UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

# IMPLEMENTACIÓN DE UN SISTEMA DE MONITORIZACIÓN PARA LA DIVISIÓN DE INGENIERÍAS CIVIL Y GEOMÁTICA UTILIZANDO LA HERRAMIENTA NAGIOS

**T E S I S**

QUE PARA OBTENER EL TITULO DE

I N G E N I E R O  E N  C O M P U T A C I Ó N

P R E S E N T A

**José María Ulises Corona Guerrero**

**DIRECTORA DE TESIS:**
**M.I. Tanya Itzel Arteaga Ricci**

CIUDAD UNIVERSITARIA 2013

# Índice general

# Índice de figuras

# Índice de tablas

# Introducción

"La División de Ingenierías Civil y Geomática tiene como objetivo formar profesionistas que posean una formación multidisciplinaria conformada con excelentes fundamentos en las áreas de Construcción, Estructuras, Geodesia, Cartografía, Geotecnia, Hidráulica, Sanitaria y Ambiental, Sistemas y Planeación, Topografía, Fotogrametría, así como conocimientos y habilidades adicionales que los complementen como computación, comunicación gráfica, informática, administración y evaluación de proyectos, planeación, diseño, organización, operación y conservación de obras civiles, para prestar servicios útiles a la sociedad con alto grado de ética y profesionalismo, logrando así una competitividad laboral."

Lo anterior es el primer párrafo que nos presenta la página principal de la DICYG (División de Ingenierías Civil y Geomática) con esto nos podemos dar cuenta de lo importante que es la Unidad de Cómputo (UC) de la DICYG, ya que ésta proporciona servicios tales como el Sistema de Inscripción para los Laboratorios de Hidráulica, el Sistema de Inscripción para Altas, Bajas y Cambios de Asignaturas, el Sistema de Titulación, entre otras cosas, además gestiona y mantiene las páginas *web* asociadas a los sistemas antes mencionados, administra la Red y proporciona el acceso a internet a estudiantes, académicos y funcionarios de la DICYG, facilita la difusión de eventos, conferencias, simposiums y mesas redondas de la DICYG, proporciona soporte técnico al personal académico y alumnos, imparte cursos intersemetrales que están disponibles para toda la comunidad universitaria.[22]

Los variados sistemas que administra la UC tienen bases de datos asociadas a éstos y son de vital importancia para su correcto funcionamiento, las páginas *web* que administra la UC también son un punto fundamental a cuidar ya que son las encargadas de mostrar la información que manejan los diferentes sistemas.

La UC es responsable de aspectos críticos que facilitan el trabajo al personal de la DICYG, y por esta razón la presente tesis propone la implementación de un Sistema de Monitorización, con el objetivo de que se prevengan incidencias en los servicios HTTP y MySQL, que están asociados a las páginas Web y a las Bases de Datos que manejan los sistemas, además de verificar continuamente la disponibilidad de su servidor que proporciona los servicios antes mencionados y que en caso de que se presente alguna incidencia, ésta pueda ser resuelta de manera rápida, eficaz y transparente para los usuarios por la UC con las herramientas que esta tesis proporciona.

# Objetivos

## Objetivo General

Mantener la disponibilidad continua de los servicios HTTP, MySQL y la disponibilidad del servidor *dicyg* mediante la implementación de un Sistema de Monitorización utilizando la herramienta Nagios y el planteamiento de un Plan de Contingencia a seguir en caso de interrupción en la disponibilidad de los servicios y/o de la disponibilidad del servidor antes mencionado.

## Objetivos Específicos

- Verificar la continua disponibilidad en los servicios HTTP, MySQL y la disponibilidad del servidor *dicyg* de la División de Ingenierías Civil y Geomática (DICYG) utilizando un Sistema de Monitorización con la herramienta Nagios.
- Notificar interrupciones en la disponibilidad de los servicios HTTP, MySQL y la disponibilidad del servidor *dicyg* vía correo electrónico al administrador de la red utilizando la herramienta Nagios.
- Solucionar las incidencias por interrupción en la disponibilidad de los servicios HTTP y MySQL utilizando la herramienta Nagios como primera opción.
- Plantear un Plan de Contingencia a seguir en caso de que se presente una interrupción en la disponibilidad de los servicios HTTP y/o MySQL.

# Marco Teórico

*Si has construido castillos en el aire,*

*tu trabajo no se pierde;*

*ahora coloca las bases debajo de ellos.*

George Bernard Shaw (1856-1950) Escritor irlandés.

# Capítulo 1

# Conceptos Básicos de Redes de Datos

[1]

## 1.1. Redes de Comunicación de Datos

Las redes de datos se desarrollaron como consecuencia de aplicaciones comerciales diseñadas para microcomputadoras. Por aquel entonces, las microcomputadoras no estaban conectadas entre sí, por lo cual no había una manera eficaz de compartir datos entre varias computadoras. Se tornó evidente que el uso de disquetes como almacenamiento externo, para compartir datos no era un método eficaz ni económico para desarrollar la actividad empresarial. La red *a pie* creaba copias múltiples de los datos. Cada vez que se modificaba un archivo, había que volver a compartirlo con el resto de los usuarios. Si dos usuarios modificaban el archivo, y luego intentaban compartirlo, se perdía alguno de los dos conjuntos de modificaciones. Las empresas necesitaban una solución que resolviera con éxito los tres problemas siguientes:

- Cómo evitar la duplicación de equipos informáticos y de otros recursos
- Cómo comunicarse con eficiencia
- Cómo configurar y administrar una red

Las empresas se dieron cuenta de que la tecnología de *networking* podía aumentar la productividad y ahorrar gastos. Las redes se agrandaron y extendieron casi con la misma rapidez con la que se lanzaban nuevas tecnologías y productos de red. A principios de la década de 1980 *networking* se expandió enormemente, aun cuando en sus inicios su desarrollo fue desorganizado.[24]

De esta manera se puede definir una red de datos como: "*Un conjunto de terminales, nodos, servidores y elementos de propósito especial que interaccionan entre sí con la finalidad de intercambiar información y compartir recursos*".[24]

Las razones principales que se pueden tomar en consideración para la utilización de una red de computadoras son:

**Integridad** Hacer de un sistema de varios elementos una sola herramienta, en donde se utilicen las características y las aptitudes de cada uno de la mejor forma posible.

---

[1] Las imágenes 1.1 a la 1.14 utilizadas en éste capítulo, son propiedad de *Cisco Systems Inc.*

**Flexibilidad** Fácil de utilizar, rápida comunicación de datos en cualquier momento y disponibilidad.

### Beneficio de las Redes Locales

Sin importar cual sea el tipo de red, todas tienen los siguientes objetivos en común:

- Minimizar los costos
- Cumplir con los requerimientos de disponibilidad
- Proveer todos los requerimientos necesarios

Se pueden minimizar costos teniendo en cuenta el diseño de la red, garantizando la disponibilidad mínima requerida de los servicios.

Las redes de datos locales ayudan a tener mayor eficiencia en el uso de los dispositivos y/o servicios que maneje la red ya que se aprovechan mejor los recursos, de esta manera se pueden reducir costos, además, el uso de las redes locales aumentan la disponibilidad de los recursos y/o servicios que se proporcionan, resultando, en que el usuario tenga todo lo necesario para poder desempeñar su trabajo de la mejor manera posible.

Algunos de los beneficios de estar en red pueden ser los siguientes:

- Aumento de la productividad
- Reducción de costos
- Optimización de la administración

## 1.2. Topologías de Red

La topología de red define la estructura de una red. Una parte de la definición topológica es la topología física, que es la disposición real de los cables o medios. La otra parte es la topología lógica, que define la forma en que los *host* acceden a los medios para enviar datos. Las topologías físicas más comunes son las siguientes[24]:

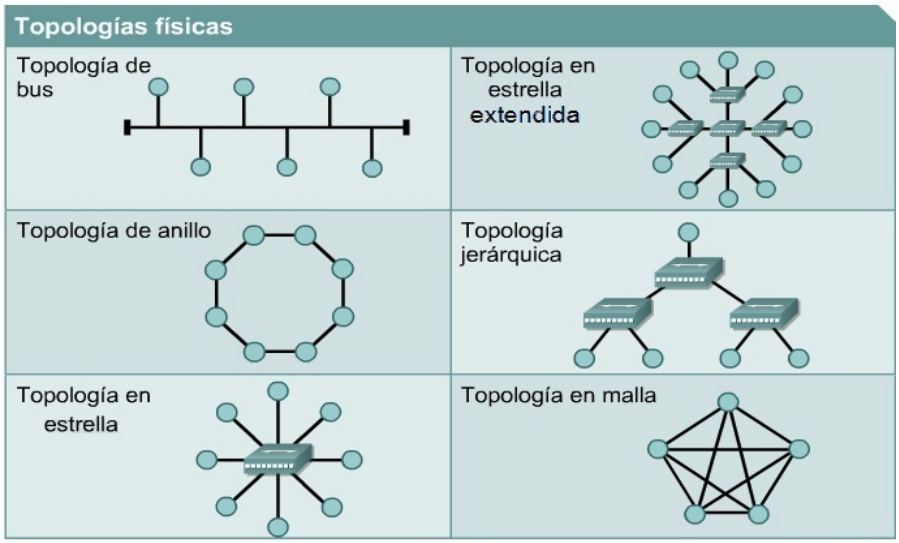


Figura 1.1: Topologías Físicas de Red más Comunes

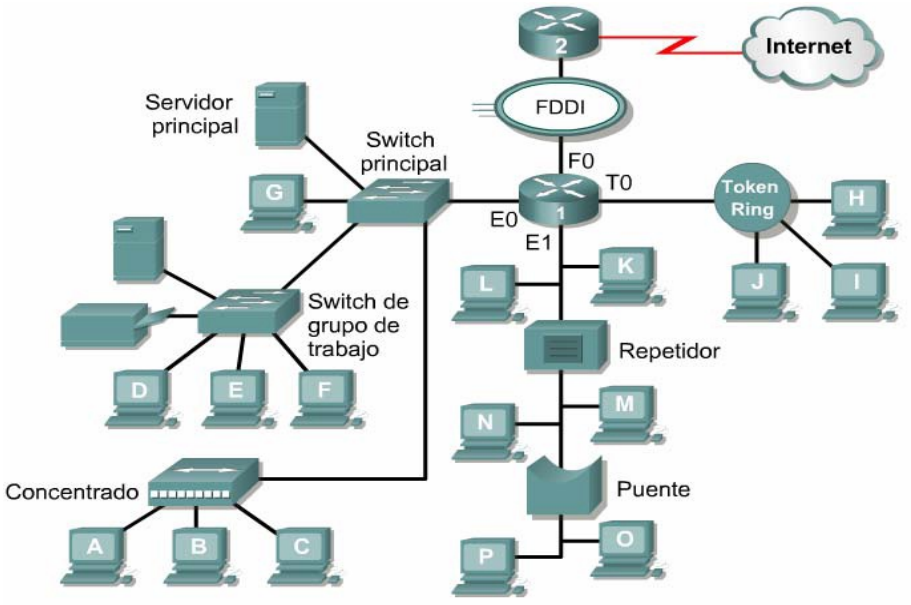


Figura 1.2: Combinación de las Topologías de Red

- Una topología de *bus* usa un solo cable *backbone* que debe terminarse en ambos extremos. Todos los *hosts* se conectan directamente a este *backbone*.
- La topología de anillo conecta un *host* con el siguiente y al último *host* con el primero. Esto crea un anillo físico de cable.
- La topología en estrella conecta todos los cables con un nodo central de concentración.
- Una topología en estrella extendida conecta estrellas individuales entre sí mediante la conexión de *hubs* o *switches*. Esta topología puede extender el alcance y la cobertura de la red.

- Una topología jerárquica es similar a una estrella extendida. Pero en lugar de conectar los *hubs* o *switches* entre sí, el sistema se conecta a una computadora que controla el tráfico de la topología.
- La topología de malla se implementa para proporcionar la mayor protección posible evitando una interrupción del servicio. Cada *host* tiene sus propias conexiones con los demás *hosts*. Aunque la Internet cuenta con múltiples rutas hacia cualquier ubicación, no adopta la topología de malla completa.

En el diagrama de la figura 1.2 se muestra diferentes topologías conectadas mediante dispositivos de red.

## 1.3. Clasificación de Algunas Redes de Datos por Cobertura Geográfica

De acuerdo a su cobertura geográfica, las redes son clasificadas de la siguiente manera:

- ***Local Area Network* (LAN):** Las cuales son típicamente extendidas hasta 1 km.[29]
- ***Wide Area Network* (WAN):** Redes de área amplia. Redes que tienen una cobertura mundial. Tienen la finalidad de interconexión de redes LAN.[29]
- ***Metropolitan Area Network* (MAN):** Redes de cobertura de 10, 20 ó 30 km.[29] Actualmente las redes MAN no son muy utilizadas, ya que han sido absorbidas por las redes WAN y LAN, aunque teóricamente se encuentran todavía.

### *LAN*

Son redes de datos de alta velocidad y bajo nivel de errores, abarcan áreas geográficas relativamente pequeñas. Las redes LAN conectan nodos que se encuentran generalmente en una misma entidad física (por ejemplo, un edificio).

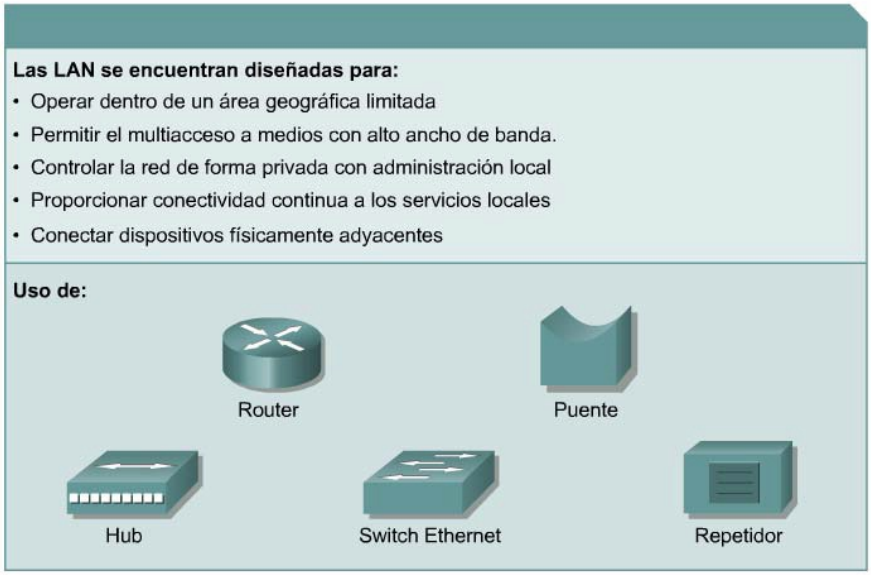


Figura 1.3: Características y Dispositivos Importantes de las Redes LAN

Las LAN permiten a las organizaciones aplicar tecnología informática para compartir localmente archivos e impresoras de manera eficiente y posibilitar las comunicaciones internas.

***WAN***

Las WAN interconectan las LAN, que a su vez proporcionan acceso a los *hosts* o a los servidores de archivos ubicados en otros lugares. Como las WAN conectan redes de usuarios dentro de un área geográfica extensa, permiten que las organizaciones se comuniquen entre sí a través de grandes distancias. Las WAN permiten que los *hosts*, impresoras y otros dispositivos de una LAN compartan y sean compartidos por redes en sitios distantes. Las WAN proporcionan comunicaciones casi instantáneas a través de zonas geográficas extensas. El *software* de colaboración brinda acceso a información en tiempo real y recursos que permiten realizar reuniones entre personas separadas por largas distancias, en lugar de hacerlas en persona. *Networking* de área amplia también dio lugar a una nueva clase de trabajadores, los empleados a distancia, que no tienen que salir de sus hogares para ir a trabajar.

Las WAN están diseñadas para realizar lo siguiente[24]:

- Operar entre áreas geográficas extensas y distantes
- Posibilitar capacidades de comunicación en tiempo real entre usuarios
- Brindar recursos remotos de tiempo completo, conectados a los servicios locales
- Brindar servicios de correo electrónico, *World Wide Web*, transferencia de archivos y comercio electrónico

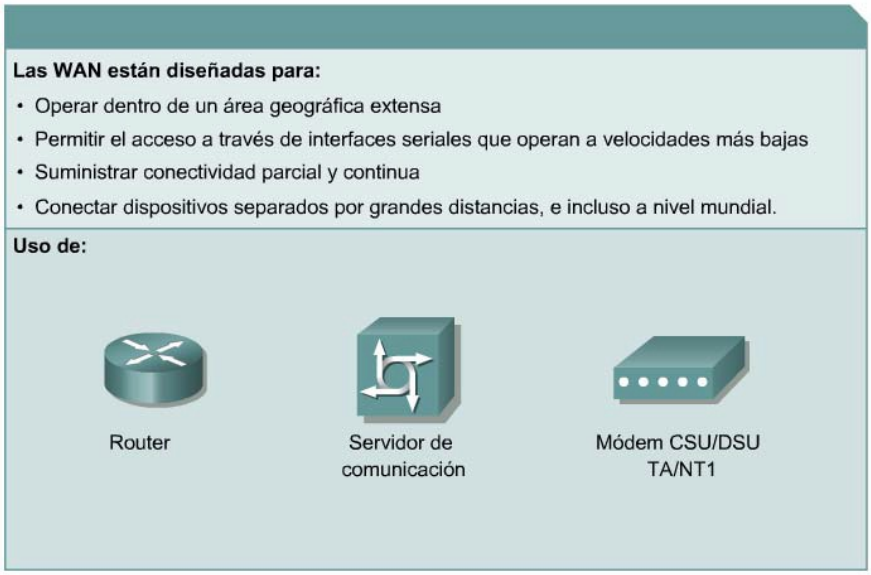


Figura 1.4: Características y Dispositivos de las Redes WAN

***MAN***

La MAN es una red que abarca un área metropolitana, como, por ejemplo, una ciudad. Una MAN generalmente consta de una o más LAN dentro de un área geográfica común. Por ejemplo, un banco con varias sucursales puede utilizar una MAN. Normalmente, se utiliza un proveedor de servicios para conectar dos o más sitios LAN utilizando líneas privadas de comunicación.

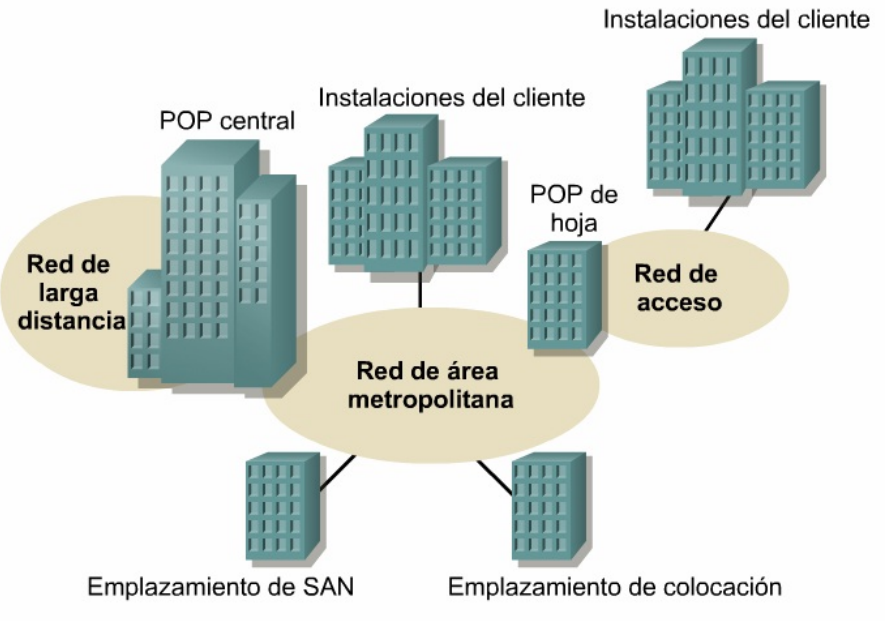


Figura 1.5: Ejemplo de una red MAN

## 1.4. Organismos de Estandarización

La estandarización asegura que un producto o servicio siga ciertas reglas que serán aceptadas por todos los que hagan uso del mismo, contando con calidad, seguridad, eficiencia e interoperabilidad y a un costo aceptable. Existe una gran cantidad de organizaciones que son usados para la estandarización, algunos de ellos son: ANSI *(American National Standards Institute)*, TIA *(Telecommunications Industry Association)*, EIA *(Electronics Industry Association)* y NOM (Norma Oficial Mexicana) en México. Por otro lado, aun cuando se define un estándar que es aceptado por todos los que forman el consorcio, es necesario en varios casos cumplir con "reglas" que el gobierno define. Estas agencias gubernamentales son las encargadas de regular el mercado, desde la parte técnica, así como comercial (cumplir con la constitución, evitar monopolios, etc). En México, el organismo regulador es la COFETEL (Comisión Federal de Telecomunicaciones)[2].

