entender.

En el momento en que las políticas son complejas, un usuario puede faltar a la regla incluso sin que él lo considere incorrecto, debido a la complejidad de la misma.

Libertad

Las restricciones para los usuarios, deben ser minimizadas en la medida de lo posible.

Cuando hay muchas restricciones, el sistema se vuelve complejo y en algunos casos inutilizable.

Autoridad

Se debe mantener el control y monitoreo del sistema para evitar el abuso del mismo.

Aunque el sistema tenga políticas de uso, es obligación del administrador hacer un monitoreo regular, en caso de cualquier eventualidad, deberá aplicar las medidas pertinentes.

Hostigamiento

El abuso en el uso de un recurso público, puede ser considerado como hostigamiento por aquellos usuarios que lo comparten.

Cuando un usuario inicia y crea una sesión en un sistema, pone en riesgo el entendimiento del sistema para los demás usuarios.

Fallas humanas

No existe un sistema perfecto, por lo cual no se pueden evitar todas las fallas. Para esto existen planes de contingencia en el caso de que ocurra una falla humana, intencionada o accidental.

Uniformidad y variación

Una configuración uniforme minimiza el número de diferencias y excepciones que se tomarán en cuenta incrementando la previsibilidad estática del sistema. Sin embargo una variación en la configuración puede minimizar el riesgo de que todo el sistema falle por un evento particular. Si una falla en el sistema ocurre en una forma esperada, los efectos

pueden ser controlados y mitigados. Si se cuenta con políticas adecuadas, la recuperación del sistema se hará de manera rápida.

Causalidad

Cada cambio o efecto ocurre en respuesta a una causa que lo precede.

Es tarea del administrador del sistema determinar el origen del cambio e identificar una cadena de eventos que resultan en el evento desafortunado.

Regularmente un cambio inesperado es el resultado de un error, pero se pueden distinguir esos errores en tres etapas que hacen pasar del estado deseado al estado opuesto.

- Una falta/falla-menor (*fault*) en un sistema surge cuando su comportamiento viola la especificación de un servicio.
- Un error es una parte del estado de un sistema susceptible de causar una falla. Esto quiere decir que no necesariamente lo provoca, o por lo menos no inmediatamente. El error está latente en tanto no provoca una falla.
- Una falla/avería (failure) es todo evento, acción o circunstancia provocada por un error o un defecto. Implica la interrupción, mal funcionamiento o degradación en la operación del sistema o un componente del mismo.

Conectividad en supercómputo

Como se mencionó anteriormente, un clúster de cómputo necesita de una o varias redes informáticas para poder realizar las tareas indicadas como si fuera una sola máquina. De esta manera, los nodos de un clúster pueden compartir e incluso dividir las diferentes tareas entre ellos para resolver problemas en paralelo, haciendo que el sistema sea más óptimo tanto en su administración como en su ejecución, Esto conlleva a que el administrador de un clúster también debe administrar la red del mismo y poderla optimizar de acuerdo a las necesidades del clúster, porque no es lo mismo un clúster que se utilizará para dar un servicio como lo es HTTP (web) o SMTP (correo), a ser un clúster que su función principal sea el cálculo numérico intensivo.

Redes informáticas

Se considera una red informática al conjunto de dispositivos conectados entre si a través de un medio físico que envía impulsos eléctricos para después ser interpretados como datos relevantes, con el objetivo de compartir información.

Su funcionamiento se basa en diversos estándares realizados específicamente para la comunicación entre dispositivos informáticos, como lo son el modelo OSI y más específicamente el modelo TCP/IP.

El modelo OSI

La ISO (Organización de Estándares Internacionales por sus siglas en inglés) creó un modelo para el uso y diseño de los protocolos de una red informática. El modelo de Interconexión de sistemas abiertos (OSI) se compone de siete capas las cuales se utilizan para describir una red o el funcionamiento de la misma. Cada una de las capas actúa de forma incremental de manera que la siguiente capa siempre dependerá de la anterior. Cabe destacar que en la especificación del modelo OSI, la ISO no asocia ningún estándar de redes o protocolos en particular.

Las capas del modelo OSI son las siguientes:

1. Física.
2. Enlace de datos.
3. Red.
4. Transporte.
5. Sesión.
6. Presentación.
7. Aplicación.

Su representación es la siguiente:

Con esta representación es más sencillo entender que cada una de las capas es dependiente de la otra, de abajo hacia arriba.

Administración de redes

Como parte del trabajo del administrador de sistemas, y más específicamente, de sistemas de supercómputo, la administración de la red es un tópico que debe tenerse en cuenta, principalmente por el impacto en la eficiencia de la infraestructura y en general de los resultados que se puedan obtener de los procesos en paralelo así como el tiempo de escritura en los sistemas de almacenamiento en red.

Deben considerarse varios aspectos en la administración de la red, como lo son la eficiencia, seguridad, estabilidad, disponibilidad, entre otros. Existe un estándar para la administración adecuada de una red informática llamado FCAPS.

FCAPS

Es un framework de administración de red creado por la ISO, con el objetivo de estandarizar la administración de redes informáticas a partir de la definición de cinco niveles:

- F (*Fault management*): Administración de fallos. En este nivel, los problemas de red se encuentran y corrigen, además los problemas probables son identificados con más facilidad y se toman medidas para prevenirlos. Esto conlleva a tener una red funcional y se reducen los problemas de disponibilidad.
- C (*Configuration management*): Administración de la configuración. La actividad de la red es controlada y monitoreada en este nivel. Eso se hace con el fin de verificar los cambios al momento que suceden, ya sea a nivel hardware o software, y que estos cambios se puedan integrar al entorno de red con facilidad.
- A (*Accounting management*): Administración de la contabilidad. Este nivel esta dirigido a dos tareas principalmente. Primero a obtener estadísticas relevantes sobre el uso de la red y qué usuarios o suscriptores están haciendo uso de ella. Y segundo para, de acuerdo a esas estadísticas, tomar las medidas pertinentes para distribuir los recursos de manera óptima.
- P (*Performance management*): Administración del rendimiento. Aquí involucra toda la parte de administración que manipula el rendimiento de la red, los cuellos de botella se intentan minimizar y el ancho de banda se aprovecha al máximo. Para poder resolver estas complicaciones, se debe identificar de dónde surgen los problemas, que es la mayor parte que corresponde a este nivel.
- S (*Security management*): Administración de la seguridad. El nivel de seguridad es el encargado de proteger la red de accesos no autorizados, *hackers* y de cualquier tipo de sabotaje, ya sea físico o software. Los administradores de la red son los encargados de decidir qué pueden o no hacer los usuarios dentro de ésta.

Arquitecturas de gestión de red

Además de seguir estándares para la administración y configuración de una red, se debe tomar en cuenta la topología de red que es óptima para las necesidades de los sistemas. Generalmente se conocen varias topologías de red en niveles básicos de operación, como la topología estrella, anillo, bus, malla, etc, pero cuando hay una infraestructura más grande, las redes se pueden clasificar en centralizadas, jerarquizadas o distribuidas.

Centralizada

Una red centralizada tiene similitud con la topología estrella, en la cual todos los nodos están conectados a un nodo central, el cual se encargará de la comunicación de todos los nodos (Ilustración 4). El problema principal de éste tipo de red es cuando el nodo central falla, ya que todos los nodos quedan incomunicados hasta que el nodo principal reanude su funcionamiento.

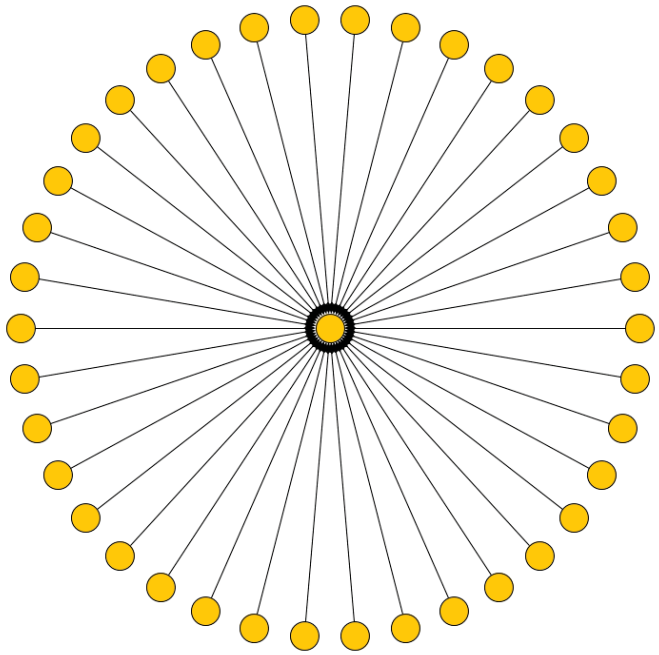


Ilustración 4: Red centralizada

Jerarquizada (descentralizada)

La principal diferencia de la red jerarquizada, es que no tiene sólo un nodo que controla a todas las máquinas, sino que tiene varios divididos en grupos o jerarquías, las cuales a su vez, están conectadas a un nodo maestro (ilustración 5). En este caso, si el nodo maestro falla por alguna razón, la mayoría de los nodos tendrán conectividad entre sí, mientras que otros no podrán comunicarse mas que con su mismo nivel de jerarquía.

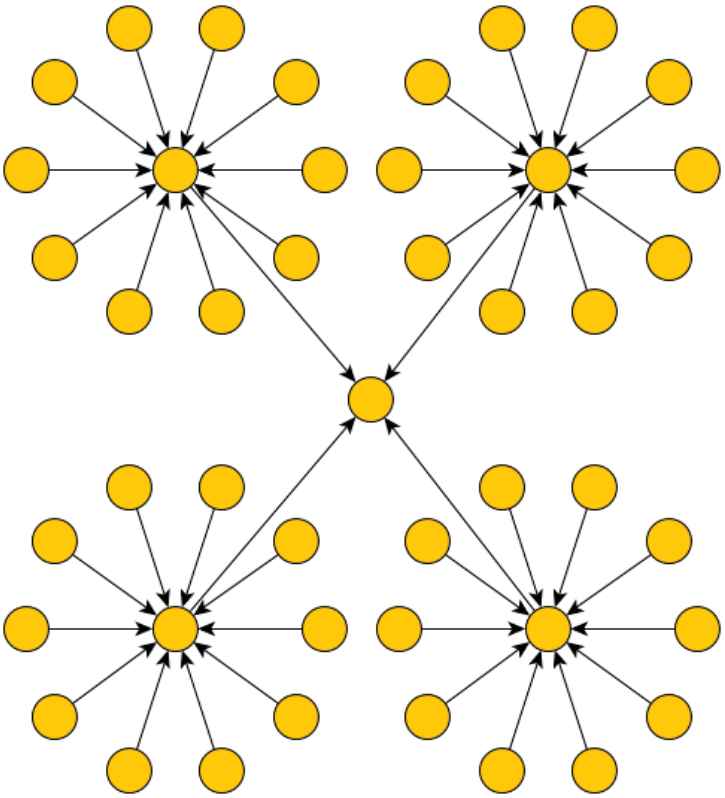


Ilustración 5: Red jerarquizada

Su ventaja es al mismo tiempo su desventaja, porque si un nodo principal de alguno de los grupos llega a fallar, todos los nodos de ese conjunto no tendrán conectividad.