### ISO *(International Organization for Standardization)*

ISO (Organización Internacional de Estandarización) es el mayor desarrollador mundial de las Normas Internacionales voluntarias. Fundada en 1947, y desde entonces ha publicado más de 19 500 normas internacionales que abarcan casi todos los aspectos de la tecnología y los negocios. Hoy cuenta con miembros de 163 países y alrededor de 150 personas trabajan a tiempo completo en su Secretaría Central en Ginebra, Suiza.[5]

### IEEE *(Institute of Electrical and Electronics Engineers)*

En 1961 las dirigencias tanto del IRE (*Institute of Radio Engineers*) como del AIEE (*American Institute of Electrical Engineers*) aprobarón un plan de fusión para unificar las actividades técnicas y las unidades geográficas de las dos sociedades y para establecer un programa de

[2]En el momento que se realizó este trabajo la COFETEL aún era el organismo regulador. A partir de Junio de 2013 el organismo regulador es la IFT (Instituto Federal de Telecomunicaciones).[21]

publicaciones unificado para la nueva organización, el Instituto de Ingenieros Eléctricos y Electrónicos (IEEE).
El IEEE es una de las organizaciones más grandes y prestigiada del mundo en la rama.[1][20]

## ANSI *(American National Standards Institute)*

Fundada en 1918, actúa como interfaz gubernamental de todos los estándares producidos por otros organismos para Norteamérica como son la IEEE, EIA, etc. ANSI define los estándares de la industria eléctrica e industrial tanto para la comunidad nacional (americana) como la comunidad internacional.[17]

## EIA *(Electronics Industries Alliance)*

Alianza de Industrias Electrónicas, que hasta 1997 fué conocida como *Electronic Industries Association*, es una organización formada por la asociación de las compañías electrónicas y de alta tecnología de los Estados Unidos, cuya misión era promover el desarrollo del mercado y la competitividad de la industria de alta tecnología de los Estados Unidos. La EIA tenía establecida su central en Arlington, Virginia. Los productos y servicios que ofrecía abarcaban desde los componentes electrónicos más pequeños a los sistemas más complejos usados para la defensa, el espacio y la industria, incluyendo la gama completa de los productos electrónicos de consumo.[18]

## TIA *(Telecommunications Industry Association)*

Asociación de la Industria de Telecomunicaciones. Una organización de comercio norteamericana que se especializa en el desarrollo de estándares para cableados de telecomunicaciones y sus estructuras de soporte.

## NOM *(Normal Oficial Mexicana)*

La NOM es la regulación técnica de observancia obligatoria expedida por las dependencias competentes, que establece reglas, especificaciones, atributos, directrices, características o prescripciones aplicables a un producto, proceso, instalación, sistema, actividad, servicio o método de producción u operación, así como aquellas relativas a terminología, simbología, marcado o etiquetado y las que se refieran a su cumplimiento o aplicación. Los certificados NOM se expedirán por producto o familia, por tipo y modelo y solo se otorgarán a importadores, fabricantes y comercializadores mexicanos y nacionales de otros países con los que el gobierno mexicano haya suscrito algún acuerdo o tratado de libre comercio.[29]

## COFETEL *(Comisión Federal de Telecomunicaciones)*

La Comisión Federal de Telecomunicaciones es el órgano administrativo independiente de la Secretaría de Comunicaciones y Transportes, con autonomía técnica, operativa y de gestión, encargado de regular, promover y supervisar el desarrollo eficiente y la cobertura social amplia de las telecomunicaciones y la radiodifusión en México, con autonomía plena para dictar sus resoluciones.[29]

## 1.5. Modelo OSI

En sus inicios, el desarrollo de redes sucedió con desorden en muchos sentidos. A principios de la década de 1980 se produjo un enorme crecimiento en la cantidad y el tamaño de las redes. A medida que las empresas tomaron conciencia de las ventajas de usar tecnología de *networking*, las redes se agregaban o expandían a casi la misma velocidad a la que se introducían las nuevas tecnologías de red.[24]

Para enfrentar el problema de incompatibilidad de redes, la Organización Internacional de Normalización (ISO) desarrolló un modelo de red que ayuda a los fabricantes a crear redes que sean compatibles con otras redes.

El modelo de referencia de Interconexión de Sistemas Abiertos (OSI) lanzado en 1984 fue el modelo de red descriptivo creado por ISO. Proporcionó a los fabricantes un conjunto de estándares que aseguraron una mayor compatibilidad e interoperabilidad entre los distintos tipos de tecnología de red producidos por las empresas a nivel mundial.[24]

El modelo de referencia OSI se ha convertido en el modelo principal para la enseñanza de las comunicaciones por red. Aunque existen otros modelos, la mayoría de los fabricantes de redes relacionan sus productos con el modelo de referencia de OSI.

### 1.5.1. Las capas del Modelo OSI

El modelo de referencia OSI es un marco que se puede utilizar para comprender cómo viaja la información a través de una red, y explica de qué manera los paquetes de datos viajan a través de varias capas a otro dispositivo de una red, aún cuando el remitente y el destinatario poseen diferentes tipos de medios de red.

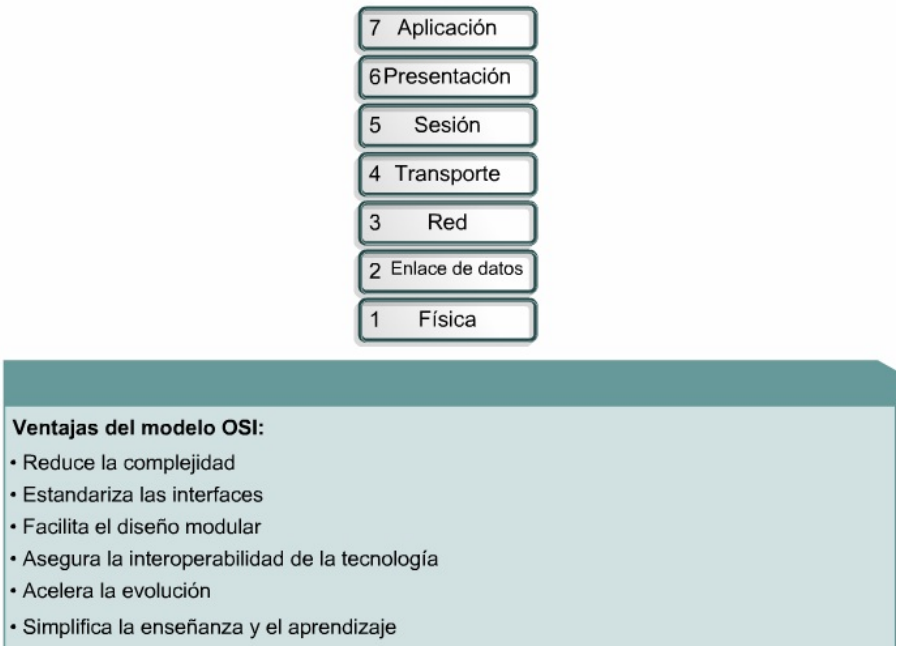


Figura 1.6: Capas y Ventajas del Modelo de Referencia OSI

En el modelo de referencia OSI, hay siete capas numeradas, cada una de las cuales ilustra una función de red específica. A continuación se dará una breve descripción de las siete capas del modelo de referencia OSI.

### Capa Física

Este nivel dirige la transmisión de flujos de bits sobre un medio de conexión. Se encuentra relacionado con condiciones eléctricas-ópticas, mecánicas y funcionales de la interfaz al medio de transmisión. A su vez esta encargado de aportar la señal empleada para la transmisión de los datos generados por los niveles superiores.

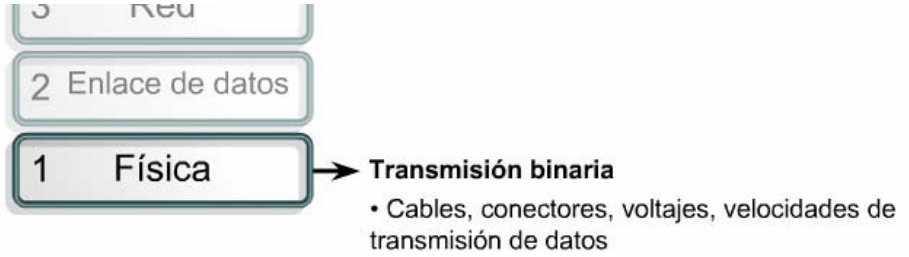


Figura 1.7: Características de la Capa Física del Modelo OSI

### Capa de Enlace de Datos

Establece un patrón de comunicaciones libre de errores entre los nodos de la red sobre el canal físico.


Figura 1.8: Características de la Capa de Enlace de Datos del Modelo OSI

### Capa de Red

Se encarga del encaminamiento (*routing*) de paquetes entre el origen y el destino, atravesando tantas redes intermedias como sean necesarias. Los mensajes se fragmentan en paquetes y cada uno de ellos se envía de forma independiente.

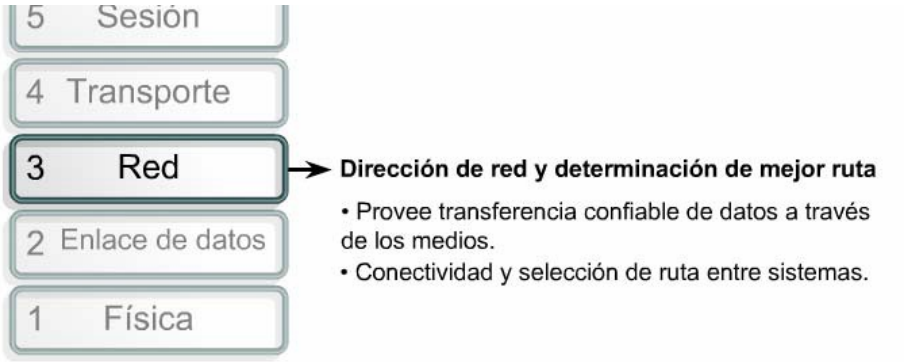


Figura 1.9: Características de la Capa de Red del Modelo OSI

### Capa de Transporte

Esta capa asegura que se reciban todos los datos y en el orden adecuado realiza un control de extremo a extremo.

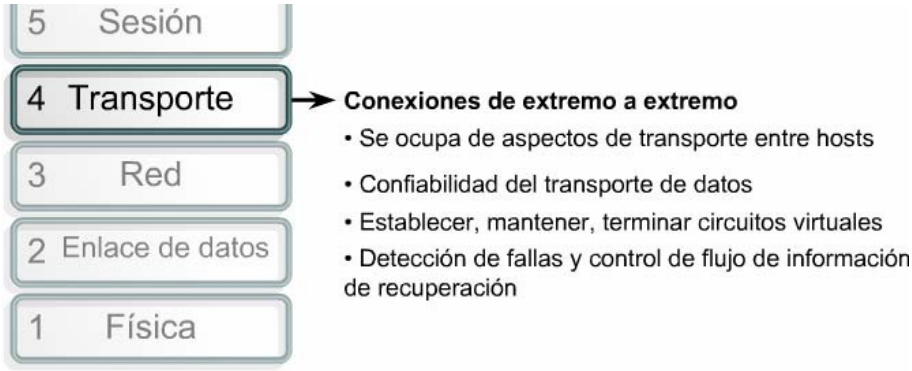


Figura 1.10: Características de la Capa de Transporte del Modelo OSI

### Capa de Sesión

Provee mecanismos para organizar y estructurar diálogos entre procesos de aplicación. Actúa como un elemento moderador capaz de coordinar y controlar el intercambio de los datos. Controla la integridad y el flujo de los datos en ambos sentidos.


Figura 1.11: Características de la Capa de Sesión del Modelo OSI

### Capa de Presentación

Traduce y convierte los datos codificados transmitidos en formatos que puedan ser entendidos y manipulados por el usuario, determina el formato a usar para el intercambio de datos en la red.

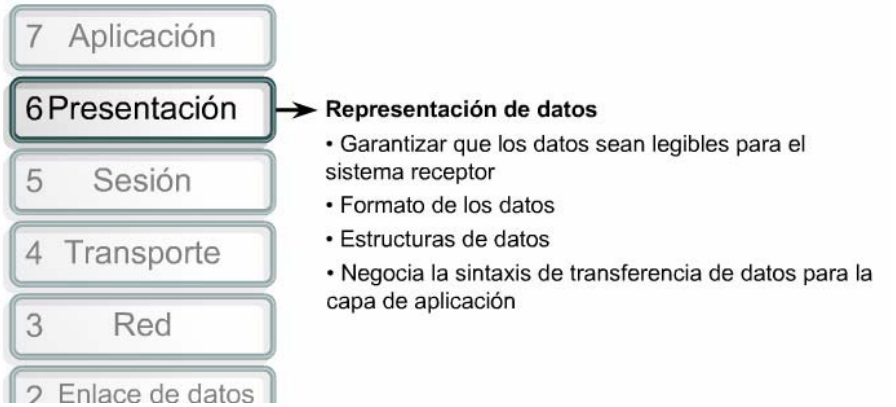


Figura 1.12: Características de la Capa de Presentación del Modelo OSI

### Capa de Aplicación

Aquí se encuentran los protocolos y programas que utiliza el usuario para sus comunicaciones en red.


Figura 1.13: Características de la Capa de Aplicación del Modelo OSI

## 1.6. Modelo de Referencia TCP/IP

El estándar histórico y técnico de la Internet es el modelo TCP/IP. El Departamento de Defensa de EE.UU. (DoD) creó el modelo de referencia TCP/IP porque necesitaba diseñar una red que pudiera sobrevivir ante cualquier circunstancia, incluso una guerra nuclear. Este difícil problema de diseño dio origen a la creación del modelo TCP/IP.[24]

El modelo TCP/IP se desarrolló como un estándar abierto. Esto significaba que cualquier persona podía usar el TCP/IP. Por lo tanto contribuyó a acelerar el desarrollo de TCP/IP como un estándar.

El modelo TCP/IP tiene las siguientes cuatro capas:


Figura 1.14: Capas del Modelo de Referencia TCP/IP

### Capa de Acceso a la Red

También se conoce como la capa de *host* a red. Esta capa guarda relación con todos los componentes, tanto físicos como lógicos, necesarios para lograr un enlace físico. Incluye los detalles de tecnología de *networking*, y todos los detalles de las capas Física y de Enlace de Datos del modelo OSI.
La capa de acceso de red se refiere a cualquier tecnología en particular utilizada en una red específica.

### Capa de Internet

El propósito de la Capa Internet es dividir los segmentos TCP en paquetes y enviarlos desde cualquier red. Los paquetes llegan a la red de destino independientemente de la ruta que utilizaron para llegar allí. El protocolo específico que rige esta capa se denomina *Internet Protocol* (IP). En esta capa se produce la determinación de la mejor ruta y la conmutación de paquetes.

La relación entre IP y TCP es importante. Se puede pensar en el protocolo IP como el que indica el camino a los paquetes, en tanto que el protocolo TCP brinda un transporte seguro.

### Capa de Transporte

La Capa de Transporte se encarga de los aspectos de calidad del servicio con respecto a la confiabilidad, el control de flujo y la corrección de errores. Uno de sus protocolos es el *Transmision Control Protocol* (TCP).

TCP es un protocolo orientado a conexión. Mantiene un diálogo entre el origen y el destino mientras empaqueta la información de la capa de aplicación en unidades denominadas segmentos. Orientado a conexión no significa que existe un circuito entre los *hosts* que se comunican. Significa que segmentos de la Capa 4 viajan de un lado a otro entre dos *hosts* para comprobar que la conexión exista lógicamente para un determinado período.

### Capa de Aplicación

Aunque algunas de las capas del modelo TCP/IP tienen el mismo nombre que las capas del modelo OSI, las capas de ambos modelos no se corresponden de manera exacta. Los diseñadores de TCP/IP incluyeron detalles de las Capas de Sesión y Presentación a la Capa de Aplicación del modelo TCP/IP. Crearon una Capa de Aplicación que maneja aspectos de representación, codificación y control de diálogo.

La figura 1.15 ilustra algunos de los protocolos comunes especificados por las capas del modelo de referencia TCP/IP.

Algunos de los protocolos de capa de aplicación más comúnmente usados incluyen los siguientes:

- Protocolo de Transferencia de Archivos (FTP)
- Protocolo de Transferencia de Hipertexto (HTTP)
- Protocolo simple de transferencia de correo (SMTP)
- Sistema de denominación de dominios (DNS)
- Protocolo Trivial de Transferencia de Archivos (TFTP)

Los protocolos de capa de transporte comunes incluyen:

- Protocolo para el Control del Transporte (TCP)
- Protocolo de Datagrama de Usuario (UDP)

El protocolo principal de la capa Internet es:

- Protocolo Internet (IP)

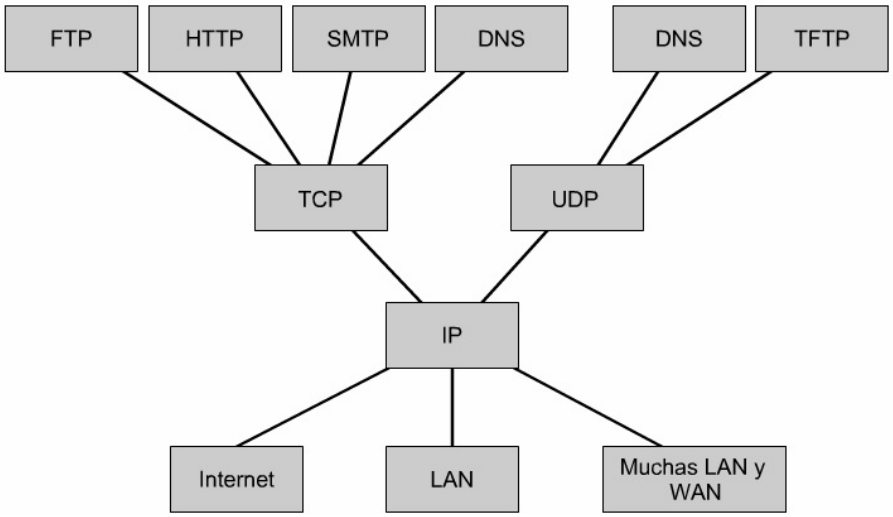


Figura 1.15: Protocolos comunes usados por el Modelo de Referecia TCP/IP

Independientemente de los servicios de aplicación de red que se brinden y del protocolo de transferencia que se utilice, existe un solo protocolo de Internet, IP. Esta es una decisión de diseño deliberada. IP sirve como protocolo universal que permite que cualquier computadora en cualquier parte del mundo pueda comunicarse en cualquier momento.[24]

# Capítulo 2

# Conceptos Básicos de la Administración de Redes

Una red es extremadamente importante, por lo que se convierte en algo que se debe diseñar, implementar, dirigir y controlar, además es necesario que exista un administrador de red, el cual será el encargado de mantener el correcto funcionamiento de la misma.

## 2.1. Definición de Administración de Redes

En el ámbito empresarial, la administración de redes es un instrumento vital en la planeación, organización, integración, dirección y control de los elementos de la comunicación, para garantizar un adecuado nivel de servicio, de acuerdo a un costo determinado. Al igual que las arquitecturas de redes, los sistemas de administración de redes no son idénticos, por esta razón el administrador de red tiene que encontrar el equilibrio entre los tipos de servicios que se ofrecen a los usuarios, la calidad de estos, además de determinar los medios que se deben implementar con el fin de lograr el nivel de calidad deseado.

Un servicio de red es un producto que es consumido por el usuario o cliente, que goza de derechos y cumple las obligaciones que se estipulan en el contrato del servicio.

Desde el punto de vista del usuario, la arquitectura, el funcionamiento y la administración de la red debe ser totalmente transparente.

La red debe satisfacer sus necesidades, proporcionando una transmisión fiable de información, sin pérdida ni errores, y con un tiempo de respuesta satisfactorio.

Cualquier sistema de administración de red debe gestionar y controlar la red, independientemente de la arquitectura de la red y la complejidad, de tal manera que las necesidades de los usuarios y operadores sean plenamente cubiertas

Las actividades que debe cumplir el administrador de la red son:

- Planeación: Debe considerar la infraestructura de la red, las políticas y la ética que rige el cómputo.
- Organización: Implementar los modelos de la administración de los protocolos utilizados en las redes de datos.

- Integración: El administrador debe reunir e implementar las tecnologías actuales para el adecuado funcionamiento de la red.
- Dirección: Adquirir o desarrollar la capacidad de liderazgo que le permita interactuar con el equipo de trabajo, con el objetivo de conjuntar esfuerzos.
- Control: Monitorización de la red para optimizar el rendimiento, disponibilidad y funcionalidad de ésta, manejando estándares para la medición, ejecución, acciones preventivas y correctivas.

## 2.2. Objetivos de la Administración de Redes

La administración de redes es un conjunto de técnicas que ayudan a mantener una red operativa, eficiente, segura, constantemente monitorizada, con una planeación adecuada y propiamente documentada. Los principales objetivos de la administración de redes son:

- Mejorar la continuidad en la operación de la red con mecanismos adecuados de control, monitorización, detección de errores y suministro de recursos.
- Utilizar los recursos correctamente para una mejor eficiencia.
- Fortalecer la red, usando métodos de seguridad efectivos.
- Controlar cambios y actualizaciones en la red sin atentar contra la disponibilidad de ésta.
- Crear redes convergentes (voz, datos y video)
- Interconectar redes LAN, MAN, WAN, utilizando diferentes medios de comunicación como par trenzado, fibra óptica, satelital, entre otros.

El modelo OSI considera que las cinco funciones de administración básicas son[27]:

- Configuración
- Errores
- Contabilidad
- Comportamiento
- Seguridad

## 2.3. Elementos de la Administración de una Red

La administración de una red requiere de la habilidad de supervisar, comprobar, sondear, configurar y controlar los componentes de *hardware* y *software* de una red. Dado que los dispositivos de red son distribuidos, el administrador debe ser capaz de recopilar datos, para la supervisión de entidades remotas, así como realizar cambios sobre ellas, para controlarlas. Con el fin de realizar las actividades mencionadas se hace uso de una arquitectura de sistemas de administración de redes, la cual se compone de los siguientes elementos:

**Entidad Administradora:** Consiste en una aplicación de control humano que se ejecuta en una estación centralizada de administración de red en el Centro de Operaciones de Red (NOC, *Network Operations Centers*). En el NOC se realiza la actividad de la administración de la red, se controla la recolección, procesamiento, análisis o visualización de la información de la administración, así como la iniciación de las acciones que controlan el comportamiento de la red, y la interacción del administrador de red con los dispositivos que lo conforman.

**Dispositivo Administrado:** Parte del equipamiento de la red, incluido el *software*, que se localiza en la red administrada. Un dispositivo administrado puede ser desde un *host* hasta una impresora o un modem. Dentro del dispositivo existen varios objetos administrados, como el *hardware* o la tarjeta de red.

**Base de Información de Administración:** Lugar donde se almacenan los datos referentes a los objetos administrados.

**Agente de Administración de Red:** Proceso que se ejecuta en cada dispositivo administrado y que se comunica con la entidad administradora, realizando acciones locales bajo el control de los comandos de la entidad administradora.

**Protocolo de Administración de Red:** Permite a la entidad administradora consultar el estado de los dispositivos, e indirectamente realizar acciones en dichos dispositivos a través de los agentes. Los agentes pueden utilizar el protocolo de administración de red para informar a la entidad administradora de eventos excepcionales, tales como el fallo de componentes.

El protocolo de administración proporciona una herramienta que ayuda al administrador de red a cumplir sus funciones.

# Capítulo 3

# Monitorización

A través del tiempo, las tecnologías han ido avanzando y evolucionando para soportar redes complejas actuales, además de que las organizaciones se han dado cuenta que la mejor manera de optimizar sus procesos de TI es integrando soluciones de administración de sistemas. La administración de las Tecnologías de Información se apoyan con soluciones de monitorización que ayudan a identificar rápidamente fallas, cuellos de botella o bajo rendimiento del sistema.

Las redes de cómputo de las organizaciones, se vuelven cada vez más complejas y la exigencia de la operación es cada vez más demandante. Las redes soportan aplicaciones y servicios estratégicos de las organizaciones. Por lo cual el análisis y monitorización de redes se ha convertido en una labor cada vez más importante y de carácter proactivo para evitar problemas.

La monitorización de sistemas se encarga de supervisar continuamente los diferentes recursos y servicios de las organizaciones para garantizar el nivel de disponibilidad requerido y en caso de un posible fallo alertar al administrador para que lo solucione. El objetivo de la monitorización de sistemas es garantizar que éste funcione correctamente y minimizar el tiempo de caída de un servicio.

En la figura 3.1 podemos ver la evolución temporal de un cíclo de incidente. El tiempo empleado en la detección del incidente es crítico, ya que si se tarda mucho en detectar el fallo difícilmente se podrá solucionar. El objetivo de la monitorización es garantizar que el tiempo de detección de un fallo sea mínimo.

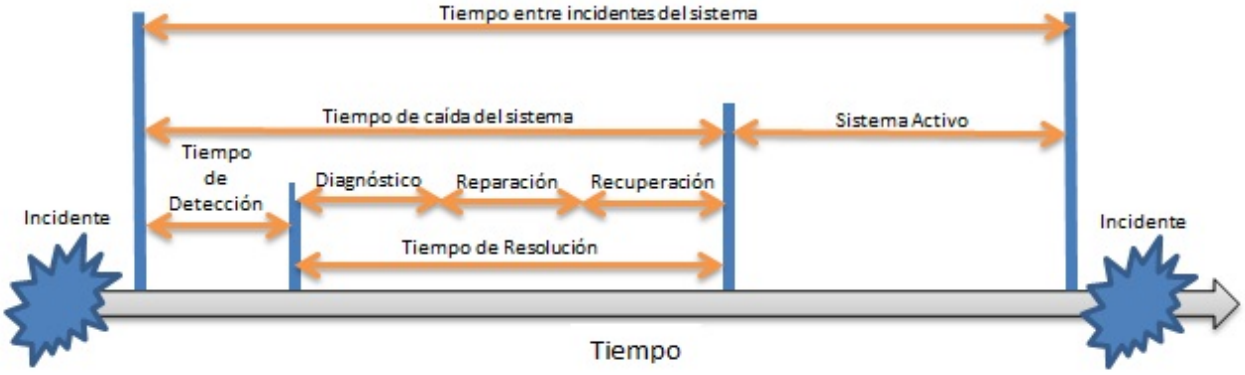


Figura 3.1: Cíclo de un incidente en un sistema[26]

De forma resumida, se puede decir que algunos de los principales objetivos de la monitori-

zación de sistemas son:

- Asegurar que el nivel de disponibilidad requerido esté proporcionado.
- Permitir la supervisión y análisis de la disponibilidad para asegurar que los niveles de servicios se están cumpliendo.
- Supervisar continuamente la disponibilidad de los servicios y en caso de que se produzca un fallo alertar al administrador para iniciar los procedimientos a desarrollar.
- Minimizar el tiempo de detección de los incidentes para garantizar la disponibilidad de los recursos y servicios.

Dado el gran número de equipos de algunas organizaciones (servidores, *routers*, puntos de acceso, etc.) resulta de vital importancia poder monitorizar dichos recursos de forma remota y centralizada.

Los recursos de TI no disponibles provocan un bajo desempeño además de que bajan las expectativas que se puedan tener de la organización. Por lo tanto es importante prevenir fallos en los procesos críticos de la organización.

Algunas de las principales causas que provocan dos tipos de pérdidas en la productividad son[3]:

- **Caídas del sistema (*Hardware*):** El problema surge cuando debido al fallo de un equipo no se realiza una actividad de la organización o se presenta un problema físico o un ataque en algún punto de la infraestructura, que da como resultado que los usuarios de la organización que realizan esa actividad detengan el trabajo que se esté realizando.
- **Bajo desempeño (*Software*):** Este problema se presenta en los sistemas que componen la red, servidores de aplicación, dispositivos de interconexión, segmentos y enlaces. Un bajo desempeño afecta la productividad, el potencial del personal a cargo y los recursos de la organización.

Más allá de los proyectos, en el día a día, la prevención es de suma importancia, las "paradas" por incidencias ya no son aceptables y se exige a los departamentos de TI estar al 100 % de tiempo operativo a medida de lo posible.[26]

Hoy en día existen soluciones de monitorización que apoyan al administrador de red a evitar en una gran proporción que procesos críticos presenten problemas graves, haciendo una administración proactiva y anticipando problemas que podrían presentarse a futuro.

## 3.1. Cuadro Comparativo de Sistemas de Monitorización

Cabe mencionar que hay una gran cantidad de herramientas de sistemas de monitorización de *software libre*, para la presente tesis se eligió hacer la comparación de tres: Nagios, Zenoss y Pandora FMS.

Con base en las caracteristicas proporcionadas por las páginas *web* de cada uno de los sitios, se realizó una comparación entre los Sistemas de Monitorización.[10, 16, 15, 8, 9, 2]

Los parámetros a evaluar fuerón los siguientes:

- Sistemas Operativos que soporta el servidor
- Sistemas Operativos que soporta el cliente
- Requerimientos de *Hardware*
- Tipos de Notificaciones que realiza
- Características Relevantes

| | S.O. Soportados en el Servidor | S.O. Soportados en el Cliente | Requerimientos de Hardware | Notificaciones | Características Relevantes |
|---|---|---|---|---|---|
| Nagios | **Kernel 2.4+ Distribuciones Linux:** RHEL, SLES, Ubuntu, Debian, CentOS. **Unix:** Solaris 9+, FreeBSD 6.4+. | **Windows:** 2000, XP, 2003, Vista y 7 **Linux/Unix:** Kernel 2.4+, Solaris 9+, FreeBSD 6.4+, AIX 5.2/5.3 | 1GHz CPU / 512MB RAM (mínimo) 2GHz+ CPU / 1GB+ RAM (recomendado) | Email, SMS, RSS, Jabber, Custom Script | Reinicio automático de aplicaciones, servicios y dispositivos usando event handlers |
| Zenoss | RHEL, Fedora Core, Mac OSX, Ubuntu, SUSE, CentOS, Windows(Usando VMware) | Windows, Linux y Unix | CPU 2 núcleos / 4GB RAM / Disco Duro 1x300GB 10K RPM / RAID 1 (mínimo) CPU 4 núcleos / 8GB RAM / Disco Duro 1x300GB 10K RPM / RAID 1 (medio) CPU 8 núcleos / 16-32GB RAM / Disco Duro 1x300GB 18K RPM / RAID 1 (grande) | Paging, Email, SMS | Autodescubrimiento de servidores y dispositivos de red |
| Pandora FMS | SUSE, Debian, Ubuntu, RHEL/CentOS | AIX 5.X, Solaris, Android, Dispositivos Empotrados, HP-UX11, Linux, BSD, Windows: NT4, 2000, XP, Vista y 7 | 2GHz CPU / 4GB RAM / Disco Duro 7200 RPM+ | Email, Alerta con Sonido, SMS, Jabber | Conexión SSH/Telnet a dispositivos desde la interfaz web |

Tabla 3.1: Comparativo de los Sistemas de Monitorización: Nagios, Zenoss y Pandora FMS

## 3.2. Tiempo de Resolución de Incidencias

El tiempo de identificación de un problema, mejora notablemente con la utilización de Nagios. Su meta es asegurar que el administrador de la red identifique el problema antes que lo hagan los usuarios.

La supervisión y análisis de todos los servicios críticos y la notificación correcta es clave para reducir el tiempo de resolución.

Los informes de incidencias ayudarán en la predicción de problemas y en la identificación de necesidades de crecimiento.

## 3.3. Nagios

Cuando se quiere monitorizar muchos dispositivos es imposible hacerlo manualmente, para ello, lo mejor es, utilizar herramientas de monitorización. Una de las características de Nagios es que va comprobando de forma periódica que los servicios y/o dispositivos funcionen correctamente. Y en caso de fallo enviar notificaciones al administrador de la red.

Para prevenir errores en un sistema podemos utilizar un equipo que se ocupe de estar "*controlando y observando*" el funcionamiento de la red, esto podemos realizarlo con Nagios.

Nagios es un sistema de monitorización de equipos y de servicios, escrito en C y publicado bajo la *GNU General Public License*, que lo determina como *Software Libre* y nos asegura que tendremos actualizaciones disponibles, además de que hay una gran comunidad de desarrolladores soportándolo.

Creado para ayudar a los administradores a tener siempre el control de qué está pasando en la red y conocer los problemas que ocurren en la infraestructura antes de que los usuarios de la misma los perciban, para así no sólo poder tomar la iniciativa, sino asumir la responsabilidad de hacer que los servicios sean restablecidos de manera pronta; decidir en cada momento lo que queremos hacer y cómo lo vamos a hacer, debido a que este *software* nos permite obtener datos, interpretarlos y tomar decisiones con base a ello.

Nagios está constituido por un núcleo que construye la interfaz de usuario y por *plugins*, los cuales representan los ojos y oídos de Nagios, que se encargan de recopilar información (bajo demanda). Los mismos pueden estar programados en diversos lenguajes, ya que Nagios es independiente del lenguaje en el cual se desarrolle el *plugin* y solo procesa los datos recibidos de este para la posterior elaboración y envío de notificaciones a los encargados de la administración del sistema en cuestión.

Algunas de las actividades que se pueden realizar con Nagios son:

- Monitorización de los servicios.
- Monitorización de los *hosts*.
- Notificaciones a los contactos cuando ocurran incidencias en servicios o *hosts*, así como cuando son resueltos.
- Posibilidad de definir manejadores de eventos que se ejecuten al ocurrir un evento en un servicio o *host* para resoluciones proactivas de problemas.

Algunas personas que pueden usar Nagios son[23]:

- Administradores de redes con alto conocimiento técnico.
- Operadores con mínimo conocimiento técnico de la situación, o conocimiento puntual de algún servicio que puedan avisar y ayudar a determinar posibles causas de los problemas.
- Personal con conocimiento medio/avanzado en redes .

Como impacta el uso de Nagios[23]:

- Antelación de problemas.
- Reporte y aviso de incidentes.
- Agilidad en su tratamiento.
- Requerimientos.
- Plan de acción bien diagramado.

## 3.4. CGI (*Common Gateway Interface*)

La Interfaz Común de Puerta de Enlace (CGI) es el medio de comunicación que emplea un servidor *web* para enviar una información en ambos sentidos, entre el explorador *web* y el programa de cómputo en el servidor.

CGI es la interfaz del servidor *web* usando el Protocolo de Transferencia de Hipertexto (HTTP por sus siglas en inglés, *Hyper Text Transfer Protocol*) con los demás recursos del *host* del servidor.

CGI no es en realidad un lenguaje o un protocolo, en el sentido más estricto del término. En realidad sólo es un conjunto de variables y convenciones, nombradas comúnmente, para pasar información en ambos sentidos entre el servidor y el cliente.[30]

Para utilizar CGI's se necesita un lenguaje de programación que este soportado por el sistema operativo del servidor que en el caso de la herramienta Nagios utiliza Perl y PHP como lenguajes de programación para los CGI's .

# Capítulo 4

# Conceptos Básicos de Seguridad Informática

En la actualidad las actividades cotidianas de las organizaciones, así como la de los propios usuarios, requieren del correcto funcionamiento de los sistemas y redes informáticas que los soportan.

Para esto podemos definir la Seguridad Informática como "*cualquier medida que impida la ejecución de operaciones no autorizadas sobre un sistema o red informática, cuyos efectos puedan conllevar daños sobre la información, comprometer su confidencialidad, autenticidad o integridad, disminuir el rendimiento de los equipos o bloquear el acceso de usuarios autorizados al sistema*"[25].

También es necesario considerar otros aspectos relacionados a la seguridad informática como son:

- Cumplimiento de las disposiciones legales aplicables a cada sector o tipo de organización, dependiendo del marco legal de cada país.
- Control en el acceso a los servicios ofrecidos y la información guardada por un sistema.
- Control en el acceso y utilización de archivos protegidos por la ley: contenidos digitales con derechos de autor, archivos con datos de carácter personal, etc.
- Identificación de los autores de la información o de los mensajes.
- Registro del uso de los servicios de un sistema informático, etc.

En la norma ISO/IEC 17799 se define la seguridad de la información como la preservación de su confidencialidad, su integridad y su disponibilidad (medidas conocidas por su acrónimo "CIA" en inglés: "*Confidenciality, Integrity, Availability*").[25]


Figura 4.1: Seguridad de la Información según la norma ISO/IEC 17799

La norma ISO 7498 define la seguridad informática como "*...una serie de mecanismos que minimizan la vulnerabilidad de bienes y recursos en una organización*". [25]

Así mismo, podemos mencionar otra definición propuesta por el INFOSEC Glossary 2000: "*Seguridad informática son las medidas y controles que aseguran la confidencialidad, integridad y disponibilidad de los activos de los sistemas, incluyendo hardware, software, firmware y aquella información que procesan, almacenan y comunican*".[25]

Se debe tener en cuenta que la seguridad de un sistema dependerá de diversos factores, entre los que se podrían destacar los siguientes:

- La sensibilización de los directivos y responsables de la organización, que deben ser conscientes de la necesidad de destinar recursos a esta función.
- Los conocimientos, capacidades e implicación de los responsables del sistema: dominio de la tecnología utilizada en el sistema y conocimiento sobre las posibles amenazas y los tipos de ataques.
- La correcta instalación, configuración y mantenimiento de los equipos.
- La limitación en la asignación de los permisos y privilegios de los usuarios.
- El soporte de los fabricantes de *hardware* y *software*, con la publicación de parches y actualizaciones de sus productos que permitan corregir los fallos y problemas relacionados con la seguridad.
- Contemplar no sólo la seguridad frente a las amenazas del exterior, sino también de las amenazas procedentes del interior de la organización.

## 4.1. Objetivos de la Seguridad Informática

Entre los principales objetivos de la seguridad informática se podrían destacar los siguientes:

- Minimizar y gestionar los riesgos y detectar los posibles problemas y amenazas a la seguridad.

- Garantizar la adecuada utilización de los recursos y de las aplicaciones del sistema.
- Limitar las pérdidas y conseguir la adecuada recuperación del sistema en caso de un incidente de seguridad.
- Cumplir con el marco legal y con los requisitos impuestos por los clientes en sus contratos.

Para cumplir con estos objetivos una organización debe contemplar cuatro planos de actuación:

- Técnico: tanto a nivel físico como a nivel lógico.
- Legal: algunos países obligan por Ley a que en determinados sectores se implanten una serie de medidas de seguridad.
- Humano: sensibilización y formación de empleados y directivos, definición de funciones y obligaciones del personal.
- Organizativo: definición e implantación de políticas de seguridad, planes, normas, procedimientos y buenas prácticas de actuación.

## 4.2. Consecuencias de la Falta de Seguridad

El desarrollo de las actividades de muchas organizaciones depende de los datos y de la información registrada en sus sistemas, así como del soporte adecuado de las TICs para facilitar su almacenamiento, procesamiento y distribución.

Por todo ello, es necesario dar a conocer a los directivos la importancia de valorar y proteger la información de la organización.

Resulta de especial importancia poner en conocimiento de los directivos cual es el costo e impacto de los incidentes de seguridad en términos económicos, y no a través de confusos informes plagados de tecnicismos, defendiendo la idea de que la inversión en seguridad informática sería comparable a la contratación de un seguro contra robos, contra incendios etc. (gasto no productivo pero necesario para poder mantener la actividad de la organización si se produce algún incidente).

Siempre se debe tener en cuenta que el costo de las medidas adoptadas por la organización ha de ser menor que el valor de los activos que hay que proteger. Para ello, es necesario realizar un análisis de la relación costo-beneficio de cada medida de seguridad que se desea implantar, ya que no todas las organizaciones necesitan las mismas medidas de seguridad.

A la hora de analizar las posibles consecuencias de la ausencia o de una deficiente seguridad informática, el impacto total para una organización puede resultar bastante difícil de evaluar, ya que además de los posibles daños ocasionados a la información guardada y a los equipos y dispositivos de red, deberíamos tener en cuenta otros importantes perjuicios para la organización como son:

- Horas de trabajo invertidas en las reparaciones y reconfiguración de los equipos y redes.
- Pérdidas ocasionadas por la indisponibilidad de diversas aplicaciones y servicios informáticos: costo de oportunidad por no poder utilizar estos recursos.
- Robo de información confidencial y su posible revelación a terceros no autorizados: fórmulas, diseños de productos, estrategias comerciales, programas informáticos.

- Filtración de datos personales de usuarios registrados en el sistema: empleados, clientes, provedores, candidatos, de empleo o contactos comerciales, con las consecuencias que se derivan del incumplimiento de la legislación en materia de protección de datos personales.
- Posible impacto en la imagen de la empresa ante terceros: pérdida de credibilidad en los mercados, daño a la reputación de la empresa, pérdida de confianza por parte de los clientes y los provedores etc.
- Retrasos en los procesos de producción, pérdida de pedidos, impacto en la calidad del servicio, pérdida de oportunidades de negocio.
- Posibles daños a la salud de las personas, con pérdidas de vidas humanas en los casos más graves.
- Pago de indemnización por daños y perjuicios a terceros, teniendo que afrontar además posibles responsabilidades legales y la imposición de sanciones administrativas.

Es necesario contemplar otros posibles problemas que se podrían derivar si un tercero toma el control de algunos de los equipos de alguna organización:

- Utilización de los equipos y redes de una organización para llevar a cabo ataques contra otras redes de otras empresas u organizaciones.
- Almacenamiento de contenido ilegal en los equipos de la organización, con la posibilidad de instalar un servidor FTP sin la autorización del legítimo propietario de éstos.
- Utilización de los equipos de una organización para realizar envíos masivos de correo no solicitado (*spam*), etc.

Es claro que una inadecuada gestión de la seguridad provocará, tarde o temprano, una desventaja competitiva.

La implantación de determinadas medidas de seguridad puede resultar incómoda para muchos usuarios del sistema y, por ello, resulta fundamental contemplar la adecuada formación y sensibilización de los usuarios para que estas medidas se puedan implantar de forma efectiva.