También se le conoce como red descentralizada.

Distribuida

La red distribuida no necesita de ningún nodo principal o maestro, ya que todos los nodos están conectados entre sí, o al menos están conectados a 2 o más nodos (ilustración 6).

Esto permite tener mayor conectividad a pesar de que uno o varios nodos fallen, porque la red realiza caminos distintos para llegar a su destino.

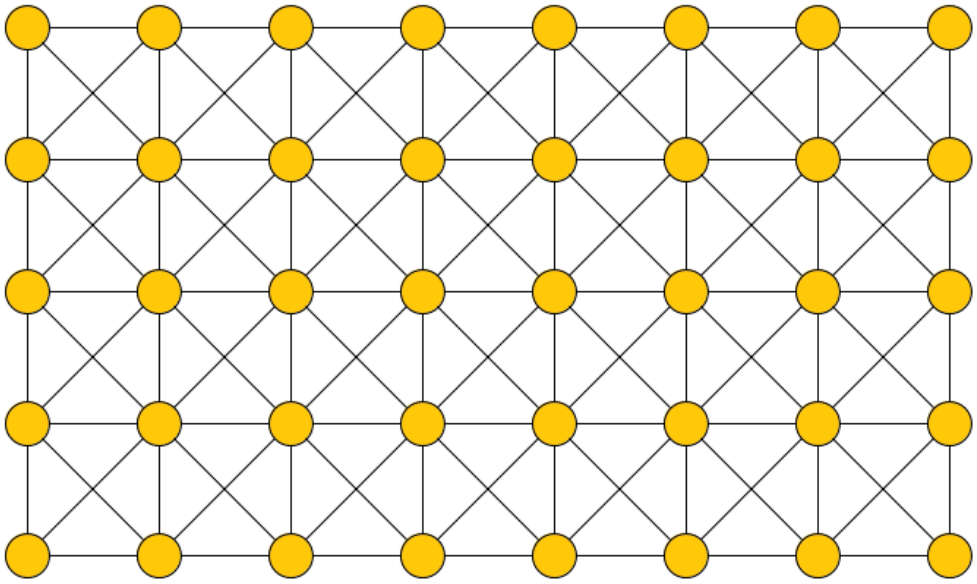


Ilustración 6: Red distribuida

La desventaja con este tipo de red es el costo que representa, ya que cada nodo debe tener más de una tarjeta de red, además de que puede verse afectado el rendimiento debido a que todos y cada uno de los nodos realizan tareas de ruteo dentro de la red.

Administración de la configuración

La configuración de un sistema es la parte esencial del funcionamiento del mismo. En sistemas de gran tamaño como lo son los clústeres, mantener una configuración adecuada no es simple. Puede darse el caso de que, por ejemplo, se tienen cien nodos, pero de esos cien nodos, 98 funcionan correctamente y los restantes se desconfiguran por cualquier razón aleatoria. Esto representa una falla potencial o incluso una vulnerabilidad para la integridad y seguridad del sistema.

Se debe procurar tener una configuración consistente en todos los dispositivos que integran el sistema. Por consistente, no quiere decir igual, porque hay nodos en un clúster que realizan una función específica dentro del sistema, como lo es un nodo maestro o los nodos de login.

Dentro de los grandes centros de cómputo, se tienen cientos e incluso miles de sistemas dando un servicio en particular, pero eso no significa que se tengan cientos o miles de administradores de sistemas detrás de eso. Como se ha hablado en este trabajo, el factor humano representa tener más probabilidad de errores y al tener demasiados administradores de sistema detrás de un clúster, la probabilidad aumenta.

En la introducción se habló un poco de las herramientas especializadas en la automatización, las cuales son necesarias para poder administrar las configuraciones en todas las máquinas. Por otro lado, cuando se tiene un elemento automatizado como una pieza de software que monitorea y configura sistemas, también genera información útil, con el fin de generar reportes que posteriormente serán analizados por el administrador. La información regularmente es acerca de lo que la herramienta hizo, detectó o reemplazó. Sin embargo también se genera información que no es relevante. En gran parte la administración de la configuración, se encarga de poder configurar tanto hardware como software para poder minimizar la generación de esa información innecesaria.

Capítulo III

Problemática

Actualmente la Coordinación de Supercómputo utiliza Nagios para realizar el monitoreo de los nodos de cálculo numérico intensivo, sin embargo, ésta herramienta no es óptima para las necesidades de la coordinación, porque genera demasiada información innecesaria que no es relevante para los administradores, y la información real e importante se oculta entre el resto.

Por este motivo, la Coordinación de Supercómputo y, en específico, el asesor de éste proyecto, hizo la propuesta sobre la creación de un clúster prototipo para la investigación, análisis, instalación y configuración de la herramienta CFEngine, así como la comparación con la herramienta actual de monitoreo (Nagios).

Construcción del clúster

Debido a que la infraestructura de la Coordinación de supercómputo está en constante operación, se planeó hacer un clúster prototipo para poder realizar el análisis comparativo, sin embargo, actualmente se están realizando pruebas de rendimiento y configuración con CFEngine en la supercomputadora Miztli para implementarlo completamente en un futuro.

Hardware

Servidores

Se utilizaron 47 nodos HP ProLiant DL 145 G2 con las siguientes especificaciones:

Ilustración 7: Nodos HP ProLiant DL 145 G2

- **Procesador:**
  - AMD 8132 Chipset Dual Core AMD Opteron(tm) Processor 285 de 2.61 Ghz..
- **Memoria:**
  - 8 GB de PC3200 DDR SDRAM a 400 Mhz.
- **Almacenamiento:**
  - Controlador integrado SATA
  - Disco duro Maxtor 6L16OMO de 160 GB a 1.5Gb/s y 7200 RPM. Tamaño de buffer 8 MB.
- **Red:**
  - Tarjeta integrada Ethernet Dual Broadcom 5721 10/100/1000 NICs (2 interfaces de red).
  - Adaptador Infiniband PCI-X y PCI-Express.
  - Adaptador propietario de HP para funcionamiento iLO.
- **Puertos USB:**
  - Cuatro puertos USB (dos delante y dos atrás).
- **Tarjeta gráfica:**
  - Tarjeta integrada NVIDIA GeForce2 GPU con 16MB DDR de memoria de video.

Switches

Se utilizaron 2 switches Ethernet HP Procurve 2650 de 48 puertos RJ45

Ilustración 8: Switch HP Procurve 2650

- **Entrada/salida**
  - 48 puertos 10/100 (IEEE 802.3 Tipo 10Base-T, IEEE 802.3u Tipo 100Base-TX); 1 Puerto de consola RS-232C DB-9; 2 puertos extra – Cada uno puede utilizarse ya sea como un puerto RJ-45 10/100/1000 (IEEE 802.3 Tipo 10Base-T; IEEE 802.3u Tipo 100Base-TX; IEEE 802.3ab 1000Base-T Gigabit Ethernet) O un slot abierto mini-GBIC.

- **Procesador**
  - Motorola Power PC MPC8245, 266 Mhz

- **Memoria**
  - Flash: 8 MB; SDRAM: 32 MB

- **Capacidad de ruteo**
  - 13.6 Gbps

Conectividad

El clúster se interconectó con tres redes distintas:

1. Red local de administración Ethernet.
2. Red de consolas iLO.
3. Red de alta velocidad Infiniband.

La red de administración Ethernet, funciona primordialmente para el uso común de los nodos dentro del clúster, ejecución de comandos de administración y transmisión de datos normales.

Ésta se conectó por medio de la interfaz número uno (`eth0`) de cada uno de los nodos, hacia el switch en una topología de red tipo estrella. El único nodo que no utilizó esa interfaz, fue el nodo maestro. La interfaz de la red local para el nodo maestro fue la segunda (`eth1`) y la interfaz número uno, se utilizó para la red WAN (Internet).

La red iLO de consolas, se utilizó para la funcionalidad iLO de los nodos, como lo es el conocer el estado del nodo, temperatura, estado de los ventiladores, etc.

De igual manera, la red iLO se conectó por el puerto específico de iLO hacia un switch Ethernet.

Por último, la red de alta velocidad Infiniband, se utilizó principalmente para la comunicación entre procesos. Esto es porque, como se mencionó anteriormente, algunos procesos, hacen uso del paralelismo y por lo tanto, necesitan varios nodos ejecutando un mismo proceso.

Los nodos tienen un puerto especial Infiniband y se conectaron por medio de éste al switch Infiniband.

A continuación se presentan los diagramas de red.

Las direcciones de red asignadas se encuentran en la sección Anexos.

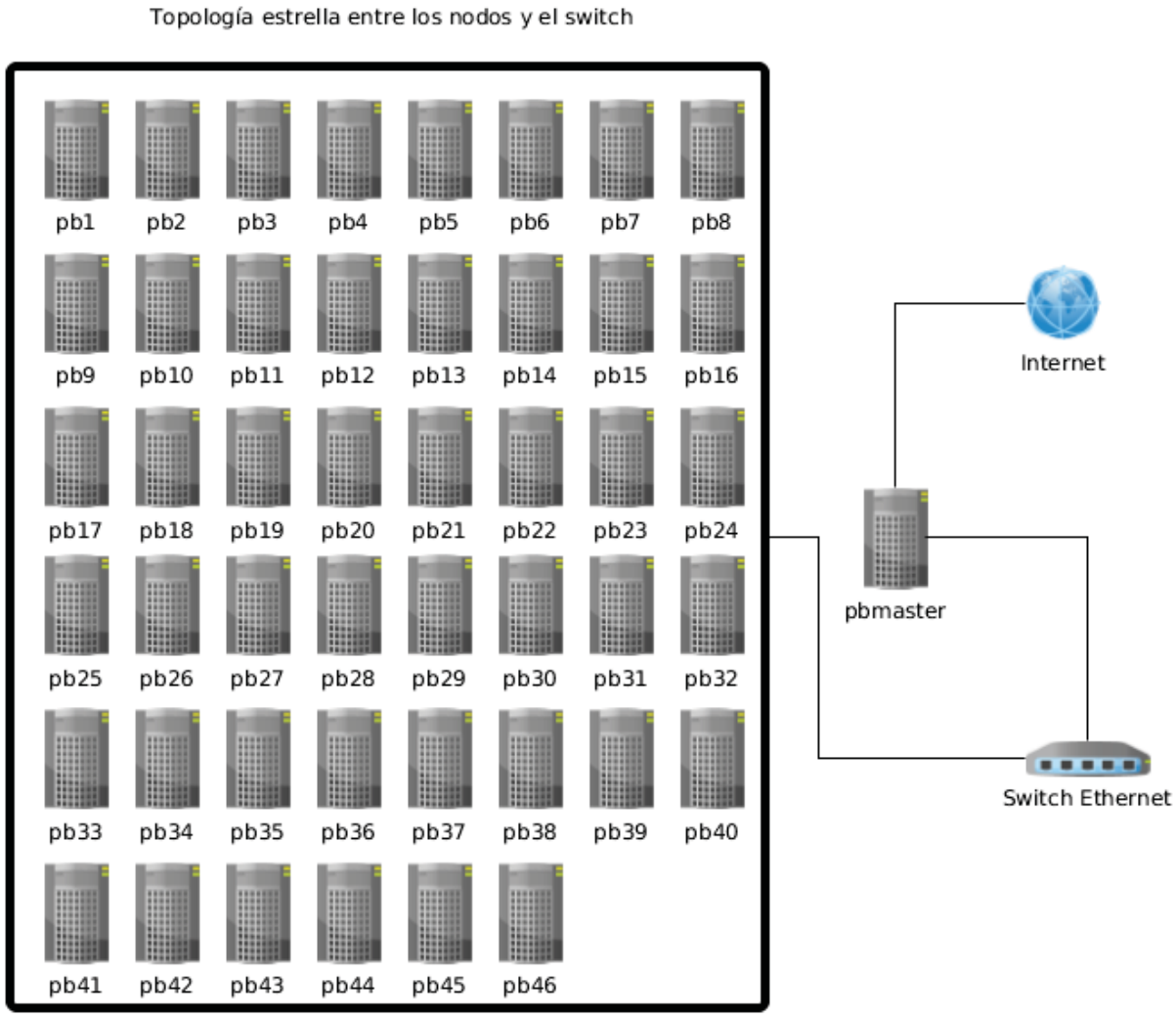


Ilustración 9: Red Ethernet.

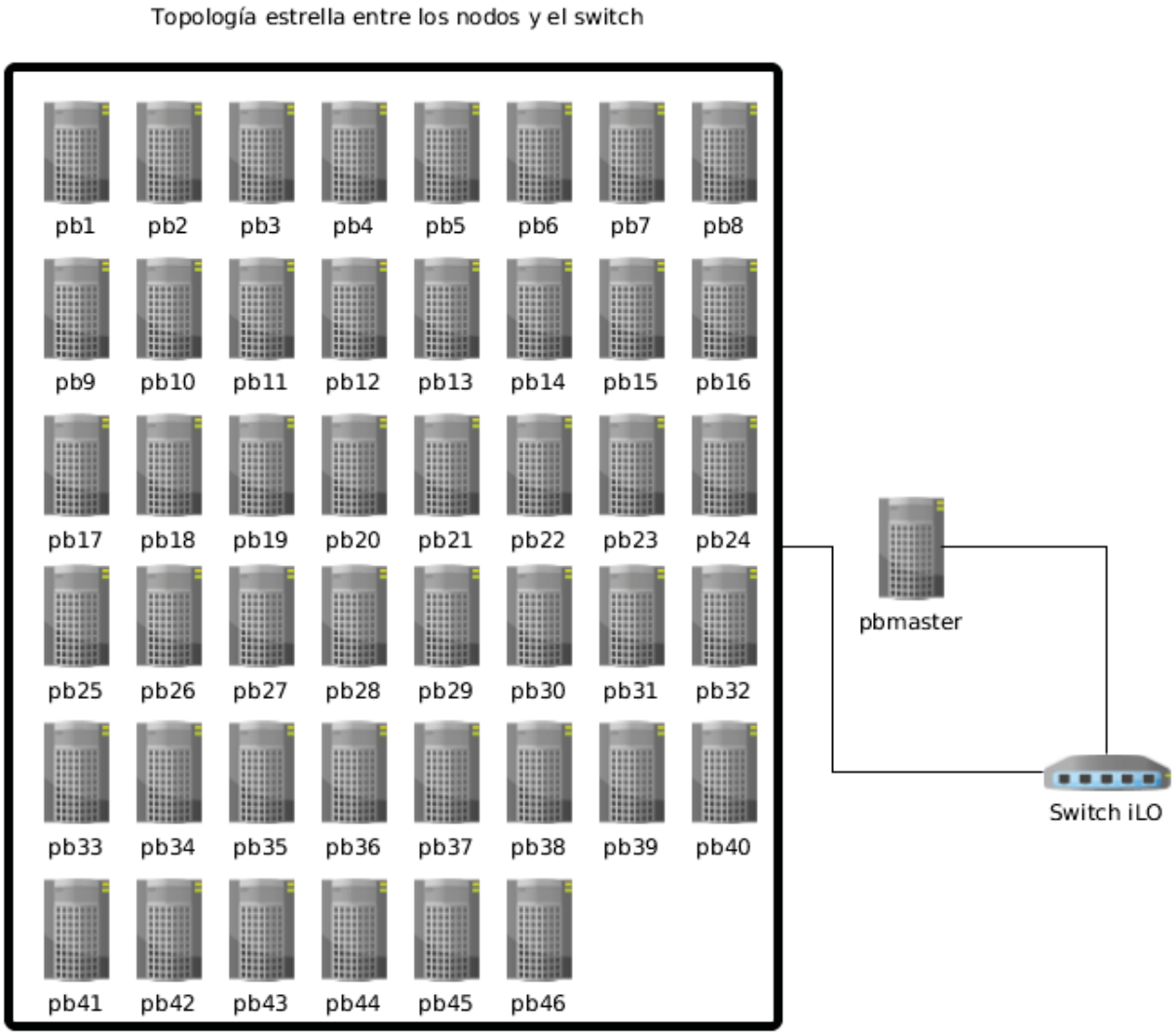


Ilustración 10: Red iLO.

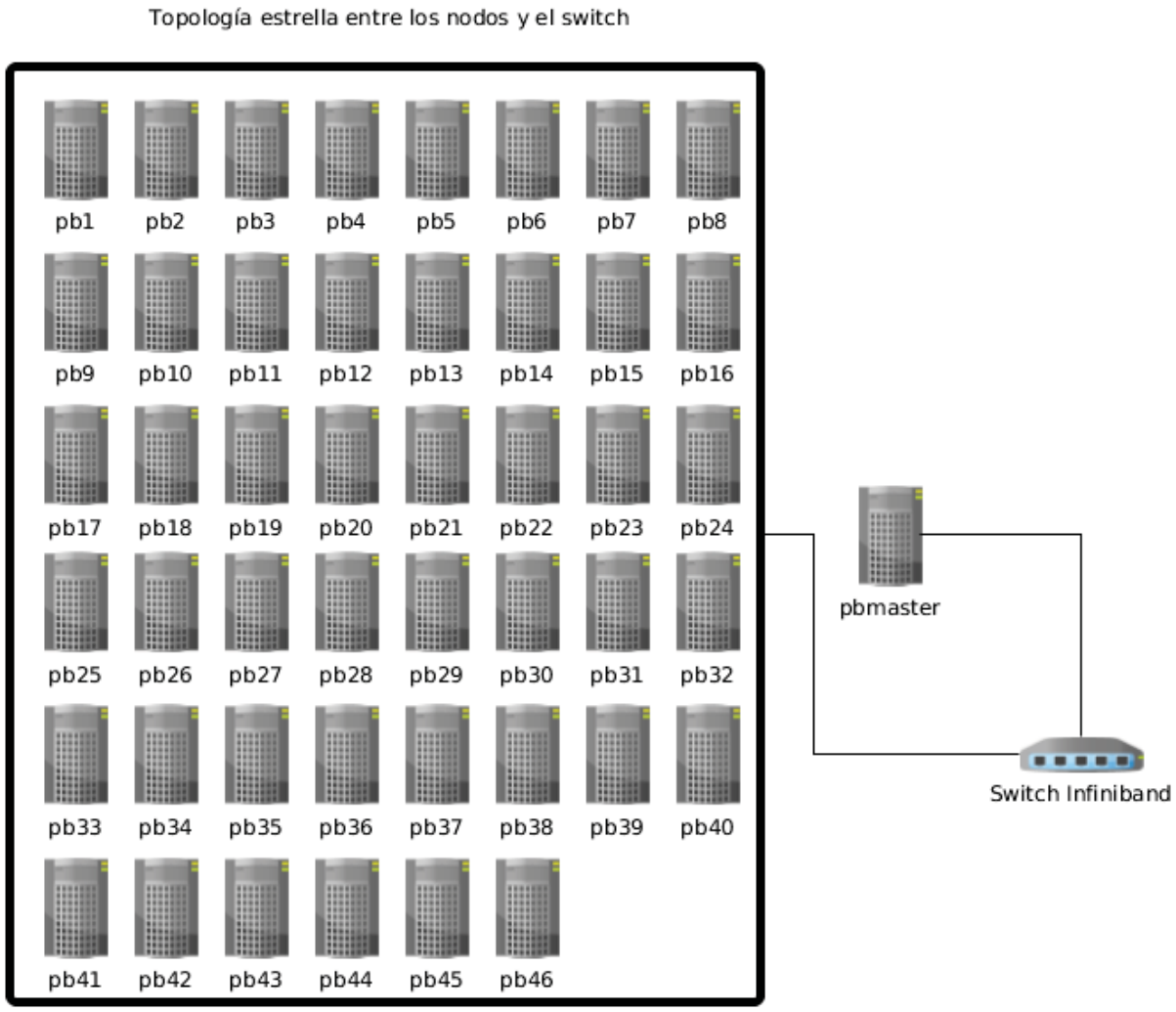


Ilustración 11: Red Infiniband.

Software

Sistema operativo

Se instaló en todos los nodos el sistema operativo GNU/Linux distribución CentOS 6.5.

Esto es debido a la similitud con el sistema operativo que se utiliza actualmente en la supercomputadora Miztli.

Procedimiento de la instalación del sistema operativo:

- Se definió un nodo para que actuara como el nodo maestro, `pbmaster`.
- Se instaló el sistema operativo de forma manual en el nodo maestro.
- A partir de esa instalación, se utilizó la herramienta *Kickstart* para la instalación de los demás nodos por medio de la red local.

Instalación del sistema operativo utilizando Kickstart (PXE Boot)

El procedimiento para la instalación de los demás nodos fue el siguiente:

1. Se instalaron los paquetes:
    - `tftp`
    - `dhcp`
    - `nfs`
    - `syslinux`
2. Se configuró el archivo del servidor DCHP: `/etc/dhcp/dhcpd.conf`
3. Se creó un directorio llamado `/share` ; fue utilizado como directorio contenedor. Dicho directorio contiene `/share/images/` ; en donde se copiaron las imágenes del sistema operativo CentOS 6.5; y `/share/kickstart/` en donde se creó el archivo de configuración kickstart.
   Nota: El directorio se compartió por medio de NFS.
4. También se creó el directorio `/tftpboot` ; fue utilizado para almacenar el archivo de boot y la imagen inicial del kernel de Linux, así como la configuración para indicar dónde se encuentra la imagen del sistema operativo (`initrd.img`, `pxelinux.0`, `pxelinux.cfg/default` y `vmlinuz`).