## 4.3. El Factor Humano en la Seguridad Informática

La implantación de unas adecuadas medidas de seguridad informática exige contemplar aspectos técnicos, organizativos y legales. No obstante, en muchas ocasiones se presta muy poca atención a la importancia del factor humano en la seguridad informática.

Las personas representan el eslabón más débil dentro de la seguridad informática: a diferencia de las computadoras, las personas no pueden seguir las instrucciones tal y como fueron dictadas. Además, pueden llevar a cabo acciones que provoquen un agujero de seguridad en la red de la organización: instalación de *software* malicioso (por ejemplo, un *spyware*) en su computadora, revelación de información sensible a terceros, etc.

Así en palabras de Kevin Mitnick, uno de los *hackers* más famosos de la historia, "*usted puede tener la mejor tecnología, firewalls, sistemas de detección de ataques, dispositivos biométricos... Lo único que se necesita es una llamada a un empleado desprevenido y acceden al sistema sin más. Tienen todo en sus manos.*"[25]

El propio experto en criptografía y seguridad Bruce Scheneier llegaba a afirmar en uno de sus últimos libros, *Secrets and Lies* (Verdades y Mentiras) que "*... si piensas que la tecnología puede resolver tus problemas de seguridad, entonces no entiendes el problema y no entiendes la tecnología*".[25]

## 4.4. SSH

*Secure Shell* provee un protocolo abierto para la comunicación segura en las redes de comunicación, es poco complejo y está basado en soluciones *VPN (Virtual Private Network)*. El cliente/servidor de *Secure Shell* proporciona una terminal de comandos, transferencia de archivos y servicios para aplicaciones TCP/IP por medio de túneles. Las conexiones por SSH proveen conexiones por autenticación de alta seguridad, encriptación e integridad de los datos en redes de comunicaciones.

La implementación de SSH proporciona las siguientes características: una terminal de comandos segura, transferencia de archivos segura y acceso remoto vía túnel seguro. Las aplicaciones cliente/servidor de *Secure Shell* están disponibles para los sistemas operativos más populares.

*Secure Shell* ofrece una buena solución para el problema de envío de datos seguro a través de una red pública y para la presente tesis es utilizado para acceder de manera remota al servidor Nagios en caso de requerirse configuraciones en la herramienta Nagios, ya que solo requiere una conexión a internet y un cliente SSH instalado en la máquina remota.

### 4.4.1. Terminal de Comandos Segura

Terminales de comandos están disponibles en distribuciones Linux, Unix y Windows (*MS-DOS*) y tienen la capacidad de ejecutar programas y otros comandos, usualmente con salida de caracteres. Una terminal de comandos segura o de autenticación remota permite editar archivos, ver el contenido de directorios y accesar a aplicaciones. Los Administradores de Sistemas y Redes pueden ejecutar *scripts* remotamente, pueden iniciar, ver o parar servicios y procesos, crear cuentas de usuario, cambiar permisos a los archivos y/o directorios y más. Cualquier cosa que pueda hacer una terminal de comandos en una máquina puede ser hecha de forma segura remotamente.

### 4.4.2. Transferencia de Archivos Segura

El Protocolo de Transferencia de Archivos Segura (*SFTP*, *Secure File Transfer Protocol*) es un subsistema del protocolo *Secure Shell*. *SFTP* tiene muchas ventajas sobre el protocolo *FTP* no seguro. Primero, *SFTP* encripta el *username/password* (nombre de usuario/contraseña) y los datos comienzan a ser transferidos. Segundo, usa el mismo puerto que el servidor *Secure Shell*, eliminando la necesidad de abrir otro puerto en un *firewall* o *router*. Algunos de los usos más comunes pueden incluir la subida y descarga con el objetivo de proporcionar un mecanismo de administración remota seguro.

### 4.4.3. Autenticación de Usuario

La autenticación se refiere a la identidad del usuario, lo que significa que el sistema verifica que el acceso solo sea proporcionado a los usuarios con permiso y denegado a todos los demás.

Diversos métodos de autenticación son usados actualmente. Muchas de las implementaciones de *Secure Shell* incluyen contraseña y autenticación de clave pública.

#### 4.4.3.1. Autenticación por Contraseña

Una contraseña en combinación con un nombre de usuario, es comúnmente una manera de avisar a otra computadora que es un usuario permitido quien intenta acceder. Si el nombre de usuario y contraseña obtienen la autenticación, entonces el nombre de usuario y contraseña son almacenados en el sistema remoto, en ese momento está registrado y se le permite el acceso. *Secure Shell* encripta estos parámetros antes de la transmisión de los mismos y permite un envío seguro hasta llegar a su destino.

Aunque las contraseñas son convenientes, si no tiene una configuración adicional para el acceso a sus cuentas de usuario, esto puede ser una vulnerabilidad ya que alguien podría adivinar la contraseña y obtener el acceso al servidor. Para combatir estas vulnerabilidades es necesario combinar o reemplazar la autenticación por contraseña con otros métodos como la clave pública.

#### 4.4.3.2. Autenticación por Clave Pública

La autenticación por clave pública, es uno de los métodos más seguros para autenticar usando *Secure Shell*. La autenticación por clave pública usa un par de claves (una pública y una privada). Cada clave es de 1024 y 2048 bits de longitud respectivamente[4].

El par de claves pública-privada son creadas utilizando un generador de claves, ambas, generadas al mismo tiempo.

Para acceder a una cuenta de un servidor por SSH usando la autenticación por clave pública, debe ser cargada una copia de la clave pública del cliente en el servidor. Cuando el cliente intenta una conexión al servidor, debe proporcionar la *frase secreta* y la contraparte a la clave pública, es decir, la clave privada para que el acceso se permita.

La clave privada tiene una "*frase de contraseña*" asociada con la misma por lo que incluso si la clave privada es robada, el atacante debe adivinar la *frase secreta* para acceder. La autenticación de clave pública no permite el acceso hasta que el cliente pruebe que tiene la "*frase de contraseña*" de la clave privada.

# Desarrollo

*No se graban tanto mil palabras como un solo hecho.*

Henrik Johan Ibsen (1828-1906) Dramaturgo noruego.

# Capítulo 5

# Servidor de Monitorización Nagios

En la presente tesis el escenario de red es el siguiente:

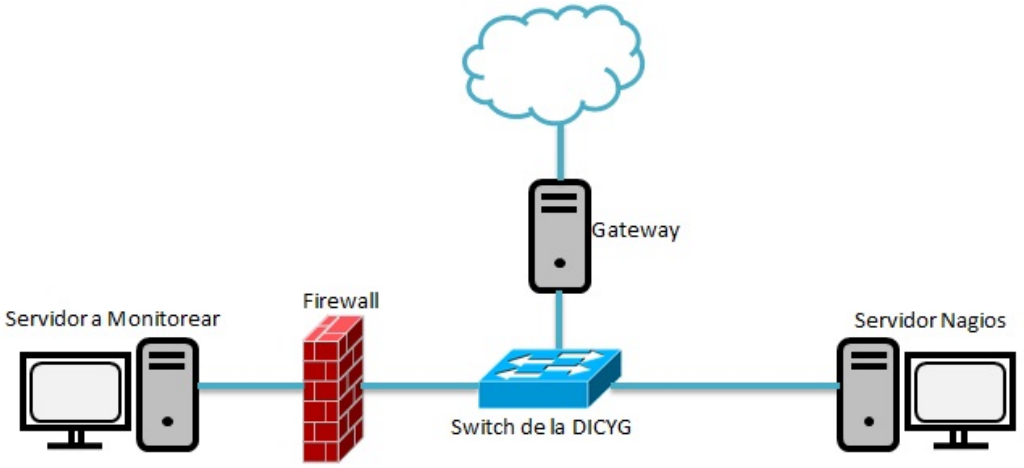


Figura 5.1: Escenario: Servidor Nagios y Servidor a Monitorizar

## 5.1. Nagios

Nagios es un sistema de monitorización de equipos y de servicios de red que tiene como objetivo ayudar al administrador a tener el control de lo que está pasando en la red.

### 5.1.1. Dependencias

Para una correcta instalación de Nagios, con todas sus características es necesario tener instalados ciertos paquetes de *software* en el sistema, la instalación puede variar según la distribución de Linux que se elija, si están empaquetados, o si hay que compilar e instalar manualmente.

Para la presente tesis la instalación de Nagios se realizó de forma manual en un Debian 6 y además se instalaron las siguientes dependencias:

- Apache 2
- Compilador *gcc*
- Perl
- PHP

- RRD Tool

Y se usarón los siguientes paquetes:

- Nagios Core 3.4.3
- Plugins Nagios 1.4.16
- Plugins Nagios 1.4.5
- PNP4Nagios 0.6.21

### 5.1.2. Estructura de Archivos

Una vez que se compila e instala el paquete Nagios el resultado es una estrucutra de directorios como la siguiente:

- **bin:** Aquí se almacenan los binarios ejecutables.
- **etc:** Contiene los archivos de configuración de Nagios.
- **libexec:** Se almacenan los plugins que efectuarán las comprobaciones.
- **sbin:** Dentro de este directorio se mantienen los ejecutables CGI de la interfaz *web*.
- **share:** Organiza el contenido *web* a mostrar, íconos, html, php etc.
- **var:** Guarda los datos de ejecución de la monitorización, estado de servicios, estado de *hosts*, y *logs*.

El siguiente cuadro detalla más la estructura de directorios creada por Nagios:

| Directorio | Descripción |
|---|---|
| bin | Dentro de este directorio se encuentran los ejecutables principales, como el binario nagios que se ejecuta como proceso en segundo plano. |
| etc | Este directorio guarda la configuración de Nagios, los hosts y servicios a comprobar, comandos de ejecución, contactos de notificación, intervalos de monitorización. |
| libexec | Este directorio contiene los archivos ejecutables de los plugins que efectúan las comprobaciones, SNMP, SAP, Oracle, SSH, que pueden ser binarios, scripts en Perl, PHP, Shell, Java, etc. |
| sbin | Este directorio almacena los ejecutables cgi que se ejecutarán para la visualización por web de la consola Nagios. |
| share | Este directorio almacena el contenido web, imágenes, logos, los complementos adicionales como PNP y los datos que necesita para que funcione. |
| var | Este directorio almacena los datos internos de Nagios, estadísticas de las comprobaciones, información de ejecución, archivos de sockets, registros de logs. |

Tabla 5.1: Estructura de Archivos para Nagios

### 5.1.3. Archivos de Configuración Nagios

**Directorio: /etc**

**Archivo:cgi.cfg**

A continuación se presentan algunas de las directivas del archivo *cgi.cfg* con su descripción correspondiente:

- La siguiente directiva define la ubicación del archivo de configuracion principal de Nagios

```
main_config_file=/usr/local/nagios/etc/nagios.cfg
```

- Ruta donde se ubican los archivos a mostrar vía *web*

```
physical_html_path=/usr/local/nagios/share
```

- Ruta de la url a donde se ubica Nagios desde el navegador

```
url_html_path=/nagios
```

- Usar autenticación para acceder a Nagios

```
use_authentication=1
```

- Tener usuario autenticado por *default* (no recomendado por cuestiones de seguridad, dejar comentado)

```
#default_user_name=guest
```

- Usuarios con acceso permitido para ver la información de objetos (separados por comas)

```
authorized_for_system_information=nagiosadmin
```

- Usuarios con acceso permitido para ver la información de configuración (separados por comas)

```
authorized_for_configuration_information=nagiosadmin
```

- Usuarios con acceso permitido para la ejecución de comandos en Nagios (separados por comas)

```
authorized_for_system_commands=nagiosadmin
```

- Usuarios permitidos a ver información de *hosts* y servicios (separados por comas)

```
authorized_for_all_services=nagiosadmin
authorized_for_all_hosts=nagiosadmin
```

- Usuarios permitidos para ejecutar comandos sobre *hosts* y servicios (separados por comas)

```
authorized_for_all_service_commands=nagiosadmin
authorized_for_all_host_commands=nagiosadmin
```

- Tasa de actualización para la interfaz *web* en segundos

```
refresh_rate=90
```

**Archivo:htpasswd.users**

Archivo con contraseñas encriptadas de los usuarios que se autentificarán por HTTP.

**Archivo:nagios.cfg**

Archivo de configuración principal de Nagios, en este archivo se especifican los directorios de trabajo y se incluyen las ubicaciones de los archivos de configuración extra a utilizar por Nagios con diversos parámetros como se muestran en la siguiente tabla:

| | |
|---|---|
| log_file | Se especifica el archivo de *log* a utilizar por Nagios. |
| cfg_file | Se especifica un archivo de configuración extra a incluir en la ejecución de Nagios. |
| cfg_dir | Se especifica un directorio con archivos de configuración extra a incluir recursivamente en la ejecución de Nagios. |
| log_archive_path | *Path* donde se alojarán los archivos *log*. |
| use_syslog | Integración con *syslog*. |

Tabla 5.2: Algunas directivas que se pueden configurar en el archivo *nagios.cfg*

Algunas directivas deben ser creadas en el archivo de configuración *nagios.cfg* para especificar donde se encuentran las definiciones de servicios, grupos, contactos etc.

Como ejemplo:

```
# Directorio con la configuración de grupos de Hosts de los Servidores
cfg_dir=/usr/local/nagios/etc/gruposhosts

# Directorio con la configuración de grupos de servicios de los Servidores
cfg_dir=/usr/local/nagios/etc/gruposservicios

# Directorio con la configuración de contactos
cfg_dir=/usr/local/nagios/etc/contactos

# Directorio con la configuración de grupos de contacto
cfg_dir=/usr/local/nagios/etc/gruposcontactos

# Directorio con la configuración de servicios
cfg_dir=/usr/local/nagios/etc/servicios

# Directorio con la configuración de los comandos
cfg_dir=/usr/local/nagios/etc/comandos

# Directorio con la configuración de los equipos a monitorizar
cfg_dir=/usr/local/nagios/etc/servidores
```

Con la directiva *cfg_dir* indicamos a Nagios que tome como configuración los archivos con extensión "*cfg*" encontrados en el directorio especificado.

**Archivo:resource.cfg**

Archivo de configuración donde se definen *macros* de ejecución.

**Directorio: /objects**

Directorio de archivos generales de configuración.

**Archivo: commands.cfg**

Definición de comandos de ejecución por *default*, con los alias que se quieren usar.

**Archivo: contacts.cfg**

Definición de contactos a los cuales se les realizarán las notificaciones.

**Archivo: localhost.cfg**

Plantilla inicial para la comprobación del *localhost*.

**Archivo: templates.cfg**

Plantillas generales de *hosts*, contactos, y servicios.

**Archivo: timeperiods.cfg**

Plantilla inicial para definir periodos de comprobación, aquí se definen los rangos de tiempo donde son válidos el envío de alertas y las comprobaciones de servicios que están funcionando.

**Directorio: /var/rw**

En este directorio se encuentra un archivo especial de *socket* que realiza la comunicación de los comandos y órdenes de la interfaz *web* hacia Nagios, como cambiar horarios de comprobación, deshabilitar notificaciones etc.

El archivo que aquí se encuentra (*nagios.cmd*) debe tener permisos de escritura y lectura por el propietario y el grupo de pertenencia *nagios:nagcmd* (660), *nagcmd* es un grupo especial en el cual vamos a incluir al usuario que ejecuta el servidor *web* (ej. en apache sobre Debian *www-data*), y así poder enviar órdenes desde la interfaz *web* CGI. Esta es una característica avanzada de Nagios que permite vía *web* la ejecución de ciertas tareas más allá del propio conjunto de CGI's que vienen por defecto de instalación, como por ejemplo la caída o el reinicio del propio Nagios, etc. Para poder ejecutar este tipo de comandos es necesario también configurar el sistema de una forma un tanto especial. Configurando Nagios de este modo, se está permitiendo que desde la *web* se pueda activar o desactivar opciones que en principio sólo estaban disponibles desde la consola del sistema. Para configurar Nagios de esta forma, hay que editar el archvio principal *nagios.cfg* y añadir (o modificar si ya existen) las siguientes líneas:

```
check_external_commands=1
command_check_interval=-1
command_file=/usr/local/nagios/var/rw/nagios.cmd
```

Lo que hará que Nagios active la comprobación para buscar comandos externos, con tanta frecuencia como sea posible por el sistema y buscará los comandos en el archivo *nagios.cmd*.

A continuación se presenta la relación de los archivos de configuración de Nagios:

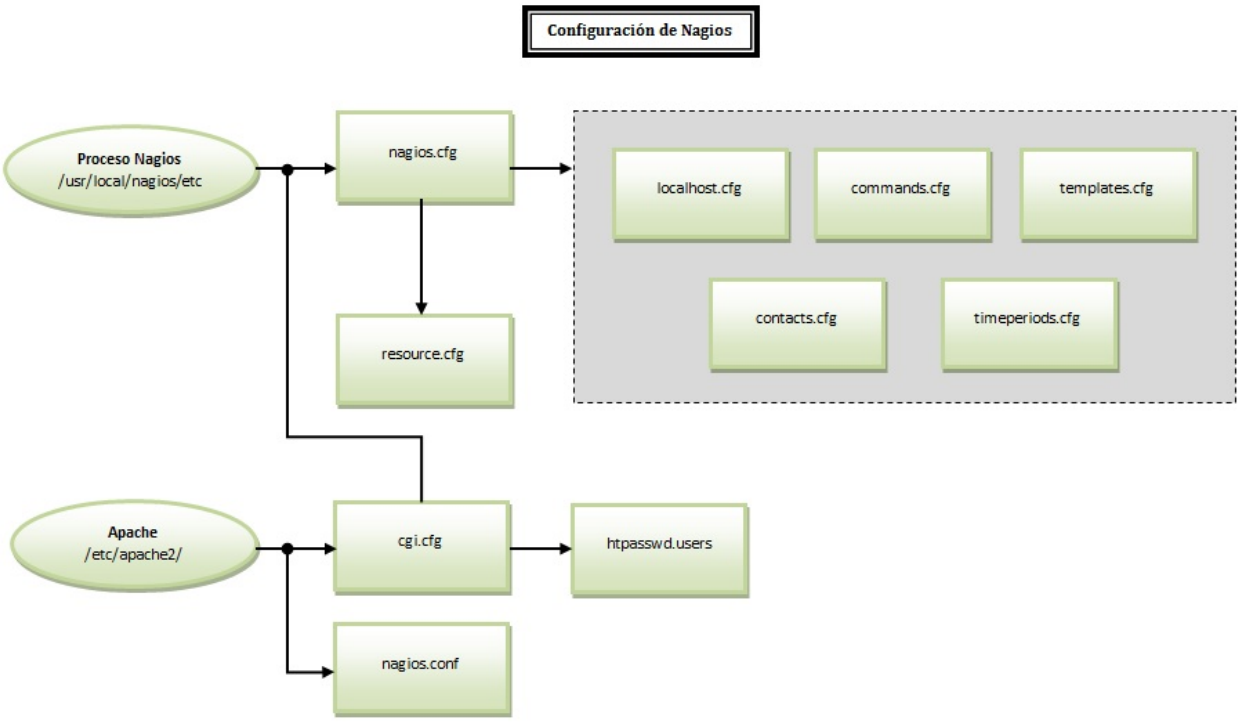


Figura 5.2: Como se Relacionan los Archivos de Configuración de Nagios Core 3.4.3

### 5.1.4. Agregar un *Host*

| Símbolo | Significado |
|---|---|
| ✎ | Las directivas con un (*) previo son obligatorias en la definición de los elementos definidos en los subsubtemas 5.1.4 al 5.1.10. Cabe mencionar que existe una gran variedad de directivas que se pueden utilizar en las diferentes definiciones, si el lector desea más información se recomienda que consulte la bibliografía recomendada.[19]<br><br>**Importante:** El asterisco (*) no se escribe en las definiciones. |

Para configurar un *host* previamente instalado y configurado, para su posterior monitorización se debe crear una entrada en la configuracion de Nagios.