5. Se modificó el archivo: `/etc/xinetd.d/tftp`
   Para habilitar el servicio `tftp` y para indicar el directorio para el servicio `tftp` (`/tftpboot`).

6. Se exportó el directorio `/share/` vía NFS, modificando el archivo `/etc/exports`.

7. Se creó el archivo de kickstart descrito a continuación:

```
Text
Install
nfs --server 192.168.1.47 --dir /share/images/CentOS/
lang es_ES.UTF-8
keyboard us

network --onboot yes --device eth1 --bootproto dhcp --noipv6
rootpw  --iscrypted
$6$Ki9O73/IMANMkEqU$B3io/hhNgi8udnwyiWG0GqsdfaBz/DeHeJ17o6sy2RJ1Ykufw8x9UT0yMr1JFWFzSaGq6h6UKGjO
vYMoCGA3sOYH0
firewall --enabled --port=22:tcp
authconfig --enableshadow --passalgo=sha512
selinux –disabled
timezone --utc America/Mexico_City

bootloader --location=mbr --driveorder=sda
clearpart --all
zerombr

part / --fstype=ext4 --asprimary --size=49336
part /appl --fstype=ext4 --asprimary --size=49333
part swap --asprimary --size=16383
part /home --fstype=ext4 --grow --asprimary --size=200

Halt

%packages
@base
@core
@infiniband
qperf
perftest
infiniband-diags
opensm
gcc
tokyocabinet-devel
pcre-devel
openssl-devel
postgresql-devel
libvirt-devel
libacl-devel
%end

%post
#!/bin/bash
/bin/echo -e 'DEVICE=eth1\nIPADDR='`ifconfig eth1|egrep -o '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.
[0-9]{1,3}' |head -1`'\nNETMASK=255.255.255.0' > /etc/sysconfig/network-scripts/ifcfg-eth1

/sbin/service iptables start
/sbin/iptables -F
/sbin/iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
```

```
/sbin/iptables -A INPUT -s 192.168.1.0/24 -j ACCEPT
/sbin/iptables -A INPUT -s 127.0.0.1/32 -j ACCEPT
/sbin/iptables -A INPUT -j DROP
/sbin/service iptables save

/usr/bin/wget -O - http://192.168.1.47/cfengine-3.5.3.tar.gz | tar xzvf -
cd cfengine-3.5.3
./configure --prefix=/appl/cfengine
/usr/bin/make
/usr/bin/make install
/appl/cfengine/bin/cf-key
/bin/cp -Rp /appl/cfengine/share/CoreBase/* /appl/cfengine/var/cfengine/masterfiles
/bin/cp  /appl/cfengine/bin/* /appl/cfengine/var/cfengine/bin/
/appl/cfengine/bin/cf-agent --bootstrap 192.168.1.47
/appl/cfengine/var/cfengine/bin/cf-agent -f /appl/cfengine/var/cfengine/inputs/update.cf
/appl/cfengine/var/cfengine/bin/cf-agent -K -I

/sbin/service sshd start
/sbin/chkconfig sshd on
/bin/echo -e '#!/bin/bash\n# chkconfig: 345 99 00\n# description: Inicio de CFEngine\ncase $1
in\n      start)\n                /appl/cfengine/var/cfengine/bin/cf-execd\n      ;;\n
stop)\n         killall cf-execd\n      ;;\n    *)\n            echo "Use start o
stop"\n       ;;\nesac' > /etc/init.d/cfengined ; /bin/chmod 755 /etc/init.d/cfengined;
/sbin/chkconfig cfengined on
/bin/echo -e "NETWORKING=yes\nNETWORKING_IPV6=no\nHOSTNAME=pb"`ifconfig eth1|egrep -o '[0-9]
{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}'|head -1|cut -f4 -d.`"\nNTPSERVERARGS=iburst" >
/etc/sysconfig/network
```

Como parte de la funcionalidad del clúster, también se instalaron algunas utilerías y paquetes como:

- OpenIPMI.
- OpenMPI.
- Torque.

Nagios

Nagios es un sistema de monitoreo de software y hardware tanto de sistemas computacionales como redes informáticas. Tiene capacidades para poder identificar el comportamiento esperado de un sistema y de reportar cuando éste comportamiento cambia o parece no ser el adecuado.

Principalmente se utiliza para obtener información sobre el sistema, como la carga del procesador, uso de disco, memoria, temperatura, entre otras. También funciona para monitorear los servicios de red como lo son SSH y HTTP, intentando identificar un

comportamiento normal, de manera que, si el comportamiento cambia, reporte alguna incidencia.

En la Coordinación de Supercómputo se utiliza Nagios para el monitoreo de los sistemas, sobre todo para la carga del sistema, pero uno de los problemas principales que tiene Nagios, es que (a pesar de poderse configurar muy a fondo) su funcionamiento reside en el aprendizaje de lo que considera lo “normal” en un sistema. Por lo tanto, en una infraestructura de cálculo numérico intensivo, el comportamiento normal difiere mucho a lo que Nagios considera normal. Debido a esto, el sistema de monitoreo reporta muchas cosas que en realidad no son peligrosas ni mucho menos anormales.

Nagios depende también de un servidor web (en este caso apache), de una base de datos (MySQL) y de un lenguaje interpretado (PHP) para su funcionamiento y para desplegar los datos si se desean consultar.

El funcionamiento de Nagios consta de un servidor el cual es el que realizará el monitoreo, tanto el propio, como de todos los clientes que se le indiquen. El servidor es el que dará aviso al administrador en caso de que algún aspecto del sistema necesite atención.

Instalación

La instalación de Nagios requiere tener un entorno LAMP, esto es, un entorno con Apache, MySQL y PHP instalado en el sistema.

- Se procedió a instalar el entorno LAMP con el gestor de paquetes `yum`.

```
# yum -y install httpd httpd-devel mysql mysql-server mysql-devel php php-mysql php-common php-gd php-mbstring php-mcrypt php-devel php-xml
```

- También se procedió a instalar los repositorios EPEL (*Extra Packages for Enterprise Linux*) por medio del gestor de paquetes, ya que el binario compilado de Nagios se encuentra en ellos.

```
# yum -y install epel-release
```

- Posteriormente, se instalaron todos los paquetes de Nagios.

```
# yum -y install nagios*
```

Para efectos de este proyecto, se configuró Nagios de la siguiente forma:

- El nodo maestro del clúster actuó como el servidor de Nagios.
- Los otros 46 nodos fueron clientes.

Configuración en el servidor

- Se editó el archivo `/etc/nagios/objects/contacts.cfg` que es en donde se especifica la información de contacto al administrador. Ahí, se cambió la opción `email`, indicando cuál fue el correo electrónico del administrador al que dará aviso:

```
email                           administrador_del_sistema@super.unam.mx;
```

Al instalar Nagios, se creó automáticamente un archivo con la descripción de las reglas y accesos que debe tener apache para Nagios en `/etc/httpd/conf.d/nagios.conf`

- Se editó la siguiente directiva dentro del archivo para permitir las conexiones de la red local del clúster:

```
<Directory "/usr/lib64/nagios/cgi-bin/">
#  SSLRequireSSL
   Options ExecCGI
   AllowOverride None
   Order allow,deny
   Allow from all
#  Order deny,allow
#  Deny from all
   Allow from 127.0.0.1 192.168.1.0/24
   AuthName "Nagios Access"
   AuthType Basic
```

```
   AuthUserFile /etc/nagios/passwd
   Require valid-user
</Directory>
```

- Después, se asignó una contraseña para la administración de Nagios con el comando `htpasswd` de apache:

```
# htpasswd /etc/nagios/passwd nagiosadmin
```

- Por último, se agregó Nagios y apache a los *scripts* de inicio de sistema y se iniciaron los servicios:

```
# service nagios start
# service httpd start
# chkconfig nagios on
# chkconfig httpd on
```

Después de la instalación de Nagios, se procedió a agregar los clientes o *hosts* de los cuales hará el monitoreo.

Para esto, en cada cliente se hizo lo siguiente:

- Se editó el archivo `/etc/nagios/nrpe.cfg` y se agregó la dirección del servidor de Nagios.

```
allowed_hosts=127.0.0.1 192.168.1.47
```

- Se inició el servicio nrpe.

```
#service nrpe start
#chkconfig nrpe on
```

Y del lado del servidor se agregaron cada uno de los clientes al directorio `/etc/nagios/servers`.

- Se creó el directorio.

```
# mkdir /etc/nagios/servers
```

- Se creó y editó el archivo `/etc/nagios/servers/clients.cfg` con el contenido siguiente:
- Nota: Se define en el trabajo sólo para un nodo, pero la directiva `define host` se estableció para cada uno de los nodos.

```
define host{
use                             linux-server
host_name                       pb1
alias                           nodo1
address                         192.168.1.1
max_check_attempts              5
check_period                    24x7
notification_interval           30
notification_period             24x7
}
```

Para finalizar la configuración, simplemente se reinició el servicio `nagios`.

```
# service nagios restart
```

Para poder conocer un aspecto de monitoreo en particular, se necesita ingresar a la interfaz web de Nagios.

A continuación se presentan imágenes que representan la interfaz web del monitoreo con Nagios:

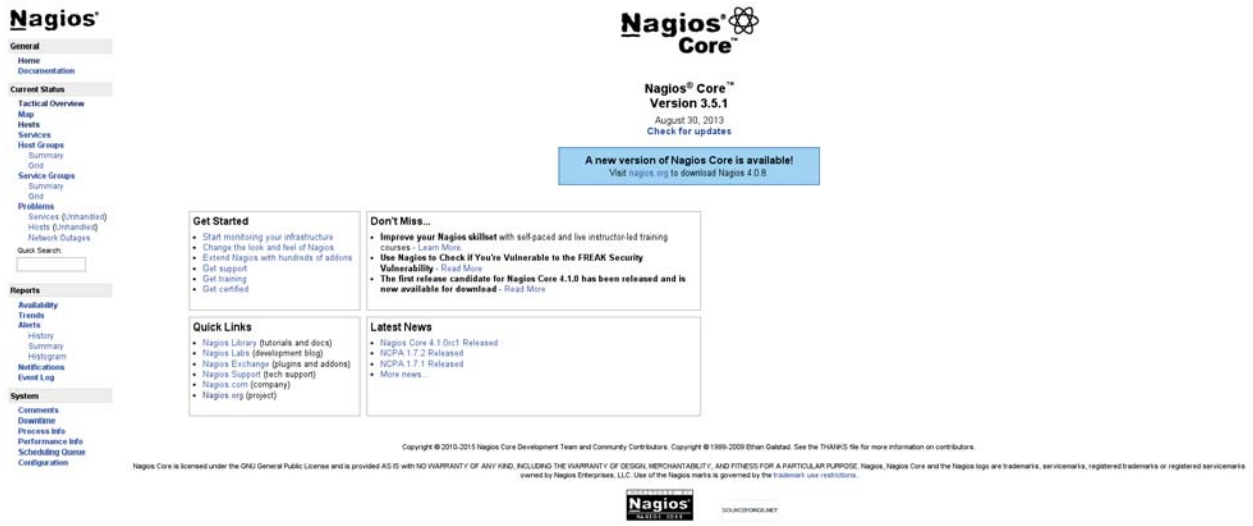


Ilustración 12: Interfaz web de Nagios, también llamada "Consola de administración".