Un ejemplo de definición de un *hosts* puede ser el siguiente:

```
    define host{
    use                     servidores        #Nombre del template a utilizar
*       host_name               dicyg             #Nombre del host
*       alias                   DICYG SERVER      #Alias del host
*       address                 192.168.0.1       #Dirección IP del host
*       max_check_attempts      4                 #Número de comprobaciones antes de enviar una
    notificación
*       check_period            24x7              #Periodo de Comprobación
*       contacts                ulises            #Contactos
*       contact_groups          admins             #Grupo de Contactos
```

```
*       notification_interval   120             #Periodo de tiempo entre notificaciones
*       notification_period     24x7            #Periodo de Notificación
      notification_options    d,u,r           #Estados en los cuales se enviará una
          notificación
      }
```

### 5.1.5. Agregar Servicios

Una definición de servicio es usada para identificar un determinado servicio que se ejecuta en un *host* determinado. El término "servicio" se usa de manera muy informal. Lo cual significa que puede ser un servicio que se ejecute en un *host* (MySQL, HTTP, etc.) o algún otro tipo de métrica asociada al *host* (respuesta a un *ping*, número de usuarios autenticados, espacio libre del disco duro, etc.).*[19]*

Un ejemplo de la definición de un servicio puede ser la siguiente:

```
      define service{
          use                     servicio-general     #Nombre de la plantilla a utilizar
*     host_name               dicyg               #Nombre del host que tiene el servicio
*     service_description     Servicio Ping       #Descripción del servicio
*     check_command           check_ping          #Nombre del comando
*     max_check_attempts      4                   #Número de comprobaciones antes de enviar una
      notificación
*     check_interval          5                   #Periodo de tiempo entre comprobaciones de
     estado
*     retry_interval          3                   #Periodo de tiempo entre comprobaciones
     cuando hay un cambio de estado a no-OK
*     check_period            24x7                #Periodo de comprobación
*     notification_interval   120                 #Periodo de tiempo entre notificaciones
*     notification_period     24x7                #Periodo de Notificación
      notification_options    w,u,c,r             #Estados en los cuales se enviaran
          notificaciones
*     contacts                ulises              #Contactos a los que se notificara
*     contact_groups          admins               #Nombre del grupo de contactos al que se
     notificara
      }
```

### 5.1.6. Agregar un Grupo de Servicios

Una definición de un grupo de servicios se usa para agrupar uno o más servicios con el objetivo de simplificar la configuración.

```
    define servicegroup{
*       servicegroup_name   DICYG-UC                                #Nombre del grupo
*       alias               Servidores de la Unidad de Cómputo #Alias del grupo
        members             dicyg                                   #Miembros del grupo
}
```

### 5.1.7. Agregar Comandos

En Nagios los encargados de recabar los datos de la monitorización y de mostrar alertas de todas las tareas, son los comandos.

Los mismos se dividen en comandos de rendimiento y en comandos de comprobación.

Los comandos de comprobación recolectan los datos de los equipos a monitorizar, como consumo de CPU, Memoria, Disco, procesos corriendo, puertos abiertos etc., es decir todos los datos necesarios sobre la monitorización.

Los comandos de rendimiento se utilizan cuando hay que guardar ciertos datos o enviarlos a algún *host* externo con información de algún servicio.

Una entrada en un archivo de configuración de comandos puede ser como la siguiente:

```
    define command{
*     command_name    check_ping                          #Nombre del comando
*     command_line    $USER1$/check_ping -H $HOSTADDRESS$ #Ubicación del archvio junto con los
     argumentos
    }
```

Lo que se define entre signos $ son variables internas de Nagios, llamadas macros, las más comunes son:

**$USER1$:** Contiene datos del *path* de ejecución de los plugins de Nagios

**$HOSTADDRESS$:** Tiene la IP de *hosts* desde el cual se está corriendo el servicio

**$ARG1$ $ARG2$ $ARG3$ $ARG4$:** Son los números en orden de argumentos que recibe el comando a ejecutar.

Se pueden definir macros personalizadas configurando las variables en el archivo *resource.cfg*.

### 5.1.8. Agregar Contactos

Para recibir las notificaciones generadas por Nagios es necesario definir contactos que estén incluidos en diferentes grupos de contactos, una configuración simple para un contacto se ve como la siguiente entrada:

```
define contact{
*   contact_name                    Ulises                          #Nombre del contacto
    alias                           Administrador Nagios 1er Nivel #Alias del contacto
    contactgroups                   admins                          #Grupo al que pertenece el
        contacto
*   host_notifications_enabled      1                              #Habilitación de
     notificaciones para host
*   service_notifications_enabled       1                              ##Habilitación de
     notificaciones para servicio
*   service_notification_period     24x7                            #Periodo de notificación
     para servicio
*   host_notification_period        24x7                            #Periodo de notificación
     para host
*   service_notification_options    w,u,c,r                         #Estados del servicio en
     los cuales se enviara notificación
*   host_notification_options       d,u,r                           #Estados del host en los
     cuales se enviara notificación
*   service_notification_commands   notify-by-email                 #Comando de notificación
     para servicios
*   host_notification_commands      host-notify-by-email            #Comando de notificación
     para hosts
    email                           uli.dicyg.fi@unam.mx              #Correo electrónico del
        contacto
}
```

### 5.1.9. Agregar Grupos de Contactos

Para que Nagios envíe notificaciones sobre el estado de los servicios es necesario definir grupos de contactos, y dentro de ellos estarán definidos los miembros del grupo de contacto.

A continuación podemos ver una configuración típica de un grupo de contactos:

```
define contactgroup{
*    contactgroup_name     admins                                    #Nombre del grupo contactos
*    alias                 Administradores de la Red en la DICYG     #Descripción
     members               admin-Zeus,admin-Ulises                   #Miembros del grupo
}
```

### 5.1.10. Agregar *Templates*

Nagios clasifica las definiciones como objetos y sus características pueden ser heredadas por otros objetos.

Se puede definir un *template* (en español plantilla) y heredar sus características a otros objetos como una base, así sólo será necesario añadir las características que son diferentes o que falten.

Un ejemplo de un template podría ser el siguiente:

```
define host{
*       name                    Generic-Host        #Nombre de la Plantilla
*       register                0                   #Valor de registro para que Nagios lo
    interprete como un template
        check_command           check-host-alive    #Comando a comprobar
        max_check_attempts      4                   #Número de comprobaciones antes de enviar una
            notificación
        check_period            24x7                #Periodo de comprobación
        contact_groups          localadmins         #Grupo de contactos a notificar
        notification_interval   120                 #Periodo de tiempo entre notificaciones
        notification_period     24x7                #Periodo de notificación
        notification_options    d,u,r             #Estados en los cuales se enviarán notificaciones
    }
```

Al *template* se le asigna primero un nombre de manera que se pueda hacer referencia más adelante. El siguiente parámetro, el *register* 0, impide que Nagios interprete este *template* como un verdadero *host*. Si el parámetro *register* no se especificara Nagios mandaría un mensaje de error notificando que faltan parámetros que son obligatorios para esta definición, por ejemplo:

```
Error: Host name is NULL
```

Todos los demás parámetros implican configuraciones que se aplicarán a todas las definiciones que dependan de *Generic-Host*.

Un uso del *template* realizado podría ser:

```
define host{
host_name     dicyg01
use           Generic-Host
alias         Servidor Sistema de Titulación
address       192.168.0.2
}
```

De esta manera, sólo será necesario completar los parámetros que varían entre los dos *hosts*.

Los parámetros definidos en los *templates* también pueden aparecer en las definiciones de *host*. En este caso la definición en el *host* tiene prioridad y por lo tanto se sobrescribe el valor de la *template*.

Los *templates* creados de esta manera se pueden utilizar generalmente para todos los tipos de objetos.

### 5.1.11. Agregar Tiempos de Comprobación

Un periodo de tiempo es la definición de un rango de tiempo que es considerado válido para las notificaciones y comprobaciones. Soporta varios tipos de definiciones como: días especificos de la semana, días de algun mes, días de algunos meses y fechas de calendario.

- Monitorización las 24 horas.

```
define timeperiod{
   timeperiod_name  24x7
   alias            24 Horas al día, 7 días a la semana
   sunday           00:00-24:00
   monday           00:00-24:00
   tuesday          00:00-24:00
   wednesday        00:00-24:00
   thursday         00:00-24:00
   friday           00:00-24:00
   saturday         00:00-24:00
}
```

- Monitorización en las horas laborales lunes a viernes de 7a.m. a 10p.m.

```
define timeperiod{
   timeperiod_name     horaslaborales
   alias               Horas Laborales
   monday              07:00-22:00
   tuesday             07:00-22:00
   wednesday           07:00-22:00
   thursday            07:00-22:00
   friday              07:00-22:00
}
```

- En ciertos días excluir la monitorización de servicios.

```
define timeperiod{
   name               vacaciones
   timeperiod_name    vacaciones
   alias              Vacaciones
   january 1          00:00-00:00 ; Año Nuevo
   september 16       00:00-00:00 ; Día de Independencia
   monday 1 may       00:00-00:00 ; Día del Trabajo
   december 25        00:00-00:00 ; Navidad
}
```

Definimos un periodo de tiempo que realice la monitorización las 24 horas del día los 7 días de la semana, pero que incluya las exepciones anteriormente definidas.

```
define timeperiod{
   timeperiod_name    24x7_con_vacaciones
   alias              24x7 Considerando Vacaciones
   use                vacaciones ; Agregar exepciones
   sunday             00:00-24:00
   monday             00:00-24:00
   tuesday            00:00-24:00
   wednesday          00:00-24:00
   thursday           00:00-24:00
   friday             00:00-24:00
   saturday           00:00-24:00
}
```

### 5.1.12. Agregar *Event Handlers*

Los *event handlers* son comandos que se ejecutan cada vez que se produce un cambio de estado en un host o servicio y están asociados a un *script* o ejecutable.

Los *event handlers* en Nagios se pueden usar para solucionar problemas de forma proactiva antes de notificar al administrador de la red. Algunos usos para los *event handlers* son:

- Reinicio de un servicio caído
- Reiniciar un *host* [1]

Los *event handlers* se ejecutan cuando un servicio o *host*:

- Está en un estado de problema SOFT.
- Entra en un estado de problema HARD.
- Se recupera de un estado problema SOFT o HARD.

Se pueden definir diferentes tipos de *event handlers* para los *hosts* y cambios de estado:

- *Event handler* global de *hosts*.
- *Event handler* global de servicios.
- *Event handler* en *host* específicos.
- *Event handler* en servicios específicos.

Los *event handlers* se habilitan en el archivo *nagios.cfg* modificando el valor de la directiva *enable_event_handlers* a 1.

Para los *scripts* que se usen en los *event handlers* deberán poder ejecutarse desde una terminal y como mínimo usar los *macros* siguientes:

- Para los servicios : *$SERVICESTATE$ , $SERVICESTATETYPE$ , $SERVICEATTEMPT$*
- Para los *hosts*: *$HOSTSTATE$ , $HOSTSTATETYPE$ , $HOSTATTEMPT$*

Los *event handlers* se ejecutan con los mismos permisos que el usuario con el que Nagios funciona. Esto puede suponer un problema si lo que se quiere es reiniciar un servicio del sistema, solo teniendo privilegios como los del usuario *root* se puede realizar este tipo de tareas.

Lo ideal es evaluar los tipos de *event handlers* a implementar y conceder sólo los permisos necesarios para el usuario *nagios* para ejecutar los comandos necesarios en el sistema. Es posible solucionar esto usando *sudo* en la ejecución de los comandos.

Una definición de un *event handler* en un servicio podría ser la siguiente:

```
define service{
        host_name                       somehost
        service_description             HTTP
        max_check_attempts              4
        event_handler                   restart-httpd
        ...
    }
```

[1] Considerar las consecuencias de implementar reinicios automáticos en un *host*.[7]

Esto quiere decir que, al servicio que tiene como *host* a *somehost* y en la descripción de servicio a HTTP, el servicio se compruebe 4 veces antes de que se considere como problema real.

Una vez que el servicio ha sido definido con un *event handler*, hay que definir ese *event handler* como un comando. Una definición de comando podría ser *restart-httpd* y se definiría como sigue:

```
define command{
        command_name    restart-httpd
        command_line    /usr/local/nagios/libexec/eventhandlers/restart-httpd  $SERVICESTATE$
            $SERVICESTATETYPE$ $SERVICEATTEMPT$
}
```

El *script* de *restart-httpd* a ejecutar en el *localhost* podría ser el siguiente:

```
#!/bin/sh
#
# Script event handler para reiniciar el servidor web en la maquina local
#
# Nota: Este script solo reiniciará el servidor web si el servicio es
#       comprobado 3 veces (en un estado "SOFT") o si el servidor web
#       llega a un estado de error "HARD".
#
# ¿En qué estado se encuentra el servicio HTTP?
case "$1" in
OK)
# No hacer nada...
;;
WARNING)
#No se sabe la causa del estado WARNING del servicio, así que probablemente siga ejecutandose
    ...
;;
UNKNOWN)
# No se sabe la causa del estado UNKNOW del servicio, no hacer nada...
;;
CRITICAL)
# El servicio HTTP parece tener un problema, probablemente se pueda reiniciar el servicio...
        # El servicio ¿Está en un estado "SOFT" o "HARD"?
        case "$2" in
        SOFT)
# ¿En qué intento de comprobación está?  No se reiniciara en el primero ya que puede ser
# causa de alguna variación
                case "$3" in
                3)
# Hay un problema!. Intentar reiniciar el servicio.
                        echo -n "Restarting HTTP service (3rd soft critical state)..."
                        # Llamamos al script para reiniciar
                        /etc/rc.d/init.d/httpd restart
                        ;;
                        esac
                ;;
        HARD)
#Por algún motivo el servicio no se reinicio. Intentemos una vez más. Para este momento ya
#se ha enviado la notificación de caída del servicio.
                echo -n "Restarting HTTP service..."
                # Llamamos al script para reiniciar.
                /etc/rc.d/init.d/httpd restart
                ;;
        esac
        ;;
esac
exit 0
```

El *script* intentará reiniciar el servidor *web* en el equipo local en dos casos diferentes:

- Después de que el servicio HTTP se ha verificado por 3ª vez y está en un estado *CRITICAL SOFT*
- Después de que el servicio está en un estado *CRITICAL HARD*

El *script* debe reiniciar el servicio *web* y solucionar el problema antes de que el servicio pase a un estado de problema HARD, la primera vez se ejecuta en un estado de problema SOFT

3 y si no funciona se vuelve a ejecutar cuando el estado de problema pasa a HARD. Cabe señalar que el *event handler* sólo se ejecutará una vez cuando el servicio cambia a un estado de problema HARD. Esto evitará que Nagios ejecute continuamente el *script* para intentar reiniciar el servidor *web*.

## 5.2. Apache y PHP

Nagios muestra los estados de todos los servicios configurados a través de páginas *web*, esto implica que debe instalarse un servidor *web*.

Para una correcta visualización de la consola *web* de Nagios, se debe realizar una configuración personalizada dentro del servidor *web* Apache.

Se necesita crear un archivo de configuración (preferentemente con el nombre *nagios.conf*), y deberá estar localizado dentro del directorio de donde el servidor *web* Apache obtiene su configuración.

El archivo de configuración por defecto (*nagios.conf*) de Nagios es el siguiente:

```
Alias /nagios/cgi-bin "/usr/local/nagios/sbin"
<Directory "/usr/local/nagios/sbin">
# SSLRequireSSL
Options ExecCGI
AllowOverride None
Order allow,deny
Allow from all
# Order deny,allow
# Deny from all
# Allow from 127.0.0.1
AuthName "Nagios Access"
AuthType Basic
AuthUserFile /usr/local/nagios/etc/htpasswd.users
Require valid-user

</Directory>
Alias /nagios "/usr/local/nagios/share"
<Directory "/usr/local/nagios/share">
# SSLRequireSSL
Options None
AllowOverride None
Order allow,deny
Allow from all
# Order deny,allow
# Deny from all #
Allow from 127.0.0.1
AuthName "Nagios Access"
AuthType Basic
AuthUserFile /usr/local/nagios/etc/htpasswd.users
Require valid-user
</Directory>
```

## 5.3. PNP4Nagios

PNP4Nagios es un complemento para Nagios que genera las gráficas con los resultados de las comprobaciones realizadas por Nagios, todo esto con el objetivo de poder llevar un control más general de la monitorización de un determinado *host* o servicio.

### 5.3.1. Requerimientos del Sistema

PNP4Nagios requiere de forma obligatoria datos de rendimiento válidos de los *plugins* de Nagios.

Ejemplo del resultado arrojado al ejecutarse el *plugin check_ icmp* :

```
OK - 127.0.0.1: rta 2.687ms, lost 0% | rta=2.687ms;3000.000;5000.000;0; pl=0%;80;100;;
```

El texto a la izquierda del símbolo *pipe* ( | ) es más legible para una persona:

```
OK - 127.0.0.1: rta 2.687ms, lost 0%
```

Y los datos de rendimiento que procesa PNP estan a la derecha del símbolo *pipe* ( | ):

```
rta=2.687ms;3000.000;5000.000;0; pl=0%;80;100;;
```

Los datos de rendimiento se procesan automáticamente.

Descripción de los datos de rendimiento:

```
rta=2.687ms;3000.000;5000.000;0;
 |     |   |     |          |       | |
 |-----|---|-----|----------|-------|-|------ * Etiqueta
       |---|-----|----------|-------|-|------ * Valor Actual
           |-----|----------|-------|-|------ Unidad (UOM = UNIT of Measurement)
                 |----------|-------|-|------ Umbral de Advertencia
                            |-------|-|------ Umbral Crítico
                                    |-|------ Valor Mínimo
                                      |------ Valor Máximo
```

Los valores marcados con * son obligatorios. Todos los demás son opcionales.

Varias series de datos están separadas por espacios en blanco. Los datos no pueden contener espacios en blanco. Si la etiqueta contiene espacios en blanco, deben estar entre comillas simples.

### 5.3.2. *Software* Requerido

- Perl >= 5.x
- RRDtool >= 1.x
- PHP >= 5.1.6
- Nagios >= 2.x o 3.x

### 5.3.3. Almacenamiento

Los datos de rendimiento se almacenan en *Round Robin Databases* usando RRDtool. Esto es que después de algún tiempo, los datos que tienen más tiempo son descartados y reemplazados por los nuevos valores. El usar diferentes intervalos proporciona diferentes resoluciones. Cuando se usan los valores por defecto se permite almacenar datos con una resolución de un minuto para los dos últimos días, una resolución de cinco minutos para diez días, una resolución de 30 minutos durante 90 días y 6 horas de resolución durante cuatro años. Usando este formato de almacenamiento el tamaño de los archivos es el mismo todo el tiempo. Se necesitará aproximadamente 400 KB por fuente de datos.

### 5.3.4. Recolección de Datos

PNP4Nagios tiene varios modos para procesar los datos de rendimiento. Los modos son diferentes en cuestión de complejidad y rendimiento. Nagios llama un comando por cada equipo y cada servicio que generen datos de rendimiento para la creación de las gráficas. Dependiendo del modo elegido, los datos son pasados al archivo *process_perfdata.pl* o se almacenan en archivos temporales y posteriormente se procesan. *process_perfdata.pl* escribe los datos en archivos XML y los almacena en archivos RRD usando RRDtool.

### 5.3.5. Comparación de los diferentes modos

#### 5.3.5.1. Modo Sincrono

El "Modo Síncrono" es el más simple y sencillo de configurar. Nagios llama al *script* de Perl *process_perfdata.pl* para cada servicio y equipo, con el objetivo de procesar los datos. El modo síncrono trabaja bien hasta con 1.000 servicios con un intervalo de 5 minutos.

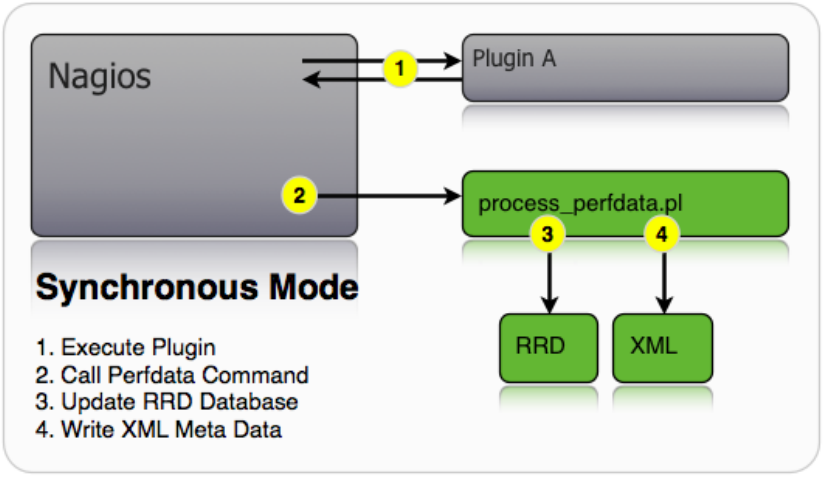


Figura 5.3: Funcionamiento del Modo Sincrono

#### 5.3.5.2. Modo Masivo

En el modo masivo Nagios, se almacenan los datos procesados en un archivo temporal. Después de un intervalo de tiempo definido el archivo es procesado y borrado. Como consecuencia el número de llamadas a *process_perfdata.pl* se reduce pero la ejecución de *process_perfdata.pl* dura más tiempo.

#### 5.3.5.3. Modo Masivo con NPCD

En este modo Nagios usa un archivo temporal para almacenar los datos y ejecuta un comando después de cierto tiempo predefinido. En lugar de procesar inmediatamente los datos usando *process_perfdata.pl*, el archivo es movido a un directorio de *spool*. Debido a que únicamente estamos moviendo un archivo del mismo sistema de archivos, esto no lleva prácticamente tiempo, por lo que Nagios es capaz de ejecutar su trabajo de forma inmediata. El *demonio* NPCD (*Nagios Performance C Daemon*) monitoriza el directorio en busca de nuevos archivos y le pasa los nombres a *process_perfdata.pl*. El procesado de los datos esta desacoplado de Nagios. NPCD es capaz de lanzar múltiples *hilos* de ejecución para el procesado de datos.

#### 5.3.5.4. Modo Masivo con npcdmod

Este escenario incluye *npcdmod.o*, un módulo NEB. Se reduce la configuración del "Modo Masivo con NPCD" a un par de líneas en el archivo *nagios.cfg*. Es similar al "Modo Masivo con NPCD" y tiene exactamente la misma funcionalidad y rendimiento.

### 5.3.6. Elección de Modo de Recolección de Datos

Debido a que en la presente tesis solo se va a monitorizar un *host* y dos servicios el modo que se usara para recolectar datos es el "Modo Síncrono".

### 5.3.7. Configuración del Modo Síncrono

El modo síncrono es la forma más fácil de integrar en Nagios, el recolector de datos *process_perfdata.pl*. Cada evento dispara la ejecución del comando *process-service-perfdata*. Inicialmente se debió habilitar el procesado de los datos de rendimiento en *nagios.cfg*. El valor por defecto es "0".

```
process_performance_data=1
```

El procesado de los datos debe ser deshabilitado en la definición de cada equipo o servicio para los que los datos de rendimiento NO deban ser procesados.

```
define service {
...
process_perf_data 0
...
}
```

La exportación de las variables de entorno debe estar habilitada para usar el modo síncrono. Se deben usar los valores por defecto (*export* está habilitado) o definir la variable en *nagios.cfg*.

```
enable_environment_macros=1
```

Además, los comandos para procesar los datos de rendimiento deben especificarse en *nagios.cfg*.

```
service_perfdata_command=process-service-perfdata
host_perfdata_command=process-host-perfdata
```

Se puede observar que la llamada a *process_perfdata.pl* no requiere de ningún argumento.

```
 define command {
 command_name     process-service-perfdata
command_line     /usr/bin/perl /usr/local/pnp4nagios/libexec/process_perfdata.pl
 }
 define command {
 command_name     process-host-perfdata
command_line     /usr/bin/perl /usr/local/pnp4nagios/libexec/process_perfdata.pl -d
    HOSTPERFDATA
 }
```

**Nota:** El *script* debe ser explícitamente llamado usando */usr/bin/perl* (o donde quiera que esté localizado el binario de Perl).

## 5.4. *Postfix*

Un servidor de correo es una aplicación que nos permite enviar mensajes (correos) de unos usuarios a otros, con independencia de la red que dichos usuarios estén utilizando.[13]

La presente tesis usa el protocolo SMTP.

SMTP, *Simple Mail Transfer Protocol* (Protocolo para la Transferencia Simple de Correo Electrónico), es un protocolo de la capa de aplicación. Protocolo de red basado en texto, utilizado para el intercambio de mensajes de correo electrónico entre computadoras y otros dispositivos. Está definido en el RFC 2821 y es un estándar oficial de Internet.[12]

Postfix es un servidor de correo electrónico de Wietse Venema que empezó como una alternativa en el área de Investigaciones de IBM.[14]

Postfix soporta los siguientes ambientes: AIX, BSD, HP-UX, IRIX, LINUX, MacOS X, Solaris, Tru64 UNIX, y otros sistemas UNIX. Estos requieren ANSI C, una libreria POSIX.1, y *sockets BSD*.[11]

## 5.5. NRPE

Un método alternativo para ejecutar *plugins* instalados en el equipo de destino por el NRPE *Nagios Remote Plugin Executor* (en español Ejecutor de Plugins Remoto Nagios).

El NRPE está instalado en el *host* de destino y a través del demonio de *xinet*, que debe estar configurado previamente. Si NRPE recibe una consulta desde el servidor de Nagios, se ejecutará la consulta correspondiente. El *plugin* o *script* que se va a ejecutar debe estar instalado en el *host* de destino.

## 5.6. Instalación del Servicio SSH

Para la instalación de *SSH* en el servidor Nagios se instalan los siguientes paquetes desde terminal. Se ejecuta como usuario *root* los siguientes comandos:

```
root@nagios:/home/nagios# aptitude install openssh-server
    root@nagios:/home/nagios# ps -ef | grep ssh
nagios    1882   1824   0 19:34 ?        00:00:00 /usr/bin/ssh-agent /usr/bin/dbus-launch --
    exit-with-session x-session-manager
root      3292      1   0 20:22 ?        00:00:00 /usr/sbin/sshd
root      3302   2183   0 20:22 pts/1    00:00:00 grep ssh
```

El primer comando instala el paquete *openssh-server* así como el paquete *openssh-client*, cuya configuración se realiza automáticamente, quedando activo el servicio SSH.

Para comprobar que se instaló correctamente se ejecuta el comando *ps* con los parametros *-ef* concatenando un filtro con el comando *grep* para ver solo las líneas que contengan la expresión *ssh*.

### 5.6.1. Configuración de Clave Pública y Clave Privada

El propósito de usar la autenticación mediante clave pública es que el usuario se pueda identificar automáticamente sin usar la autenticación clásica *usuario/contraseña.*

También se podría configurar SSH para que solo acepte la autenticación por clave pública mejorando la seguridad en el acceso remoto al servidor Nagios, permitiendo el acceso sólo a usuarios autorizados (que su clave pública se encuentre "registrada" en el servidor Nagios).

#### Configuración del servidor

La configuración del par de claves se realiza de la siguiente manera para *openssh.*

Como primer paso se debe tener previamente instalado *openssh-server*.

Se comprueba que el equipo donde está instalado SSH tiene activada la versión 2 del protocolo SSH y que esté habilitada la opción para utilizar claves RSA. Comprobando las correspondientes directivas en el archivo */etc/ssh/sshd_config*:

```
Protocol 2
RSAAuthentication yes
AuthorizedKeysFile %h/.ssh/authorized_keys
```

Para cada usuario que se necesite conectar debe existir el directorio */home/<usuario>/.ssh* y en el el archivo *authorized_keys*. Si no existen se deberán crear manualmente.

### Generación de los pares de claves RSA

SSH permite generar claves RSA o DSA, creándose un par de claves (una clave pública y una clave privada).

Las claves, pueden generarse en el servidor, en la computadora cliente, o en cualquier otra máquina.

Al final sólo la clave pública debe añadirse al archivo *authorized_keys* del servidor al que queremos conectarnos.

Para generar el par de claves se ejecutará el comando *ssh-keygen*. En el proceso, se le pedirá al usuario introducir un *passphrase*, que tiene la función de una contraseña.

```
$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/uli/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/uli/.ssh/id_rsa.
Your public key has been saved in /home/uli/.ssh/id_rsa.pub.
The key fingerprint is: 63:de:92:6d:00:fa:d9:17:55:45:ce:bc:25:42:8b:4c
uli@debh
The key's randomart image is:
+--[ RSA 1024]----+
|  E  . .oo|
|  o  o  o + |
|  .  o  + . =|
|  .  .  .  . .o|
|  .  S  .  .  |
|  .  =  *  .  |
|  o  =  =     |
|  +           |
|              |
+-----------------+
```

### Instalación de clave pública y protección de la clave privada

Para validar todo este proceso es necesario colocar la clave pública en el servidor donde se quiera autentificar.

En el servidor Nagios previamente se debe tener el archivo *id_rsa.pub* generado en el paso anterior y se ejecuta el siguiente comando:

```
$cat /home/nagios/id_rsa.pub >> /home/nagios/.ssh/authorized_keys
```

Realizado lo anterior solo faltaría reiniciar el servicio SSH y realizar una conexión al servidor Nagios para comprobar el funcionamiento.

```
$/etc/init.d/ssh restart
$ssh nagios@192.168.0.1
Using username "uli".
Authenticating with public key "rsa-key-20130511"
Passphrase for key "rsa-key-20130511":
Linux nagios 2.6.32-5-686 #1 SMP Mon Feb 25 01:04:36 UTC 2013 i686

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun May 12 21:19:44 2013 from 192.168.0.2
uli@nagios:~$
```

## 5.7. Instalación de Nagios y PNP4Nagios

A continuación se desarrollara el proceso de instalación en un equipo con distribución Debían 6.

**Paso 1: Instalación de Dependencias.**

Para instalar los paquetes en Debian, se debe ejecutar como usuario *root* los siguientes comandos:

Instalación automática del compilador *gcc*:

```
root@nagios:/home/nagios# apt-get install gcc
```

Instalación automática de los paquetes *build-essential*:

```
root@nagios:/home/nagios# apt-get install build-essential
```

Instalación automática del servidor web *apache2*:

```
root@nagios:/home/nagios# apt-get install apache2
```

Instalación automática de PHP 5 para desarrollador:

```
root@nagios:/home/nagios# apt-get install php5-dev
```

Instalación automática de PHP en modo *gd*:

```
root@nagios:/home/nagios# apt-get install php5-gd
```

Instalación automática de la herramienta RRD Tool para el complemento PNP4Nagios:

```
root@nagios:/home/nagios# apt-get install rrdtool
```

Instalación automática de la libreria *librrdtool-oo-perl* para la recopilación de los datos:

```
root@nagios:/home/nagios# apt-get install librrdtool-oo-perl
```

**Paso 2: Instalación Manual**

**Paso 2.1: Usuario *nagios* y configuración de usuario de *apache* para que pueda acceder a los archivos de conguración**

Se agrega al usuario *nagios* con el siguiente comando:

```
root@nagios:/home/nagios# useradd nagios
```

Se agrega al grupo *nagcmd* que sera usado por el servidor *web*:

```
root@nagios:/home/nagios# groupadd nagcmd
```

Se le asigna al usuario del servidor *web* y al grupo *nagcmd* acceso al archivo *nagios.cmd* para la ejecución de comandos de manera externa desde el servidor *web*:

```
root@nagios:/home/nagios# chown www-data:nagcmd /usr/local/nagios/var/rw/nagios.cmd
```

**Paso 2.2: Instalación de Nagios**

Para este paso se considera que previamente se ha realizado la descarga de los archivos *nagios-3.4.3.tar.gz*, *nagios-plugins-1.4.16.tar.gz*, *nagios-plugins-1.4.tar.gz* y *pnp4nagios-0.6.21.tar.gz*.

Se agrega al usuario *nagios* con el siguiente comando:

```
root@nagios:/home/nagios# cp nagios-3.4.3.tar.gz  nagios-plugins-1.4.16.tar.gz nagios-
    plugins-1.4.tar.gz pnp4nagios-0.6.21.tar.gz /tmp
```

Se posiciona en el directorio */tmp* y se desempaquetan los archivos:

```
root@nagios:/tmp# tar xvzf nagios-3.4.3.tar.gz
root@nagios:/tmp# tar xvzf nagios-plugins-1.4.16.tar.gz
root@nagios:/tmp# tar xvzf nagios-plugins-1.4.tar.gz
root@nagios:/tmp# tar xvzf pnp4nagios-0.6.21.tar.gz
```

Se ubica en el directorio */nagios* creado al desempaquetar el archivo *nagios-3.4.3.tar.gz* y se ejecuta el siguiente comando:

```
root@nagios:/tmp/nagios# ./configure --with-nagios-group=nagios --with-command-group=
    nagcmd
```

Parte de la salida de la ejecución de comando anterior es la siguiente:

```
...
*** Configuration summary for nagios 3.4.3 11-30-2012 ***:
  General Options: -------------------------
Nagios executable:  nagios
Nagios user/group:  nagios,nagios
Command user/group:  nagios,nagcmd
Embedded Perl:  no
Event Broker:  yes
Install ${prefix}:  /usr/local/nagios
Lock file:  ${prefix}/var/nagios.lock
Check result directory:  ${prefix}/var/spool/checkresults
Init directory:  /etc/init.d
Apache conf.d directory:  /etc/apache2/conf.d
Mail program:  /usr/bin/mail
Host OS:  linux-gnu
  Web Interface Options: ------------------------
HTML URL:  http://localhost/nagios/
CGI URL:  http://localhost/nagios/cgi-bin/
Traceroute (used by WAP):  /usr/sbin/traceroute

Review the options above for accuracy.  If they look okay,
type 'make all' to compile the main program and CGIs.
...
```

Se ejecutan los siguientes comandos para compilar:

```
root@nagios:/tmp/nagios# make all
root@nagios:/tmp/nagios# make install
root@nagios:/tmp/nagios# make install-init
root@nagios:/tmp/nagios# make install-commandmode
root@nagios:/tmp/nagios# make install-config
root@nagios:/tmp/nagios# make install-webconf
```

Se copia el contenido del directorio creado */contrib/eventhandlers/* al directorio */usr/local/nagios/libexec/*:

```
root@nagios:/tmp/nagios# cp -R contrib/eventhandlers/ /usr/local/nagios/libexec/
```

Se modifican los permisos al directorio */usr/local/nagios/libexec/eventhandlers* para que puedan ser manipulados por el usuario *nagios* y el grupo *nagios*:

```
root@nagios:/tmp/nagios# chown -R nagios:nagios /usr/local/nagios/libexec/eventhandlers
```

Se agrega el usuario *nagiosadmin* al archivo *htpasswd.users* y se le asigna una contraseña:

```
root@nagios:/tmp/nagios# htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
```

La salida de la ejecución del comando anterior es la siguiente:

```
New password:
Re-type new password:
Adding password for user nagiosadmin
```

Se agrega un *link simbólico* para que el servicio *nagios* inicie automáticamente junto con el sistema:

```
root@nagios:/tmp/nagios# ln -s /etc/init.d/nagios /etc/rcS.d/S99nagios
```

Para la instalación de los *plugins* se ejecuta el siguiente comando previamente se debe ubicar en el directorio *nagios-plugins-1.4.16* creado al desempaquetar el archivo *nagios-plugins-1.4.16.tar.gz*:

```
root@nagios:/tmp# cd nagios-plugins-1.4.16
root@nagios:/tmp/nagios-plugins-1.4.16# ./configure --with-nagios-user=nagios --with-
    nagios-group=nagios
```

Ejecutamos el siguiente comando para compilar los *plugins*:

```
root@nagios:/tmp/nagios-plugins-1.4.16# make
```

En el directorio */tmp/nagios-plugins-1.4* se busca el archivo *check_mysql.pl* que será el *plugin* a utilizar para la comprobación del servicio MySQL:

```
root@nagios:/tmp# find /tmp/nagios-plugins-1.4 | grep check_mysql
```

Salida de la ejecución del comando anterior:

```
/tmp/nagios-plugins-1.4/contrib/check_mysql.pl
```

Se copia el archvio *check_mysql.pl* al directorio */usr/local/nagios/libexec/* para que Nagios tenga acceso al *plugin*:

```
root@nagios:/tmp# cp /tmp/nagios-plugins-1.4/contrib/check_mysql.pl /usr/local/nagios/
    libexec
```

**Paso 2.3: Instalación de los Complementos**

Se ubica en el directorio *pnp4nagios-0.6.21* creado al desempaquetar el archivo *pnp4nagios-0.6.21.tar.gz*:

```
root@nagios:/tmp# cd pnp4nagios-0.6.21
```

Se ejecuta el archivo *configure*:

```
root@nagios:/tmp/pnp4nagios-0.6.21# ./configure
```

La salida del comando anterior es la siguiente:

```
...
*** Configuration summary for pnp4nagios-0.6.21 03-24-2013 ***
  General Options:
-------------------------  --------------------
Nagios user/group: nagios nagios
Install directory: /usr/local/pnp4nagios
HTML Dir: /usr/local/pnp4nagios/share
Config Dir: /usr/local/pnp4nagios/etc
Location of rrdtool binary: /usr/bin/rrdtool Version 1.4.3
RRDs Perl Modules: FOUND (Version 1.4003)
RRD Files stored in: /usr/local/pnp4nagios/var/perfdata process_perfdata.pl
Logfile: /usr/local/pnp4nagios/var/perfdata.log
Perfdata files (NPCD) stored in: /usr/local/pnp4nagios/var/spool

  Web Interface Options:
-------------------------  --------------------
HTML URL: http://localhost/pnp4nagios
Apache Config File: /etc/apache2/conf.d/pnp4nagios.conf

Review the options above for accuracy. If they look okay,
type 'make all' to compile.
...
```

Si se está de acuerdo con los parámetros mostrados se ejecuta el siguiente comando para la compilación:

```
root@nagios:/tmp/pnp4nagios-0.6.21# make all
```

El siguiente comando procederá a hacer la instalación de PNP4Nagios:

```
root@nagios:/tmp/pnp4nagios-0.6.21# make fullinstall
```

La salida de la ejecución del comando anterior es la siguiente:

```
...
*** Configuration summary for pnp4nagios-0.6.21 03-24-2013 ***
General Options:
-------------------------  --------------------
Nagios user/group: nagios nagios
Install directory: /usr/local/pnp4nagios
HTML Dir: /usr/local/pnp4nagios/share
Config Dir: /usr/local/pnp4nagios/etc
Location of rrdtool binary: /usr/bin/rrdtool Version 1.4.3
RRDs Perl Modules: FOUND (Version 1.4003)
RRD Files stored in: /usr/local/pnp4nagios/var/perfdata process_perfdata.pl
Logfile: /usr/local/pnp4nagios/var/perfdata.log
Perfdata files (NPCD) stored in: /usr/local/pnp4nagios/var/spool

Web Interface Options:
-------------------------  --------------------
HTML URL: http://localhost/pnp4nagios
Apache Config File: /etc/apache2/conf.d/pnp4nagios.conf


*** Main program, Scripts and HTML files installed ***
Enjoy.
...
```

**Paso 3: Iniciar el servicio Nagios**

Se ubica en el directorio */home/nagios*:

```
root@nagios:/tmp/pnp4nagios-0.6.21# cd /home/nagios/
```

Se descarga el archivo *verify_pnp_config* para comprobar que la configuración de PNP4Nagios sea la adecuada:

```
root@nagios:/home/nagios# wget http://verify.pnp4nagios.org/verify_pnp_config
```

Se configura el servidor *web* en modo de reescritura para que pueda procesar los datos de rendimiento:

```
root@nagios:/home/nagios# a2enmod rewrite
```

Salida de la ejecución del comando anterior:

```
Enabling module rewrite. Run '/etc/init.d/apache2 restart' to activate new configuration!
```

Se reinicia apache para que se tome la nueva configuración:

```
root@nagios:/home/nagios# /etc/init.d/apache2 restart
```

Se realizan las configuraciones necesarias en *nagios.cfg* para que PNP4Nagios en Modo Síncrono funcione adecuadamente:

```
root@nagios:/home/nagios# nano /usr/local/nagios/etc/nagios.cfg
```

Se comprueba que la configuración de Nagios sea correcta:

```
root@nagios:/tmp/nagios# /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
```

La salida del comando anterior es la siguiente:

```
Nagios Core 3.4.3
Copyright (c) 2009-2011 Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 11-30-2012
License: GPL


Website: http://www.nagios.org
Reading configuration data...
   Read main config file okay...
Processing object config file '/usr/local/nagios/etc/objects/commands.cfg'...
Processing object config file '/usr/local/nagios/etc/objects/contacts.cfg'...
Processing object config file '/usr/local/nagios/etc/objects/timeperiods.cfg'...
```

```
Processing object config file '/usr/local/nagios/etc/objects/templates.cfg'...
Processing object config file '/usr/local/nagios/etc/objects/localhost.cfg'...
   Read object config files okay...


Running pre-flight check on configuration data...

Checking services...
        Checked 8 services.
Checking hosts...
        Checked 1 hosts.
Checking host groups...
        Checked 1 host groups.
Checking service groups...
        Checked 0 service groups.
Checking contacts...
        Checked 1 contacts.
Checking contact groups...
        Checked 1 contact groups.
Checking service escalations...
        Checked 0 service escalations.
Checking service dependencies...
        Checked 0 service dependencies.
Checking host escalations...
        Checked 0 host escalations.
Checking host dependencies...
        Checked 0 host dependencies.
Checking commands...
        Checked 24 commands.
Checking time periods...
        Checked 5 time periods.
Checking for circular paths between hosts...
Checking for circular host and service dependencies...
Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors:   0

Things look okay - No serious problems were detected during the pre-flight check
```

Se inicia el servicio:

```
root@nagios:/tmp/nagios# /etc/init.d/nagios start
Starting nagios:Sin directorio, entrando con HOME=/
done.
```

Se comprueba que la configuración de PNP4Nagios sea correcta:

```
root@nagios:/home/nagios# perl verify_pnp_config --mode sync --config=/usr/local/nagios/
    etc/nagios.cfg --pnpcfg=/usr/local/pnp4nagios/etc
```