Nagios agrupa los nodos que monitorea en la sección *hosts.*

Se puede conocer el estado de los nodos y el estado general del clúster, además, muestra si hay aspectos que deben atenderse o si son críticos.

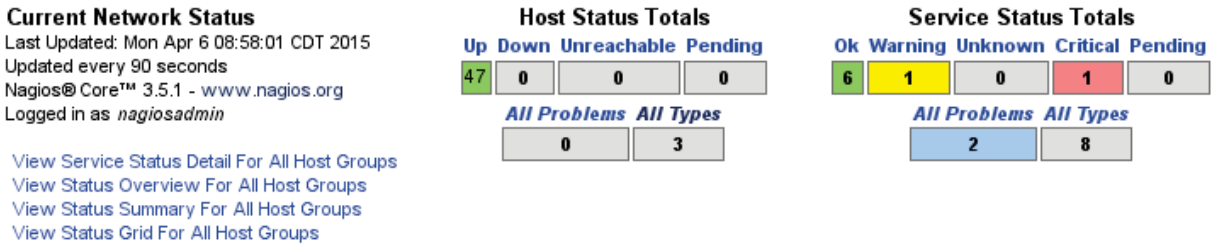


Ilustración 13: Estado general del clúster y nodos.

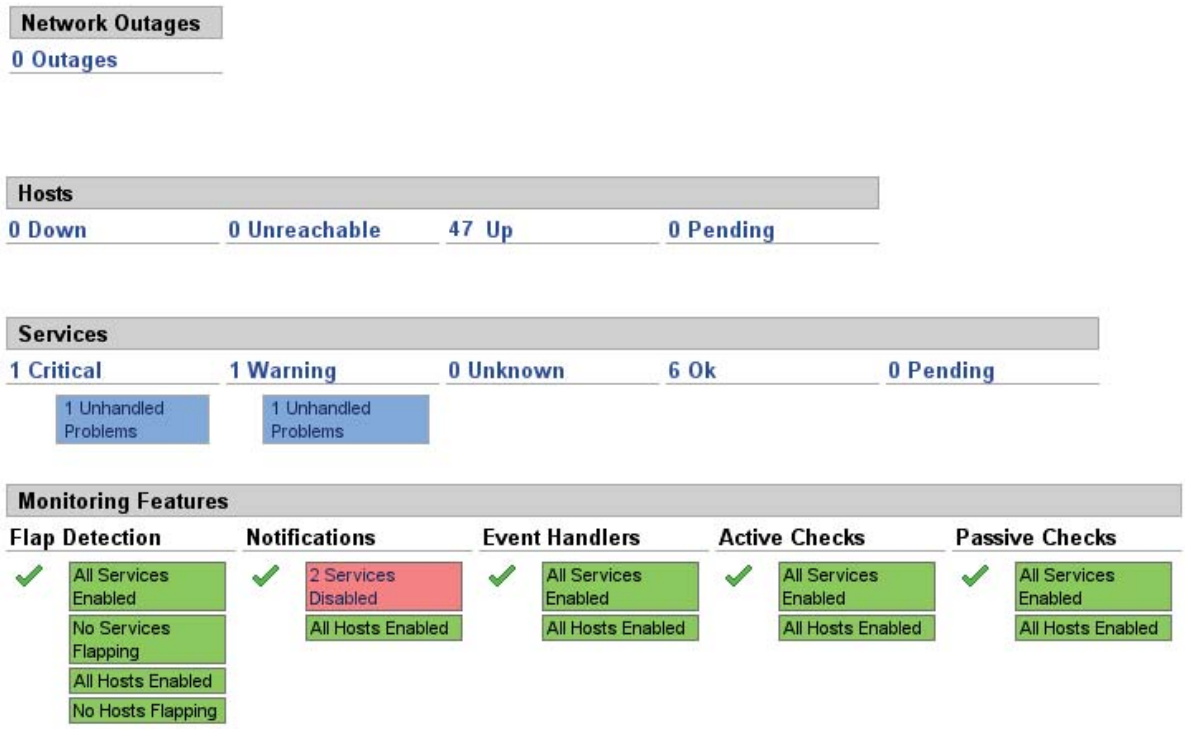


Ilustración 14: Advertencias de problemas en el clúster.

CFEngine

CFEngine es un sistema de administración, configuración, auditoría y automatización para sistemas de cualquier tamaño. La tarea principal de CFEngine es mantener a los sistemas lo más cercano posible al estado deseado, utilizando sus capacidades de monitoreo para poder tomar decisiones y realizar cambios a la configuración del sistema.

Dentro de las operaciones básicas que CFEngine puede realizar, se encuentran las siguientes:

- Extraer información del estado y configuración del sistema.
- Comprobar y manipular los permisos y propietarios de los archivos.
- Comprobar la existencia de los procesos que se están ejecutando.
- Comprobar la existencia de usuarios en el sistema.

- Ejecutar programas y comprobar su salida.
- Comprobar y manipular los paquetes instalados en el sistema.
- Comprobar y manipular los servicios en el sistema.

En la introducción se habló sobre el estado deseado, explicando que se refiere al estado ideal que debería tener el sistema de acuerdo a las configuraciones deseadas y al comportamiento esperado. CFEngine utiliza el estado deseado como un principio básico para su funcionamiento, pero también se basa en otros dos principios:

- Teoría de promesas.
- Configuración convergente.

Teoría de promesas

La teoría modela el comportamiento de agentes autónomos en un entorno sin una autoridad central, basado sólo en promesas del comportamiento que hace cada agente y demuestra que aún sin un control central, el sistema puede ser convergente a un estado estable.[2]

Básicamente una promesa es una declaración del estado que se supone debe tener una entidad en el sistema. Una entidad puede ser un archivo, una variable de entorno, un paquete de software, un comando, etc. De acuerdo a la teoría de promesas, CFEngine evaluará lo que se promete y dependiendo si esa promesa se cumple o no, se harán los cambios necesarios para que la promesa se cumpla.

Una entidad, también se le conoce como el *prometedor,* debido a que promete realizar alguna acción en el sistema o contener algún valor. A continuación se presentan ejemplos de los prometedores y las promesas asociadas a ellos.

2 Zamboni, D.. (2012). *Learning CFEngine 3*. United States of America: O'Reilly. p. 27

Prometedor	Promesa
Un archivo	Promete existir y tener un contenido específico
Un proceso	Promete estar ejecutándose
Una cuenta de usuario	Promete existir en el sistema y tener características específicas (directorio hogar, tipo de intérprete de órdenes, etc.)

Si por algún motivo el prometedor no cumple con lo que promete, CFEngine toma las acciones necesarias para corregir esa promesa y hacer que se cumpla, basándose en las políticas establecidas u otras promesas relacionadas.

Configuración convergente

CFEngine realiza una configuración convergente, esto se refiere a que los cambios que se hacen en el sistema para poder llegar al estado deseado, no necesariamente tienen que realizarse en la primera ejecución. Los cambios en las configuraciones se realizan incrementalmente, sin importar los cambios que han aparecido en el sistema.

De esta manera CFEngine se acerca al estado deseado, haciendo los cambios necesarios en el momento en el que puede realizarlos. Esto significa que CFEngine no siempre puede hacer cambios en el sistema, aunque lo intente. Se describe el siguiente ejemplo para demostrar ésto:

- Se pretende instalar el paquete `pdsh` (para la ejecución en paralelo de comandos en el clúster).
- El paquete pertenece al repositorio EPEL, pero aún no está configurado.
- Se le indica a CFEngine que el paquete `pdsh` debe estar instalado y que el repositorio EPEL debe estar configurado.

Al ejecutarse, CFEngine intentará instalar el paquete en la primera ejecución, debido a que el repositorio no está configurado, no lo podrá instalar. Tiempo después, en la misma ejecución, configurará el repositorio.

En la segunda ejecución CFEngine instalará el paquete `pdsh` y después se asegurará que el repositorio esté correctamente configurado.

Componentes

Los componentes de CFEngine permiten operaciones como monitoreo o configuración de acuerdo a las promesas que se escribieron dentro de una computadora. También cuenta con capacidades de servidor, para poder distribuir esas configuraciones entre las computadoras asociadas, ya sea en un clúster o no.

Los componentes son:

- **`cf-agent`**
  - Es el programa que evalúa las promesas y actúa de acuerdo a lo que éstas le indican.
  - Puede ejecutar archivos ".cf" individuales con la opción "–f", si no se le indica un archivo, ejecuta `promises.cf` que es el archivo predeterminado que contendrá las promesas de configuración.
  - Regularmente se invoca directamente, ya sea para actualizar la configuración del sistema, o para ejecutar alguna política individualmente.
  - También es ejecutado por `cf-execd` y por `cf-serverd`.
    - `cf-execd` lo ejecuta cada 5 minutos (configuración predeterminada).
    - `cf-serverd` lo ejecuta cuando el servidor de políticas hace una petición de ejecución.
- **`cf-execd`**
  - Se encarga de ejecutar `cf-agent` cada cinco minutos (comportamiento predeterminado).
  - Recolecta la información de salida que regresa `cf-agent`.
  - Éste es el demonio que se debe de ejecutar al inicio del sistema, ya sea como un *script* de inicio, o con `cron`.

- Cuando ejecuta `cf-agent`, éste último se encarga de la configuración, asegurándose que los otros componentes (como `cf-monitord`) se estén ejecutando como se espera.

- **`cf-key`**
  - Crea un par de llaves de cifrado para el host, que se usan para la autenticación cuando se comunica con el servidor de promesas.
- **`cf-serverd`**
  - Se encarga de escuchar peticiones de los clientes para poder distribuir archivos de configuración, que posteriormente ellos ejecutarán localmente.
  - También escucha peticiones de `cf-runagent`, el cual solicita remotamente la ejecución de `cf-agent` por parte del servidor.
  - Utiliza el puerto TCP/5308.
- **`cf-runagent`**
  - Solicita que la máquina cliente, ejecute `cf-agent` para actualizar los archivos de configuración que hay en el servidor (más específicamente en `/directorioDeTrabajoDe/cfengine/masterfiles`).
- **`cf-promises`**
  - Se encarga de comprobar la sintaxis de las políticas y archivos ".cf".
  - Sin argumentos, comprueba la sintaxis de `promises.cf`.
- **`cf-monitord`**
  - Recolecta información del sistema.
  - Lo almacena en la variable especial `mon`, que está disponible para `cf-agent`.
  - Algunos ejemplos de qué es lo que monitorea son:
    - Usuarios con procesos activos en el sistema: `mon.value_users`
    - Espacio libre en la partición: `mon.value_diskfree`

La ilustración 15 muestra el funcionamiento de los componentes de CFEngine entre un cliente y servidor.