La salida del comando anterior es la siguiente:

```
[INFO]  ========== Starting Environment Checks ==========
[INFO]  My version is: verify_pnp_config-0.6.21-R.40
[INFO]  Start Options: verify_pnp_config --mode sync --config=/usr/local/nagios/etc/
    nagios.cfg --pnpcfg=/usr/local/pnp4nagios/etc
[INFO]  Reading /usr/local/nagios/etc/nagios.cfg
[OK  ]  Running product is 'nagios'
[OK  ]  object_cache_file is defined
[OK  ]  object_cache_file=/usr/local/nagios/var/objects.cache
[INFO]  Reading /usr/local/nagios/var/objects.cache
[OK  ]  resource_file is defined
[OK  ]  resource_file=/usr/local/nagios/etc/resource.cfg
[INFO]  Reading /usr/local/nagios/etc/resource.cfg
[INFO]  Reading /usr/local/pnp4nagios/etc/process_perfdata.cfg
[INFO]  Reading /usr/local/pnp4nagios/etc/pnp4nagios_release
[OK  ]  Found PNP4Nagios version "0.6.21"
[OK  ]  Effective User is 'nagios'
[OK  ]  User nagios exists with ID '1001'
[OK  ]  Effective group is 'nagios'
[OK  ]  Group nagios exists with ID '1001'
[INFO]  ========== Checking Sync Mode Config ==========
[OK  ]  process_performance_data is 1 compared with '/1/'
[OK  ]  enable_environment_macros is 1 compared with '/1/'
```

```
[OK  ]  service_perfdata_command is defined
[OK  ]  service_perfdata_command=process-service-perfdata
[OK  ]  host_perfdata_command is defined
[OK  ]  host_perfdata_command=process-host-perfdata
[INFO]  Nagios config looks good so far
[INFO]  ========== Checking config values ==========
service_perfdata_command at verify_pnp_config line 462.
[OK  ]  Command process-service-perfdata is defined
[OK  ]  '/usr/bin/perl /usr/local/pnp4nagios/libexec/process_perfdata.pl'
[OK  ]  Command looks good
[OK  ]  Script /usr/local/pnp4nagios/libexec/process_perfdata.pl is executable
host_perfdata_command at verify_pnp_config line 462.
[OK  ]  Command process-host-perfdata is defined
[OK  ]  '/usr/bin/perl /usr/local/pnp4nagios/libexec/process_perfdata.pl -d HOSTPERFDATA'
[OK  ]  Command looks good
[OK  ]  Script /usr/local/pnp4nagios/libexec/process_perfdata.pl is executable
[INFO]  ========== Starting global checks ==========
[OK  ]  status_file is defined
[OK  ]  status_file=/usr/local/nagios/var/status.dat
[INFO]  host_query =
[INFO]  service_query =
[INFO]  Reading /usr/local/nagios/var/status.dat
[INFO]  ==== Starting rrdtool checks ====
[OK  ]  RRDTOOL is defined [OK  ]  RRDTOOL=/usr/bin/rrdtool
[OK  ]  /usr/bin/rrdtool is executable
[OK  ]  RRDtool 1.4.3  Copyright 1997-2009 by Tobias Oetiker <tobi@oetiker.ch>
[OK  ]  USE_RRDs is defined
[OK  ]  USE_RRDs=1
[OK  ]  Perl RRDs modules are loadable
[INFO]  ==== Starting directory checks ====
[OK  ]  RRDPATH is defined
[OK  ]  RRDPATH=/usr/local/pnp4nagios/var/perfdata
[OK  ]  Perfdata directory '/usr/local/pnp4nagios/var/perfdata' exists
[OK  ]  All hosts/services are providing performance data
[OK  ]  'process_perf_data 1' is set for 15 of your hosts/services
[INFO]  ==== System sizing ====
[OK  ]  15 hosts/service objects defined
[INFO]  ==== Check statistics ====
[OK  ]  Warning: 0, Critical: 0
[OK  ]  Checks finished...
```

Con la ejecución de los comandos anteriores y la configuración correcta de los diferentes archivos, queda funcionando de manera adecuada Nagios Core 3.4.3 y PNP4Nagios. Lo que quiere decir que podemos monitorizar los servicios de MySQL, HTTP y hacer PING con el servidor, además de obtener valores para las respectivas gráficas de los servicios HTTP y PING.

### Paso 4: Configuración la Interfaz *Web*

La configuración en el archvio *nagios.conf* deberá quedar como sigue:

```
root@nagios:/home/nagios# cat /etc/apache2/conf.d/nagios.conf
# SAMPLE CONFIG SNIPPETS FOR APACHE WEB SERVER
# Last Modified: 11-26-2005
#
# This file contains examples of entries that need
# to be incorporated into your Apache web server
# configuration file. Customize the paths, etc. as
# needed to fit your system.

ScriptAlias /nagios/cgi-bin "/usr/local/nagios/sbin"


<Directory "/usr/local/nagios/sbin">
# SSLRequireSSL
Options ExecCGI
AllowOverride None
Order allow,deny
Allow from all
# Order deny,allow
# Deny from all
# Allow from 127.0.0.1
AuthName "Nagios Access"
AuthType Basic
AuthUserFile /usr/local/nagios/etc/htpasswd.users
Require valid-user
```

```
</Directory>

Alias /nagios "/usr/local/nagios/share"


<Directory "/usr/local/nagios/share">
# SSLRequireSSL
Options None
AllowOverride None
Order allow,deny
Allow from all
# Order deny,allow
# Deny from all
# Allow from 127.0.0.1
AuthName "Nagios Access"
AuthType Basic
AuthUserFile /usr/local/nagios/etc/htpasswd.users
Require valid-user
</Directory>
```

Con la configuración anterior ya podemos accesar a la interfaz web de Nagios Core 3.4.3 en la dirección *http://localhost/nagios*.

## 5.8. Instalación *Postfix*

Para la instalación de Postfix todos los comandos se ejecutan como usuario *root*.
Se ejecuta el siguiente comando para la instalación automática de Postfix:

```
root@nagios:~# aptitude install postfix
```

Se realizan las configuraciones necesarias en el archivo *main.cf* [6]:

```
root@nagios:~# nano /etc/postfix/main.cf
```

Se cambian los permisos al arhcivo *passwd* para brindar mayor seguridad al archivo:

```
root@nagios:~# chmod 600 /etc/postfix/sasl/passwd
```

Se asigna el archivo *passwd* a:

```
root@nagios:~# postmap /etc/postfix/sasl/passwd
```

Se instala el paquete *ca-certificates*:

```
root@nagios:~# aptitude install ca-certificates
```

El contenido del archivo *Equifax_Secure_CA.pem* en el directorio */etc/ssl/certs/* se pasa al archivo *cacert.pem* en el directorio */etc/postfix/*:

```
root@nagios:~# cat /etc/ssl/certs/Equifax_Secure_CA.pem > /etc/postfix/cacert.pem
```

Se reinicia el servicio de Postfix para guardar los cambios:

```
root@nagios:~# /etc/init.d/postfix restart
Stopping Postfix Mail Transport Agent: postfix.
Starting Postfix Mail Transport Agent: postfix.
```

Se comprueba que el servicio funciona:

```
root@nagios:~# echo "Éste es un correo de prueba" | mail email.prueba@gmail.com
```

Se comprueba en el archivo *mail.log* en el directorio */var/log/* los *logs* de la ejecución del comando anterior:

```
root@nagios:~# tail -f /var/log/mail.log
May 2 21:25:01 nagios postfix/master[2702]: daemon started -- version 2.7.1,
    configuration /etc/postfix
May 2 21:34:40 nagios postfix/master[2702]: terminating on signal 15
May 2 21:34:40 nagios postfix/master[3314]: daemon started -- version 2.7.1,
    configuration /etc/postfix
May 2 21:35:58 nagios postfix/pickup[3320]: 4F654E0062: uid=0 from=<root>
May 2 21:35:58 nagios postfix/cleanup[3379]: 4F654E0062: message-id=<20130503023558.4
    F654E0062@nagios>
```

```
May 2 21:35:58 nagios postfix/qmgr[3321]: 4F654E0062: from=<root@nagios>, size=280, nrcpt
    =1 (queue active)
May 2 21:36:00 nagios postfix/smtp[3381]: 4F654E0062: to=<email.prueba@gmail.com>, relay=
    smtp.gmail.com[173.194.64.109]:587, delay=2.1, delays=0.03/0.11/0.85/1.1, dsn=2.0.0,
    status=sent (250 2.0.0 OK 1367548547 jv10sm2446515oeb.3 - gsmtp)
May 2 21:36:00 nagios postfix/qmgr[3321]: 4F654E0062: removed
```

## 5.9. Instalación NRPE

### En el servidor Nagios

Los comandos se ejecutan como usuario *root*.
Se necesita instalar el paquete *libssl-dev*:

```
root@vignag:/home/player1# apt-get install libssl-dev
```

Previamente se descarga NRPE en el caso de la presente tesis fué el archivo *nrpe-2.14.tar.gz* y se copia en el directorio */tmp*.

```
root@vignag:/home/player1# cp ./nrpe-2.14.tar.gz /tmp/
```

En el directorio */tmp* se desempaqueta el archivo.

```
root@vignag:/tmp# tar xvzf nrpe-2.14.tar.gz
```

En el directorio creado por la ejecución del comando anterior se ejecuta el siguiente comando y si se esta de acuerdo con la configuración se compila.

```
root@vignag:/tmp/nrpe-2.14# ./configure --enable-command-args
...
*** Configuration summary for nrpe 2.14 12-21-2012 ***:
General Options: -------------------------
NRPE port: 5666
NRPE user: nagios
NRPE group: nagios
Nagios user: nagios
Nagios group: nagios
Review the options above for accuracy. If they look okay,
type 'make all' to compile the NRPE daemon and client.
...
root@vignag:/tmp/nrpe-2.14# make all
```

Con esto queda instalado y configurado NRPE en el servidor Nagios.

### Instalación y Configuración en el Cliente.

Los comandos se ejecutan como usuario *root*.
Se necesita instalar el paquete *libssl-dev*, compilador *gcc*, paquete *build-essential* y *xinetd*:

```
root@vignag:/home/player1# apt-get install libssl-dev
root@alice:/home/player1# apt-get install gcc
root@alice:/home/player1# aptitude install xinetd
root@alice:/home/player1# aptitude install build-essential
```

En este punto se realizan los mismo pasos que en la instalación del servidor a partir de la descarga de PNP4Nagios.

Se configura el puerto en el archivo *nrpe* del directorio */etc/xinetd.d/* y se comprueba que NRPE este funcionando en el puerto adecuado.

```
root@dicygserver:/tmp/nrpe-2.14# nano /etc/xinetd.d/nrpe
root@dicygserver:/tmp/nrpe-2.14#  netstat -at | grep
nrpe tcp        0      0 *:nrpe                  *:*                     LISTEN
root@dicygserver:/tmp/nrpe-2.14# /usr/local/nagios/libexec/check_nrpe -H localhost
NRPE v2.14
```

Con esto queda instalado y configurado el Agente NRPE en el cliente.

## 5.10. Plan de Contingencia

### 5.10.1. Diagnóstico

Para el diseño de una propuesta para la solución de un problema es necesaria una revisión de cada uno de los componentes que conforman el sistema, de esta manera se asegura que las acciones de solución que se propongan tendrán un fundamento realista.

#### Servicios Producidos

Algunos de los servicios que proporciona la DICYG son:

- Servicio de HTTP (páginas *web* de la división)
- Servicio de MySQL (Bases de Datos asociadas a las páginas *web*)

Para la DICYG los servicios antes mencionados son de vital importancia ya que los usan personal académico y alumnos para realizar sus actividades diarias o periódicas pero de una prioridad alta y que en muchas ocasiones tienen que ser realizadas en un límite de tiempo.

Debido a la importancia de los servicios antes mencionados que proporciona la DICYG a su personal académico y alumnos, la presente tesis plantea un plan de contingencia a seguir en caso de que alguno de estos o ambos servicios fallen.

### 5.10.2. Definición

Un plan de contingencia es un programa alternativo para que una organización pueda recuperarse de un desastre informático y restablecer sus operaciones con rapidez.

Un Plan de Contingencia de Seguridad Informática consiste en los pasos que se deben seguir, cuando surge alguna incidencia, con el objetivo de recuperar, aunque sea en parte, el funcionamiento de la organización.

Se entiende por recuperación, "*...tanto la capacidad de seguir trabajando en un plazo mínimo después de que se haya producido el problema, como la posibilidad de volver a la situación anterior al mismo, habiendo reemplazado o recuperado el máximo posible de los recursos e información...*".[28]

La recuperación de la información se basa en el uso de una política de copias de seguridad (*backup's*) adecuada.

Para la presente tesis el plan de contingencia incluye:

- Un plan de respaldo: Que son las actividades a realizar antes de se presente un incidente.
- Un plan de recuperación: Que son las actividades a realizar durante un incidente.

Un plan de contingencia propone una serie de procedimientos alternativos al funcionamiento normal de una organización cuando alguna de sus funciones importantes se ve perjudicada. Es necesario que el plan de contingencia incluya un plan de recuperación, el cual tendrá como objetivo, restaurar el servicio de forma rápida, eficiente y con el menor costo y pérdidas posibles.

### 5.10.3. Planificación

La fase de planificación es la etapa donde se definen las actividades a realizar para intentar conservar la continuidad en el funcionamiento de la organización durante una emergencia.

Puntos a considerar:

- Identificar los servicios que son importantes para la DICYG.
- Evaluar la situación actual.
- Identificación y asignación de los grupos de trabajo con su respectiva definición de los roles y responsabilidades.
- Identificación de los riesgos que puede producir alguna interrupción en los servicios.
- Obtención de la aprobación y respaldo de la administradora de la red y del Jefe de la División.
- Si ocurre una interrupción en los servicios HTTP y/o MySQL que proporciona la DICYG durante un período de funcionamiento normal, se pondrá en práctica el plan de contingencia.

### 5.10.4. Escenarios Posibles

El alcance en la definición de los escenarios se basa unicamente en las interrupciones que afecten a los servicios definidos en la parte de Servicios Producidos.

Cabe mencionar que la implementación de un plan de contingencia "general" para una organización y/o para la misma DICYG contemplaría varios aspectos que no están definidos en este proyecto, ya que esta tesis se limita a la implementación del Sistema de Monitorización. Por lo tanto el "plan de contingencia" se enfoca a las interrupciones que se puedan presentar en los servicios HTTP, MySQL y/o en la disponibilidad del servidor *dicyg* , es decir, no es un plan de contingencia generalizado para la DICYG.

Se recomienda que si el lector busca desarrollar un plan de contingencia "general" consulte la bibliografía proporcionada que hace referencia al tema, para profundizar sobre el mismo.

Los escenarios posibles son los siguientes:

- Interrupción del servicio HTTP
- Interrupción del servicio MySQL
- Interupción del los servicios HTTP y MySQL al mismo tiempo
- Interrupción en la disponibilidad del servidor *dicyg*

### 5.10.5. Planteamiento

Para llegar a este punto, en el caso de fallo en algún servicio o en ambos, se considera que Nagios funciona correctamente además de que Nagios ya intento restablecer dichos servicios de manera automática con *event handlers* y también el administrador ha sido notificado de que hay problema con alguno o con ambos servicios.

**Escenario de interrupción del servicio MySQL**

PASOS A SEGUIR:

- El administrador debe acceder al servidor *dicyg* y ejecutar el siguiente comando:

```
ps -ef | grep mysql
```

Con el comando anterior observaremos todos los procesos relacionados con MySQL.

- Se ejecuta el siguiente comando para parar los procesos referentes a MySQL:

```
kill -9 PID_mysql
```

El comando se ejecutará para cada uno de los procesos.

- Para volver a iniciar el servicio de MySQL se ejecuta el siguiente comando:

```
/etc/init.d/mysql start
```

- Y se comprueba que haya arrancado con el siguiente comando:

```
/etc/init.d/mysql status
```

- Si no llega a levantar el servicio, revisar los *logs* de error de MySQL, el problema ya es más específico. Además de que se puede evaluar el reinicio del servidor *dicyg*.
- Si se han revisado los *logs* y se ha reiniciando el servidor *dicyg*, entonces el problema puede deberse a el rendimiento del servidor *dicyg*.

**Escenario de interrupción del servicio HTTP**

PASOS A SEGUIR:
El administrador debe acceder al servidor *dicyg* y ejecutar el siguiente comando:

```
ps -ef | grep apache2
```

Con el comando anterior observaremos todos los procesos relacionados con HTTP.

- Se ejecuta el siguiente comando para parar los procesos referentes a HTTP:

```
kill -9 PID_apache2
```

El comando se ejecutará para cada uno de los procesos listados.

- Para volver a iniciar el servicio de HTTP se ejecuta el siguiente comando:

```
/etc/init.d/apache2 start
```

- Y se comprueba que haya arrancado con el siguiente comando:

```
/etc/init.d/apache2 status
```

- Si no llega a levantar el servicio, revisar los *logs* de error de HTTP, el problema ya es más específico. Además de que se puede evaluar el reinicio del servidor *dicyg*.
- Si se han revisado los *logs* y se ha reiniciando el servidor *dicyg*, entonces el problema puede deberse a el rendimiento del servidor *dicyg*.

### Escenario de interrupción del los servicios HTTP y MySQL al mismo tiempo

Para los escenarios de falla en ambos servicios (MySQL y HTTP) al mismo tiempo, realizar el restablecimiento para cada uno de los servicios.

### Interrupción en la disponibilidad del servidor *dicyg*

Para llegar a este punto, en el caso de fallo en la disponibilidad en el servidor dicyg, el administrador ha sido notificado sobre el problema.

Algunas posibles fallas de la interrupción en la disponibilidad son:

- Que el servidor *dicyg* este apagado por algún fallo en la energía eléctrica
- Que el cable de red se haya dañado o desconectado.
- Que el DHCP haya sufrido alguna configuración y la IP haya cambiado (Nagios no podría detectar ese cambio).
- Que el mismo servidor *dicyg* haya sufrido un cambio en la configuración de su interfaz de red

PASOS A SEGUIR:

- Comprobar el correcto funcionamiento de las características físicas.
- Comprobar si se ha realizado algun cambio en configuraciones de IP's o interfaces.
- Comprobar la disponibilidad haciendo ping al servidor dicyg con el siguiente comando:

```
ping 192.168.1.1 -c 5
```

- De observarse resultados no esperados, entonces volver a revisar configuraciones y características físicas hasta tener los resultados esperados.

# Resultados

*El mundo exige resultados.*

*No le cuentes a otros tus dolores del parto.*

*Muéstrales al niño.*

Indira Gandhi (1917-1984) Estadista y política hindú.

# Capítulo 6

# Análisis de Gráficas

## 6.1. Gráficas de disponibilidad del servidor *dicyg* (PING)

Las gráficas que genera PNP4Nagios para el servicio definido PING son dos:

- *Round Trip Times*: Que hace referencia al tiempo que tarda un paquete (enviado desde el servidor Nagios) en regresar al origen de envío y que pase por el servidor *dicyg*.
- *Packets Lost*: Muestra el valor en porcentaje de los paquetes perdidos durante el envío de 5 peticiones *ping* al servidor *dicyg*.