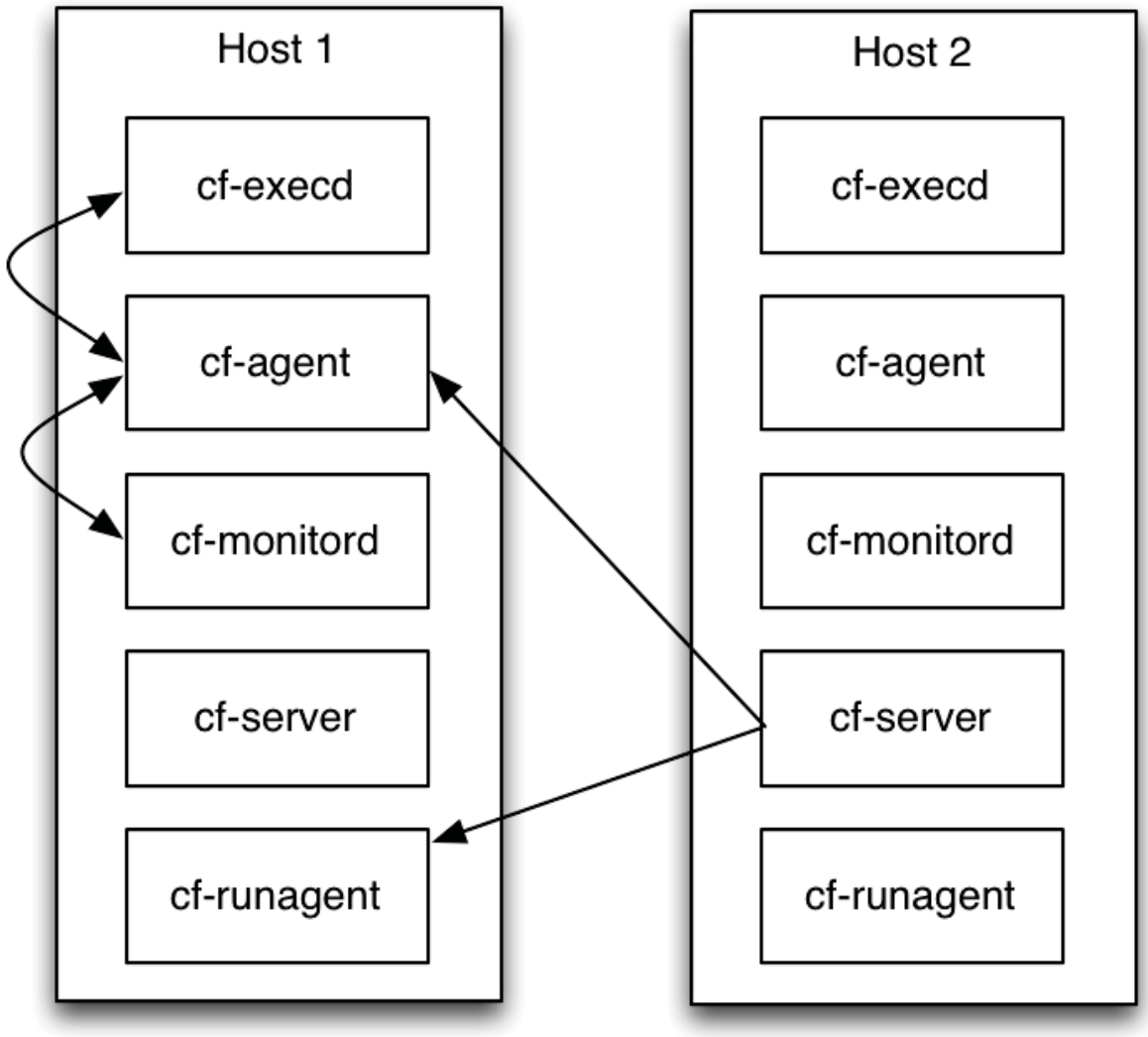


Ilustración 15: Componentes de CFEngine

Instalación

La instalación se realizó a partir del código fuente, por lo que se tuvo que realizar la compilación con las opciones necesarias para el clúster.

Procedimiento de la instalación:

- Se descargó el paquete con el código fuente de la versión 3.5.3 desde `http://cfengine.com/source-code/`
- Se descomprimió y desempaquetó el archivo.

```
# tar xzvf cfengine-3.5.3.tar.gz
```

- Se ingresó al directorio que se extrajo.

```
# cd cfengine-3.5.3
```

- Se preparó la compilación con las opciones específicas del clúster para después compilarlo e instalarlo.

```
# ./configure --prefix=/appl/cfengine --with-tokyocabinet --with-openssl –-with-pcre

# make

# make install
```

- Al tener instalado CFEngine, se necesitó crear sus llaves de cifrado para la comunicación entre el cliente y el nodo maestro.

```
# /appl/cfengine/bin/cf-key
```

- Al ser un paquete con el código fuente, los archivos de configuración predeterminados para el servidor se instalan en un directorio donde se encuentra la documentación de todo CFEngine, por lo que hay que copiarlos dentro del directorio de políticas del servidor.

```
# cp -Rp /appl/cfengine/share/CoreBase/* /appl/cfengine/var/cfengine/masterfiles
```

- CFEngine necesita saber qué nodo funcionará como servidor de políticas, es decir, qué nodo va a proporcionar las instrucciones para poder monitorear y configurar cada uno de los nodos. En el clúster, el nodo que funcionará como servidor de políticas es el nodo maestro, con la dirección IP 192.168.1.47

```
# cf-agent --bootstrap 192.168.1.47
```

El procedimiento de instalación se realizó en cada uno de los nodos, difiriendo en que sólo se copian los archivos en el servidor y no en los clientes.

Al ser 46 nodos además del nodo maestro, la instalación de CFEngine se realizó por medio de la herramienta Kickstart de forma automatizada justo después de que se instalara el sistema operativo.

Monitoreo

El componente `cf-monitord` (que es ejecutado por `cf-execd`) recolecta la información del sistema continuamente, sin embargo, lo único que hace es almacenar toda esa información en la variable especial `mon`, la cual podrá ser accesada por el componente que realiza las operaciones en el sistema (`cf-agent`).

Por ejemplo, si se quiere saber el espacio disponible de almacenamiento en disco, se necesitaría crear una promesa como la siguiente:

```
body common control

{

        bundlesequence => {"monitoreo"};

}

bundle agent monitoreo

{

        reports:

                "Hay $(mon.value_diskfree) GB de espacio libre en disco";

}
```

El monitoreo de CFEngine puede realizarse utilizando los diversos valores de la variable especial `mon`. A partir de obtener dicha variable, se pueden tomar decisiones si es que algún aspecto del sistema llegó a un valor crítico, para entonces enviar una advertencia por correo o incluso intentar repararla de acuerdo a alguna promesa que se haya escrito.

Tomando el ejemplo anterior, se realizó una promesa que envíe una alerta si el espacio libre en disco es menor a 10%:

```
body common control

{

        bundlesequence => {"monitoreo"};

}

bundle agent monitoreo

{

        classes:

                "disco" expression => islessthan("$(mon.value_diskfree)","10");

        reports:

                disco::

                        "¡¡Hay menos de 10% de espacio en disco!!";

}
```

En el tipo de promesa `classes`, se indicó que se cree una clase cuando el espacio en disco sea menor a 10%. En el tipo de promesa `reports`, se indica que si la clase `disco` existe, envíe un reporte con el texto indicado.

Siguiendo esa misma promesa de monitoreo, se escribieron dentro de la misma otras promesas sobre la carga del sistema y sobre la temperatura de el procesador. Al final el archivo de políticas se escribió así:

```
body common control

{

        bundlesequence => {"monitoreo"};

}
```

```
bundle agent monitoreo

{

        classes:

                "disco" expression => islessthan("$(mon.value_diskfree)","10");

                "temperatura1" expression => isgreaterthan("$(mon.value_temp0)","95");

                "carga" expression => islessthan("$(mon.value_cpu)","90");

        reports:

                disco::

                      "¡¡Hay menos de 10% de espacio en disco!!";

                temperatura1::

                      "¡¡La temperatura supera los 95°!!";

                carga::

                      "¡¡La carga del CPU está por debajo del 90%!!";

}
```

En este caso, se configuró para que CFEngine envíe un correo al administrador cada que esas condiciones sean correctas. Se indica el correo electrónico en el archivo `/appl/cfengine/var/cfengine/masterfiles/controls/cf_execd.cf`.

Para monitorear la temperatura es necesario que el paquete `lm_sensors` esté instalado.

Administración del sistema

Como se ha mencionado anteriormente, CFEngine también tiene la capacidad de realizar cambios en las configuraciones del sistema a partir de las promesas que se escriban. En éste proyecto también se aplicaron algunas configuraciones de sistema para realizar pruebas e implementarse posteriormente en el clúster de supercómputo Miztli.

Se aplicaron las configuraciones de distribución de *passwords* en todos los nodos, rotación de bitácoras para evitar que un sistema de archivos se sature y otras configuraciones.

Distribución de passwords

Cuando un usuario cambia su *password* con el comando `passwd`, sólo se cambia en el nodo en el que se encuentra trabajando, por lo cual se debe replicar el *password* en todos los nodos.

Este problema se resolvió de la siguiente manera utilizando un *script*, un programa en C y CFEngine, los pasos fueron los siguientes:

- Se renombró el archivo `/usr/bin/passwd` para que el usuario no pueda utilizarlo directamente.
- Se escribió un programa en C para poder otorgar permisos de root al *script* gracias al permiso especial SUID. El archivo compilado se nombró `passwd` y se colocó en `/usr/bin/` .
- Se creó un *script* que cambia el *password* en el nodo maestro, sin importar el nodo en el que se encuentre trabajando el usuario.
- Se dio acceso a CFEngine para poder replicar los archivos `/etc/passwd` y `/etc/shadow` hacia los demás nodos una vez que se han actualizado.
- Se creó una política de promesas para la replicación de dichos archivos.

Script

```
#!/bin/bash

echo "Ingresa el password actual:"

stty -echo

read actual

#Verifica que sea el actual

stty echo

usuario=`grep "^.*:x:$1:" /etc/passwd |cut -f1 -d:`

salt=`grep "^$usuario" /etc/shadow|cut -f2 -d:|cut -f3 -d'$'`

hash=`grep "^$usuario" /etc/shadow|cut -f2 -d:|cut -f4 -d'$'`
```

```
if [ "`echo "$actual"|mkpasswd -m sha-512 -S "$salt" -s|cut -f4 -d'$'`" = "$hash" ]
then
        echo "Ingresa el nuevo password:"
        stty -echo
        read nuevo1
        stty echo
        echo "Confirma el nuevo password:"
        stty -echo
        read nuevo2
        stty echo
        if [ "$nuevo1" = "$nuevo2" ]
        then
                echo "Cambiando la contraseña en el servidor central..."
                cd /root
                      echo "$nuevo2"|ssh -oStrictHostKeyChecking=no 192.168.1.1 "passwd --stdin
$usuario"
                echo "El cambio se verá reflejado dentro de 5 a 10 minutos en todos los nodos"
                exit 0
        else
                     echo -e "Error: Las contraseñas no coinciden\Aviso: No se ha cambiado la
contraseña"
                exit 1
        fi
else
        echo -e "Error: Fallo de autenticación\nAviso: no se ha cambiado la contraseña"
        exit 1
fi
```

El *script* se guardó en el directorio `/usr/bin/` y se le otorgaron los permisos `r-x --- ---` para que sólo pueda ser ejecutado por el usuario root.

Programa en C

```
#include <stdlib.h>

#include <stdio.h>

int main()

{

   int uid,euid;

   char cmd[60]; //Cadena que contendrá el comando


   // Guarda el UID para utilizarlo después

   uid = getuid();


   sprintf(cmd,"/usr/bin/oculto %d",uid);


   // Asigna el EUID a ser UID en la ejecución

   setuid(geteuid());

   system(cmd);

   return 0;

}
```

Política de promesas de CFEngine

```
bundle agent usuarios_passwords

{

       files:

       !pbmaster::

              "/etc/passwd"

                     comment => "Asegura tener integridad en la tabla de usuarios",

                     perms => mog("644","root","root"),

                     copy_from => secure_cp("/etc/passwd","$(sys.policy_hub)");

              "/etc/shadow"
```

```
                        comment => "Asegura tener las contraseñas actualizadas",

                        perms => mog("640","root","root"),

                        copy_from => secure_cp("/etc/shadow","$(sys.policy_hub)");

}
```

Posteriormente se agregó a la política, utilizando la misma lógica, las líneas necesarias para que replicara `/etc/group`, `/etc/sudoers` y el directorio `/root/.ssh/`

De este modo, el *script* realiza el cambio de *password* en el nodo maestro, utilizando al usuario root. Después, CFEngine se encarga de distribuir la tabla de usuarios y de *passwords* a todos los nodos dentro de un periodo de 5 minutos.

Rotación de bitácoras

Algo muy importante de la administración de sistemas son las bitácoras, puesto que dan información sobre el comportamiento del mismo y de errores que perjudiquen el funcionamiento óptimo. Sin embargo, puede haber problemas en sistemas de alto rendimiento, ya que una bitácora puede crecer exponencialmente hasta consumir todo el espacio de almacenamiento, provocando que el sistema se comporte de formas inesperadas.

El sistema de bitácoras `Syslog`, cuenta con una rutina para la rotación de bitácoras generadas por el sistema mismo, pero no puede rotar las bitácoras que genere independientemente el administrador de sistemas o software externo.

Por medio de una política de CFEngine, se indica qué bitácoras deben rotarse con regularidad, ya sea por un intervalo de tiempo, o si superaron un tamaño predeterminado.

```
bundle agent rota_logs

{

       vars:

              "iptables" string => "/var/log/iptables.log";

              "messages" string => "/var/log/messages";

              "1M" int => "1M"; # 1M de límite

              "10M" int => "10M"; # 10M de límite

        files:

              #Por tamaño

              "$(iptables)"

                     rename => rotate("2"),

                     ifvarclass => isgreaterthan(filesize("$(iptables)"),"$(1M)");

              #Por tiempo

              "$(messages)"

                     rename => rotate("2"),

                     action => diario;

}

body action diario

{

       ifelapsed => "1440";

}
```

Capítulo IV

Resultados

Con base en los objetivos particulares de éste proyecto, se puede establecer que se obtuvieron los siguientes resultados:

- Se creó y configuró un clúster con 47 nodos de cálculo.
- Además de las herramientas para el cálculo numérico intensivo, se instalaron y configuraron las herramientas Nagios y CFEngine.
- Se analizó y comparó el desempeño de cada herramienta, para así conocer cuál de las dos es la que mejor cubre las necesidades de la coordinación.

Cabe destacar, que al inicio del proyecto, sólo se tenía conocimiento de CFEngine como herramienta de monitoreo, pero gracias a la investigación y análisis de la herramienta, se pudo observar que CFEngine no sólo realiza el monitoreo de un sistema, sino que también puede realizar configuración y administración automatizada e incluso puede tomar decisiones a partir del monitoreo que realiza constantemente.

Con el análisis y las pruebas realizadas en el clúster prototipo, se espera que se implemente CFEngine por completo en la supercomputadora Miztli. En el capítulo tercero, se mencionó que actualmente se realizan pruebas en algunos nodos para comprobar sus capacidades de administración automatizada y no sólo de monitoreo.

Hasta el momento se ha visto la administración automatizada utilizando CFEngine es muy confiable ya que permite poder configurar los nodos de cálculo o de almacenamiento de manera uniforme y sin que el rendimiento se vea afectado.

En el clúster prototipo, se obtuvo que el monitoreo se realizó sin la necesidad de disponer de una interfaz web gracias a CFEngine. Nagios reporta a un correo electrónico las anormalidades que puede encontrar, pero si se requiere consultar algún elemento de monitoreo en particular, se necesita de la interfaz web, a diferencia de CFEngine, que se le puede indicar que muestre o incluso reporte a alguna dirección de correo electrónico cualquier información de algún nodo en particular.

Se espera que el monitoreo y la administración centralizada y automatizada en la supercomputadora Miztli se implemente por medio de CFEngine utilizando como base éste trabajo y el clúster prototipo.

Al finalizar el proyecto, el clúster prototipo será utilizado con fines académicos para los programas y planes de becarios de la Coordinación de Supercómputo, para así introducir a los estudiantes al supercómputo y se pueda utilizar para sus proyectos finales.

Como se mencionó en la introducción y debido a la falta de información en el idioma español, parte de la aportación de éste trabajo escrito es tener documentación sobre la herramienta CFEngine. Como prueba de esto, recientemente se pudo desarrollar e impartir un taller sobre la administración centralizada y el monitoreo de supercómputo con CFEngine en el 6° Congreso Internacional de Supercómputo en México ISUM 2015.

Conclusiones

Al final, se puede concluir que la herramienta CFEngine realiza el monitoreo de manera más precisa y que es de más utilidad para las necesidades de la Coordinación de Supercómputo.

La administración automatizada de la infraestructura de supercómputo se ve beneficiada por el modo de funcionamiento de CFEngine, por su administración centralizada y su teoría de promesas, en la cual, cada nodo promete tener alguna configuración en particular a diferencia de tener que configurar cada nodo de acuerdo a reglas preestablecidas por un nodo maestro o autoridad central.

No obstante, a pesar de que el objetivo del trabajo en parte era implementar CFEngine como herramienta de monitoreo y administración única en la infraestructura de supercómputo, cabe destacar que es posible mantener Nagios y CFEngine en el mismo sistema e incluso poder realizar configuraciones para que Nagios sea el encargado del monitoreo, pero CFEngine el encargado de los reportes y administración automatizada.

Anexos

Glosario

API

Interfaz de programación de aplicaciones. Especifica los componentes y operaciones relacionados a un software determinado, así como sus datos de entrada, salida y tipos de datos. Define las funcionalidades independientemente de las implementaciones.

Clúster

Computadoras individuales (nodos) conectadas entre sí, que deben tener la integridad suficiente para funcionar como si fueran una única supercomputadora.

Cracker

La palabra *cracker* viene de *criminal hacker*, haciendo referencia a que tiene las mismas habilidades de un *hacker*, pero las utiliza regularmente para realizar acciones ilegales o inmorales.

Dato

Es un elemento informativo que por sí solo no tiene significado.

DHCP

Dynamic Host Configuration Protocol, es un protocolo de red que permite la asignación de direcciones IP, máscara de red y puerta de enlace de manera dinámica y automatizada

Disponibilidad

En informática, se refiere a cuando un servicio, dispositivo o proceso está disponible cuando se espera que deba de estarlo.

Framework

Es una abstracción de contenido donde el software provee de funcionalidad genérica que puede ser modificada para satisfacer necesidades más específicas sin tener que cambiar la base general de la solución. Puede contener programas, bibliotecas, estándares, API's y herramientas para su uso.

FTP

File Transfer Protocol, es un protocolo de red para la transferencia de archivos basada en un modelo cliente/servidor.

Hacker

Persona que es experta en el área de cómputo y que tiene entusiasmo para aplicar todos sus conocimientos regularmente para contribuir a un bien común.

Host

En informática, se le conoce como *host*, a un dispositivo o computadora dentro de una red que proporciona un servicio para el usuario.

HTTP

HyperText Transfer Protocol, es un protocolo para el envío de hipertexto en un modelo cliente/servidor. Es el protocolo utilizado en la *World Wide Web*.

Información

Conjunto de datos con un significado.

Integridad

En informática, la integridad se refiere a que la información almacenada o desplegada por una computadora sea la información que dice ser. Se refiere a la confiabilidad que se puede tener sobre esa información.

LAMP

Entorno de un servidor web que contiene un servidor Apache, una base de datos MySQL y un lenguaje de programación interpretado listo para web como PHP o Python.

NFS

Network File System, es un protocolo de red a nivel de aplicación que se utiliza para el uso de sistemas de archivos en red utilizando un modelo cliente/servidor.

Proceso

Instancia de un programa en ejecución.

Programa

Es un conjunto de instrucciones escritas que serán interpretadas por una computadora para resolver algún problema en particular.

Red informática

Conjunto de dispositivos conectados entre si a través de un medio físico que envía impulsos eléctricos para después ser interpretados como datos relevantes, con el objetivo de compartir información.

Script

Generalmente un archivo de texto, con instrucciones que serán ejecutadas línea por línea por un intérprete.

Seguridad

En informática, la seguridad se refiere a que la información almacenada en una computadora sea accedida sólo por las entidades autorizadas para hacerlo.

Sistema

Conjunto de elementos organizados y relacionados que interactúan entre sí para lograr un objetivo en común.

SMTP

Simple Mail Transfer Protocol, es un protocolo de red que se utiliza para la transferencia de correo electrónico por medio de una red informática.

Supercomputadora

Son aquellas computadoras con la máxima capacidad de procesamiento, almacenamiento, memoria, velocidad de comunicación y software que existen en la actualidad.

Topología de red

Modelo que hace una representación gráfica de la localización geográfica de cada uno de los dispositivos físicos de una red informática.

Mapas de red

Red Ethernet

ETHERNET

Hostname	IP	Puerto	Switch	Puerto
pb1	192.168.1.1	1	Ethernet	1
pb2	192.168.1.2	1	Ethernet	2
pb3	192.168.1.3	1	Ethernet	3
pb4	192.168.1.4	1	Ethernet	4
pb5	192.168.1.5	1	Ethernet	5
pb6	192.168.1.6	1	Ethernet	6
pb7	192.168.1.7	1	Ethernet	7
pb8	192.168.1.8	1	Ethernet	8
pb9	192.168.1.9	1	Ethernet	9
pb10	192.168.1.10	1	Ethernet	10
pb11	192.168.1.11	1	Ethernet	11
pb12	192.168.1.12	1	Ethernet	12
pb13	192.168.1.13	1	Ethernet	13
pb14	192.168.1.14	1	Ethernet	14
pb15	192.168.1.15	1	Ethernet	15
pb16	192.168.1.16	1	Ethernet	16
pb17	192.168.1.17	1	Ethernet	17
pb18	192.168.1.18	1	Ethernet	18
pb19	192.168.1.19	1	Ethernet	19
pb20	192.168.1.20	1	Ethernet	20
pb21	192.168.1.21	1	Ethernet	21
pb22	192.168.1.22	1	Ethernet	22
pb23	192.168.1.23	1	Ethernet	23
pb24	192.168.1.24	1	Ethernet	24
pb25	192.168.1.25	1	Ethernet	25
pb26	192.168.1.26	1	Ethernet	26
pb27	192.168.1.27	1	Ethernet	27
pb28	192.168.1.28	1	Ethernet	28
pb29	192.168.1.29	1	Ethernet	29
pb30	192.168.1.30	1	Ethernet	30
pb31	192.168.1.31	1	Ethernet	31
pb32	192.168.1.32	1	Ethernet	32
pb33	192.168.1.33	1	Ethernet	33
pb34	192.168.1.34	1	Ethernet	34
pb35	192.168.1.35	1	Ethernet	35
pb36	192.168.1.36	1	Ethernet	36
pb37	192.168.1.37	1	Ethernet	37
pb38	192.168.1.38	1	Ethernet	38
pb39	192.168.1.39	1	Ethernet	39
pb40	192.168.1.40	1	Ethernet	40
pb41	192.168.1.41	1	Ethernet	41
pb42	192.168.1.42	1	Ethernet	42
pb43	192.168.1.43	1	Ethernet	43
pb44	192.168.1.44	1	Ethernet	44
pb45	192.168.1.45	1	Ethernet	45
pb46	192.168.1.46	1	Ethernet	46
pbmaster	192.168.1.47	2	Ethernet	47

Red iLO

iLO

Hostname	IP	Puerto	Switch	Puerto
pb1	192.168.1.101	1	iLO	1
pb2	192.168.1.102	1	iLO	2
pb3	192.168.1.103	1	iLO	3
pb4	192.168.1.104	1	iLO	4
pb5	192.168.1.105	1	iLO	5
pb6	192.168.1.106	1	iLO	6
pb7	192.168.1.107	1	iLO	7
pb8	192.168.1.108	1	iLO	8
pb9	192.168.1.109	1	iLO	9
pb10	192.168.1.110	1	iLO	10
pb11	192.168.1.111	1	iLO	11
pb12	192.168.1.112	1	iLO	12
pb13	192.168.1.113	1	iLO	13
pb14	192.168.1.114	1	iLO	14
pb15	192.168.1.115	1	iLO	15
pb16	192.168.1.116	1	iLO	16
pb17	192.168.1.117	1	iLO	17
pb18	192.168.1.118	1	iLO	18
pb19	192.168.1.119	1	iLO	19
pb20	192.168.1.120	1	iLO	20
pb21	192.168.1.121	1	iLO	21
pb22	192.168.1.122	1	iLO	22
pb23	192.168.1.123	1	iLO	23
pb24	192.168.1.124	1	iLO	24
pb25	192.168.1.125	1	iLO	25
pb26	192.168.1.126	1	iLO	26
pb27	192.168.1.127	1	iLO	27
pb28	192.168.1.128	1	iLO	28
pb29	192.168.1.129	1	iLO	29
pb30	192.168.1.130	1	iLO	30
pb31	192.168.1.131	1	iLO	31
pb32	192.168.1.132	1	iLO	32
pb33	192.168.1.133	1	iLO	33
pb34	192.168.1.134	1	iLO	34
pb35	192.168.1.135	1	iLO	35
pb36	192.168.1.136	1	iLO	36
pb37	192.168.1.137	1	iLO	37
pb38	192.168.1.138	1	iLO	38
pb39	192.168.1.139	1	iLO	39
pb40	192.168.1.140	1	iLO	40
pb41	192.168.1.141	1	iLO	41
pb42	192.168.1.142	1	iLO	42
pb43	192.168.1.143	1	iLO	43
pb44	192.168.1.144	1	iLO	44
pb45	192.168.1.145	1	iLO	45
pb46	192.168.1.146	1	iLO	46
pbmaster	192.168.1.147	1	iLO	47

Red Infiniband

Infiniband				
Hostname	IP	Puerto	Switch	Puerto
pb1	192.168.2.1	1	Infiniband	1-1
pb2	192.168.2.2	1	Infiniband	1-2
pb3	192.168.2.3	1	Infiniband	1-3
pb4	192.168.2.4	1	Infiniband	1-4
pb5	192.168.2.5	1	Infiniband	1-5
pb6	192.168.2.6	1	Infiniband	1-6
pb7	192.168.2.7	1	Infiniband	1-7
pb8	192.168.2.8	1	Infiniband	1-8
pb9	192.168.2.9	1	Infiniband	1-9
pb10	192.168.2.10	1	Infiniband	1-10
pb11	192.168.2.11	1	Infiniband	1-11
pb12	192.168.2.12	1	Infiniband	1-12
pb13	192.168.2.13	1	Infiniband	1-13
pb14	192.168.2.14	1	Infiniband	1-14
pb15	192.168.2.15	1	Infiniband	1-15
pb16	192.168.2.16	1	Infiniband	1-16
pb17	192.168.2.17	1	Infiniband	1-17
pb18	192.168.2.18	1	Infiniband	1-18
pb19	192.168.2.19	1	Infiniband	1-19
pb20	192.168.2.20	1	Infiniband	1-20
pb21	192.168.2.21	1	Infiniband	1-21
pb22	192.168.2.22	1	Infiniband	1-22
pb23	192.168.2.23	1	Infiniband	1-23
pb24	192.168.2.24	1	Infiniband	1-24
pb25	192.168.2.25	1	Infiniband	2-1
pb26	192.168.2.26	1	Infiniband	2-2
pb27	192.168.2.27	1	Infiniband	2-3
pb28	192.168.2.28	1	Infiniband	2-4
pb29	192.168.2.29	1	Infiniband	2-5
pb30	192.168.2.30	1	Infiniband	2-6
pb31	192.168.2.31	1	Infiniband	2-7
pb32	192.168.2.32	1	Infiniband	2-8
pb33	192.168.2.33	1	Infiniband	2-9
pb34	192.168.2.34	1	Infiniband	2-10
pb35	192.168.2.35	1	Infiniband	2-11
pb36	192.168.2.36	1	Infiniband	2-12
pb37	192.168.2.37	1	Infiniband	2-13
pb38	192.168.2.38	1	Infiniband	2-14
pb39	192.168.2.39	1	Infiniband	2-15
pb40	192.168.2.40	1	Infiniband	2-16
pb41	192.168.2.41	1	Infiniband	2-17
pb42	192.168.2.42	1	Infiniband	2-18
pb43	192.168.2.43	1	Infiniband	2-19
pb44	192.168.2.44	1	Infiniband	2-20
pb45	192.168.2.45	1	Infiniband	2-21
pb46	192.168.2.46	1	Infiniband	2-22
pbmaster	192.168.2.47	1	Infiniband	2-23

Bibliografía

[1] Burgess, M.. (2002). *Theoretical System Administration*. junio 2, 2014, de Usenix.org Sitio web: https://www.usenix.org/legacy/publications/library/proceedings/lisa2000/full_papers/burgess/burgess_html/index.html

[2] Burgess, M.. (2003). *On the theory of system administration*. junio 2, 2014, de Mark Burgess Sitio web: http://markburgess.org/papers/sysadmtheory3.pdf

[3] Burgess, M.. (2004). *Principles of Network and System Administration*. Oslo University College, Norway: John Wiley & Sons, Ltd.

[4] Burgess, M. & Bergstra, J.. (2014). *Promise Theory Principles and Applications*. Oslo, Norway: xtAxis press.

[5] CFEngine. (2014). *CFEngine Documentation*. julio 5, 2014, de CFEngine Sitio web: https://docs.cfengine.com/docs/3.5/index.html

[6] Dirección General de Cómputo y de Tecnologías de Información y Comunicación. (2010). *Acuerdo TIC*. junio 22, 2014, de DGTIC Sitio web: http://www.tic.unam.mx/pdfs/AcuerdoTIC.pdf

[7] Dirección General de Cómputo y de Tecnologías de Información y Comunicación. (2014). *¿Qué es SC?*. junio 22, 2014, de DGTIC Sitio web: http://www.super.unam.mx/index.php/home/enlace

[8] Dumas II, Joseph. (2006). *Computer Architecture Fundamentals and Principles of Computer Design*. United States of America: Taylor & Francis Group.

[9] Feit, S.. (1996). *TCP/IP: Architecture, Protocols, and Implementation with IPv6 and IP Security*. United States of America: McGraw-Hill.

[10] Flores, Y. & Caballero, R.. (2010). *Apuntes de Administración en UNIX*. junio 25, 2014, de Facultad de Contaduría y Administración, UNAM Sitio web: http://fcasua.contad.unam.mx/apuntes/interiores/docs/2005/informatica/6/2048.pdf

[11] Ford, A.. (2008). *Apache 2 Pocket Reference*. United States of America: O'Reilly.

[12] Frisch, Æ.. (2002). *Essential System Administration*, 3rd Edition. United States of America: O'Reilly.

[13] Kocjan, W.. (2014). *Learning Nagios 4*. United States of America: Packt Publishing.

[14] Nagios. (2014). *Nagios Documentation*. julio 8, 2014, de Nagios Enterprises Sitio web: http://www.nagios.org/documentation

[15] Nemeth, E., Snyder, G., Hein, T. & Whaley, B.. (2001). *UNIX and Linux System Administration Handbook*. United States of America: Prentice Hall.

[16] Rajneeesh. (2011). *CFEngine 3 Beginner's Guide*. Birmingham, UK: Packt Publishing.

[17] Ray, J.. (1999). *Using TCP/IP*. United States of América: Que.

[18] Reiss, L. & Rodin, J.. (1993). *Unix System Administration Guide*. United States of America: Osborne.

[19] Rouse M.. (2007). *FCAPS (fault-management, configuration, accounting, performance, and security)*. julio 4, 2014, de SearchNetworking Sitio web: http://searchnetworking.techtarget.com/definition/FCAPS

[20] Zamboni, D.. (2012). *Learning CFEngine 3*. United States of America: O'Reilly.