La siguiente gráfica es un ejemplo de la monitorización del servicio PING del servidor *dicyg* en un intervalo de tiempo de 4 horas:

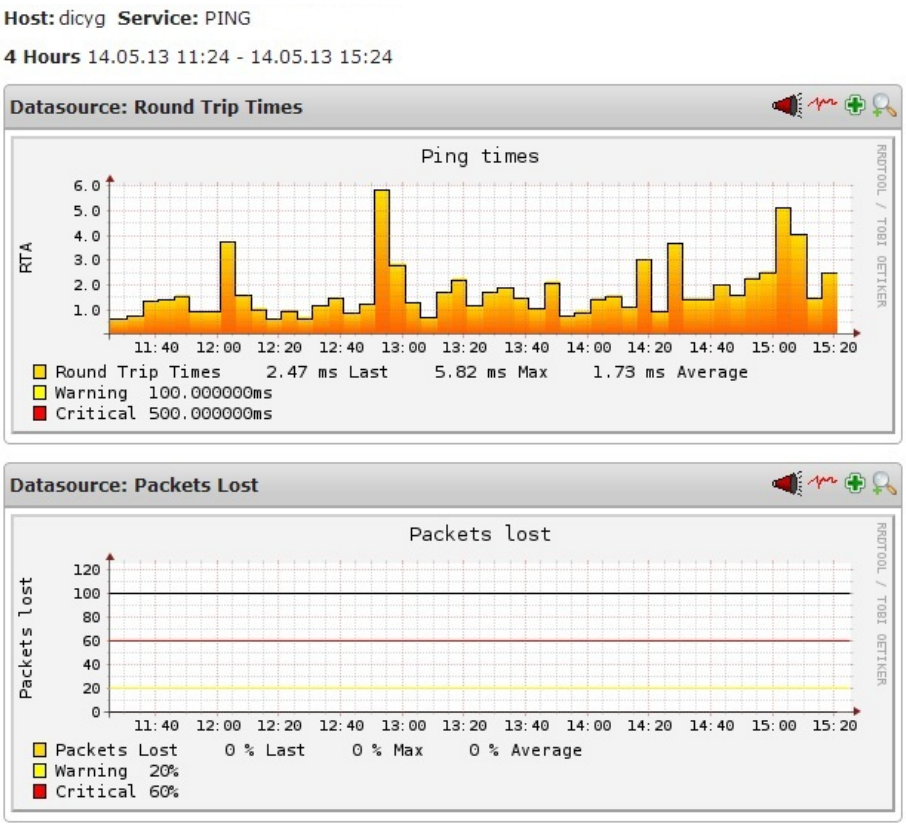


Figura 6.1: Ejemplo de gráfica del servicio PING en el servidor *dicyg*: últimas 4 horas

La siguiente gráfica es un ejemplo de la monitorización del servicio PING del servidor *dicyg* en un intervalo de tiempo de 25 horas:

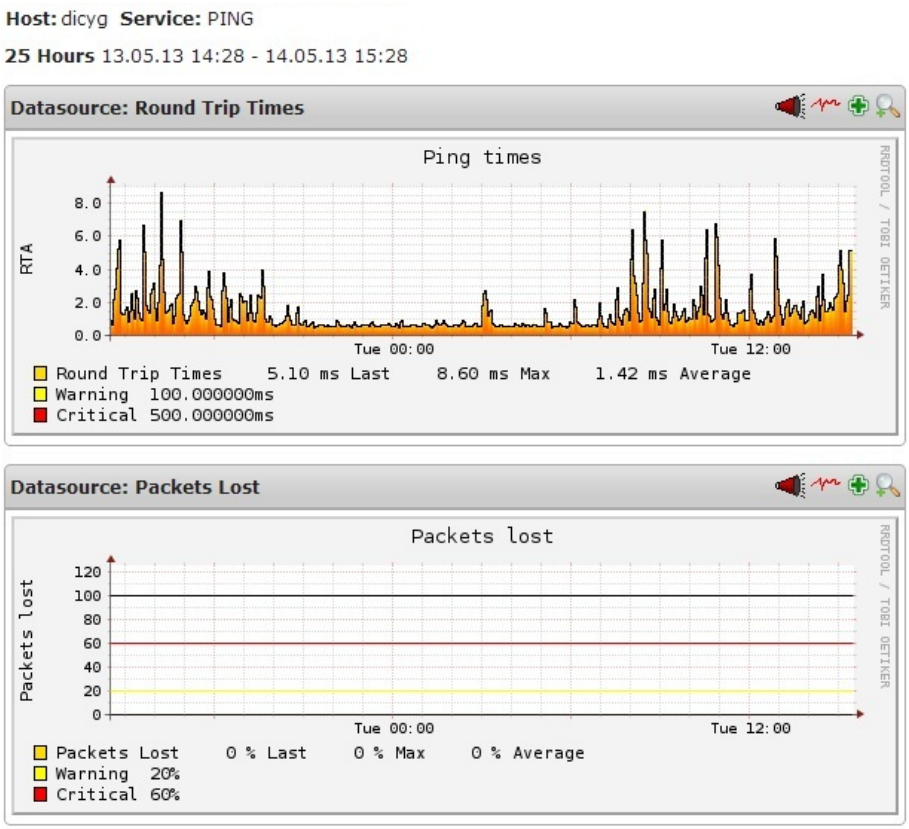


Figura 6.2: Ejemplo de gráfica del servicio PING en el servidor *dicyg*: últimas 25 horas

La siguiente gráfica es un ejemplo de la monitorización del servicio PING del servidor *dicyg* en un intervalo de tiempo de una semana:

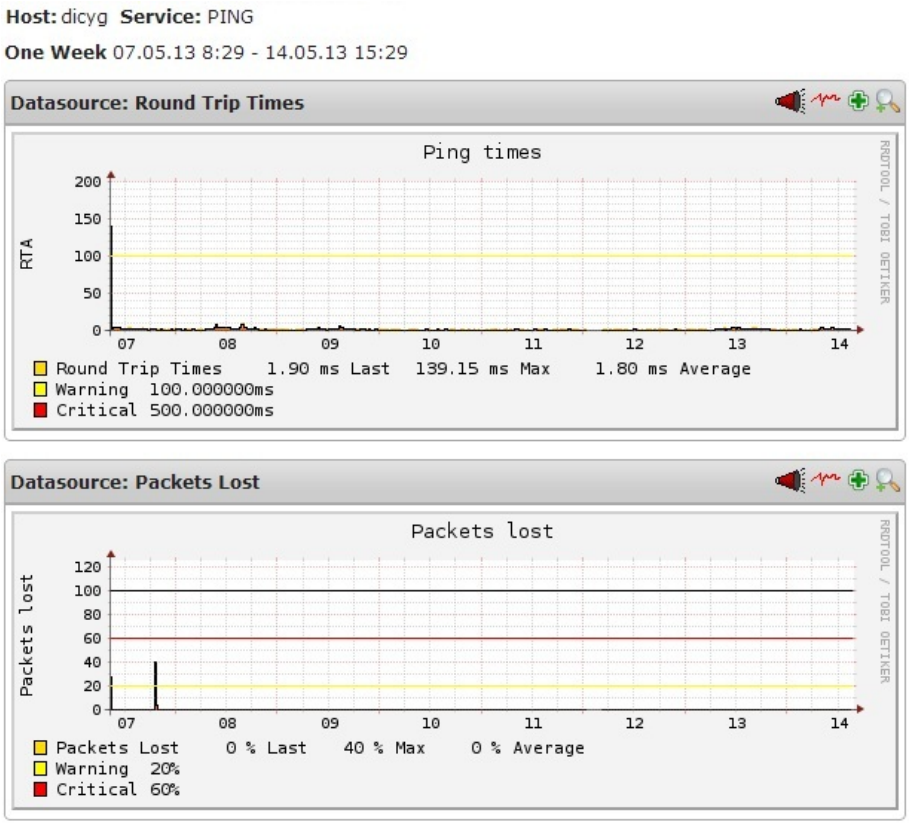


Figura 6.3: Ejemplo de gráfica del servicio PING en el servidor *dicyg*: última semana

La siguiente gráfica es un ejemplo de la monitorización del servicio PING del servidor *dicyg* en un intervalo de tiempo de un mes:

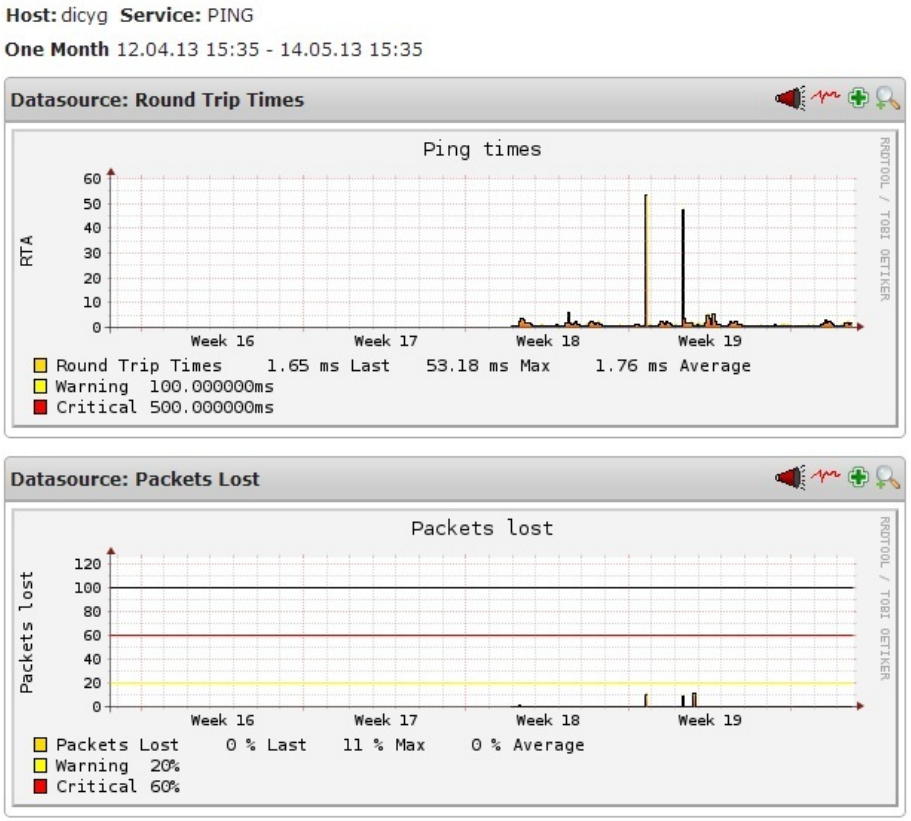


Figura 6.4: Ejemplo de gráfica del servicio PING en el servidor *dicyg*: último mes

La siguiente gráfica es un ejemplo de la monitorización del servicio PING del servidor *dicyg* en un intervalo de tiempo de un año:[1]

[1]La gráfica muestra un rango de tiempo de un año, pero los datos obtenidos que se muestran en la gráfica son de un par de semanas entre abril y mayo.

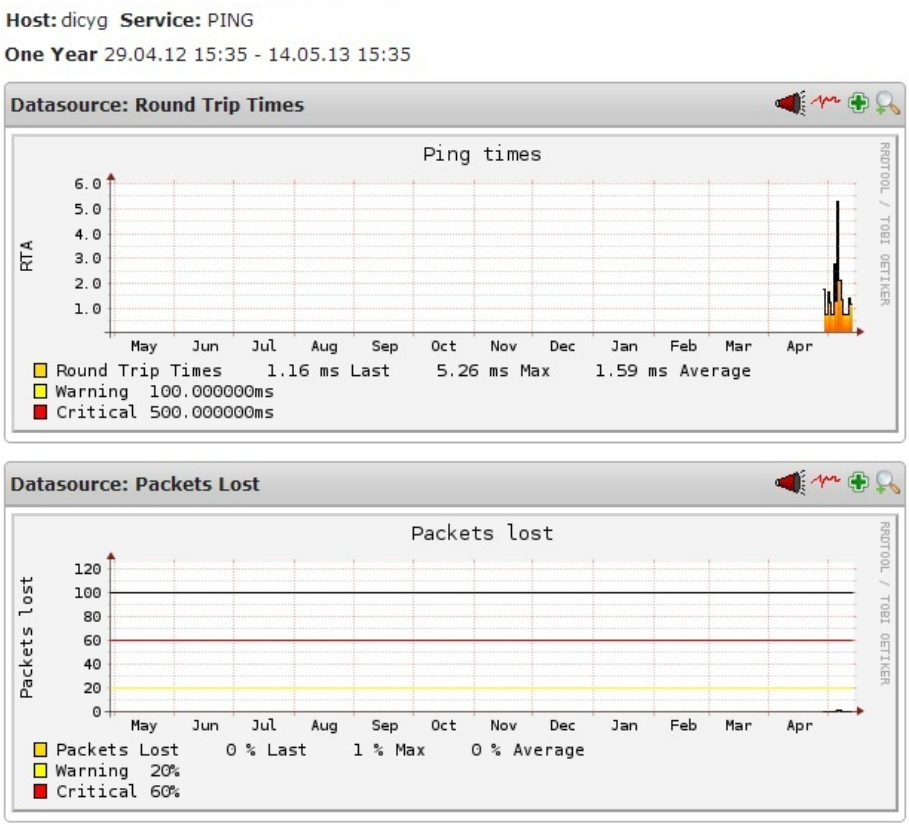


Figura 6.5: Ejemplo de gráfica del servicio PING en el servidor *dicyg*: último año

En la gráfica del servicio PING *Round Trip Times* se puede observar que hay picos, lo que significa que el servidor *dicyg* ha tardado tiempo en responder a la petición de *ping*. En los picos de la gráfica *Round Trip Times* es recomendable observar la gráfica de *Packet Lost* que informa si en ese tiempo hubo pérdida de paquetes, lo que significaría que en ese momento se pudo haber perdido información en la comunicación con el servidor *dicyg*.

## 6.2. Gráficas del Servicio HTTP

Las gráficas de monitorización del servicio HTTP muestran el tiempo que se tarda el servidor *dicyg* en responder el procesado de un archivo de 339Bytes.

La siguiente gráfica es un ejemplo de la monitorización del servicio HTTP del servidor *dicyg* en un intervalo de tiempo de 4 horas:

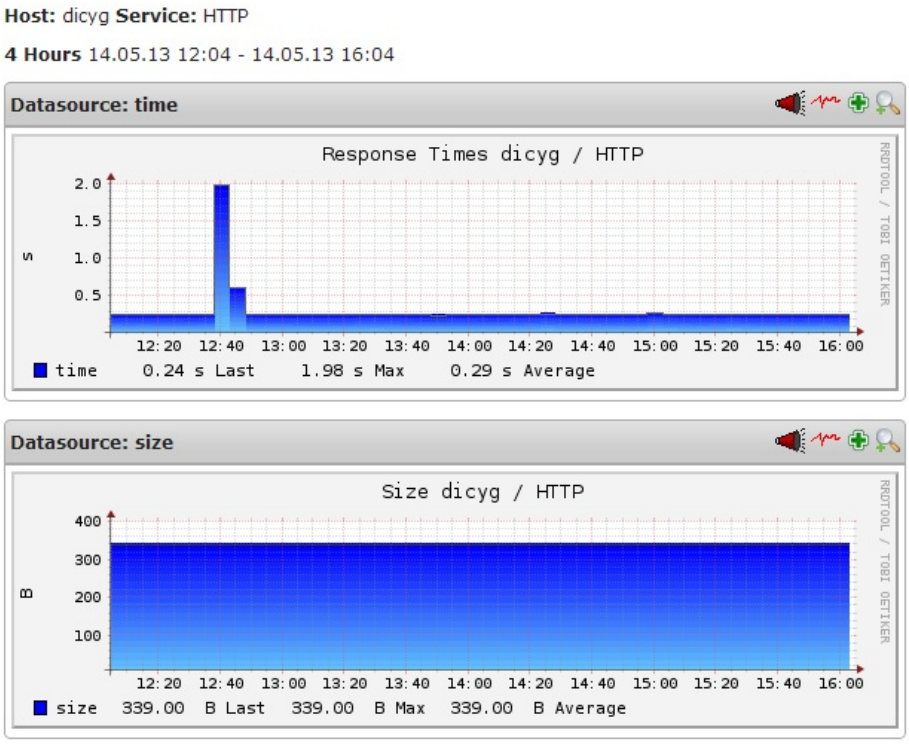


Figura 6.6: Ejemplo de gráfica del servicio HTTP en el servidor *dicyg*: últimas 4 horas

La siguiente gráfica es un ejemplo de la monitorización del servicio HTTP del servidor *dicyg* en un intervalo de tiempo de 25 horas:

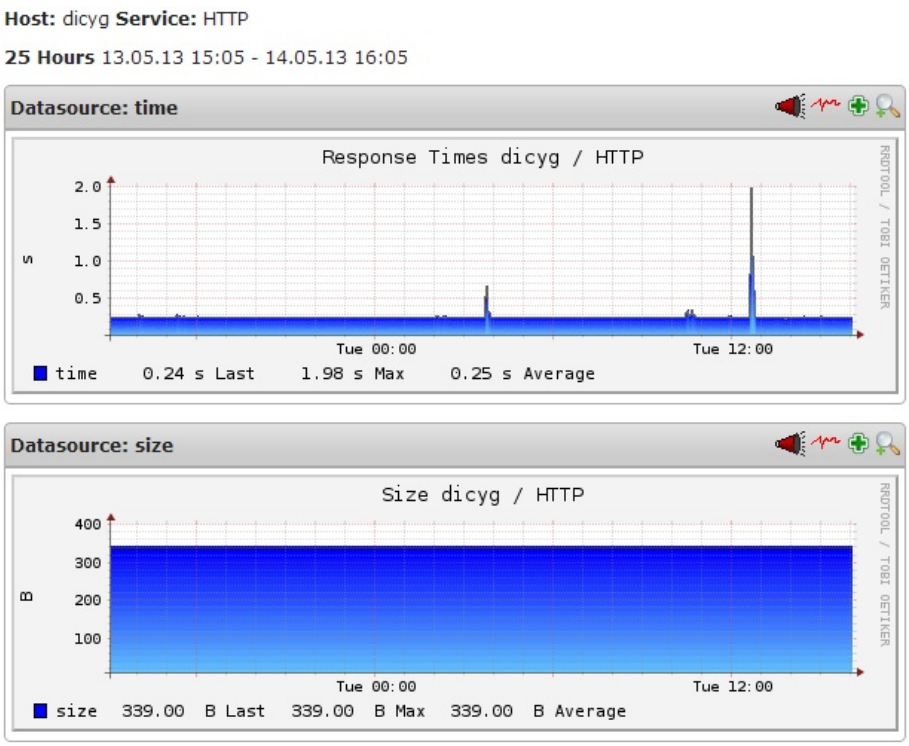


Figura 6.7: Ejemplo de gráfica del servicio HTTP en el servidor *dicyg*: últimas 25 horas

La siguiente gráfica es un ejemplo de la monitorización del servicio HTTP del servidor *dicyg* en un intervalo de tiempo de una semana:

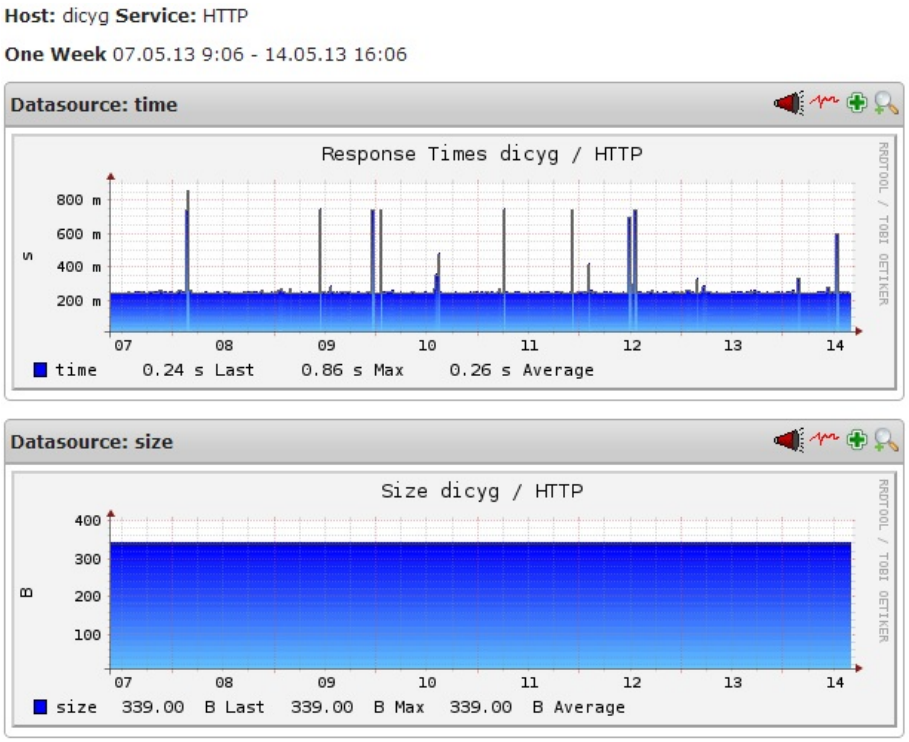


Figura 6.8: Ejemplo de gráfica del servicio HTTP en el servidor *dicyg*: última semana

La siguiente gráfica es un ejemplo de la monitorización del servicio HTTP del servidor *dicyg* en un intervalo de tiempo de un mes:

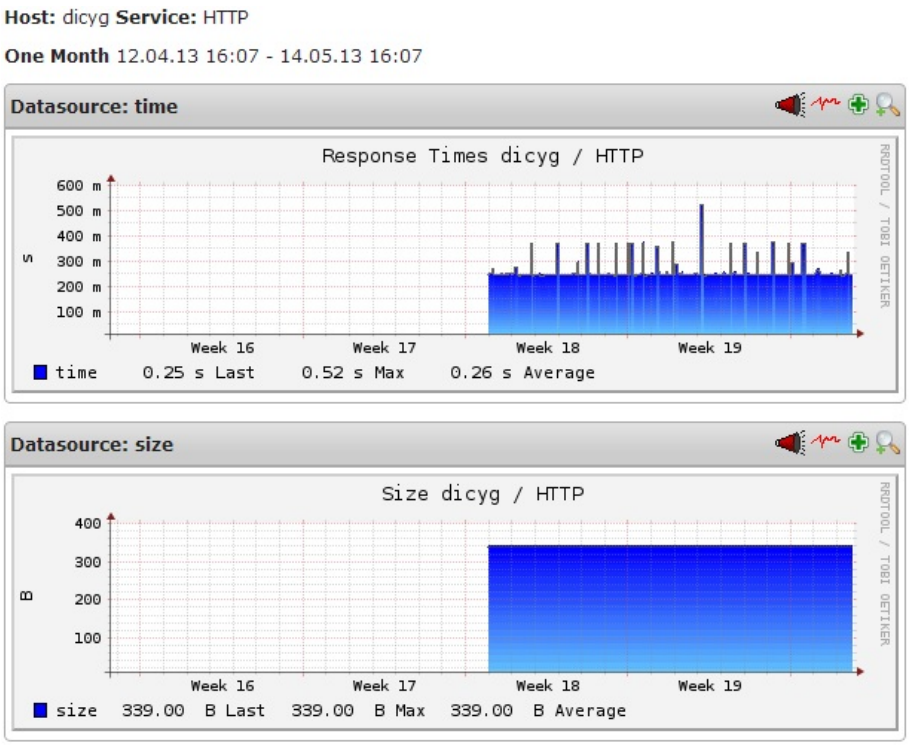


Figura 6.9: Ejemplo de gráfica del servicio PING en el servidor *dicyg*: último mes

La siguiente gráfica es un ejemplo de la monitorización del servicio HTTP del servidor *dicyg* en un intervalo de tiempo de un año:

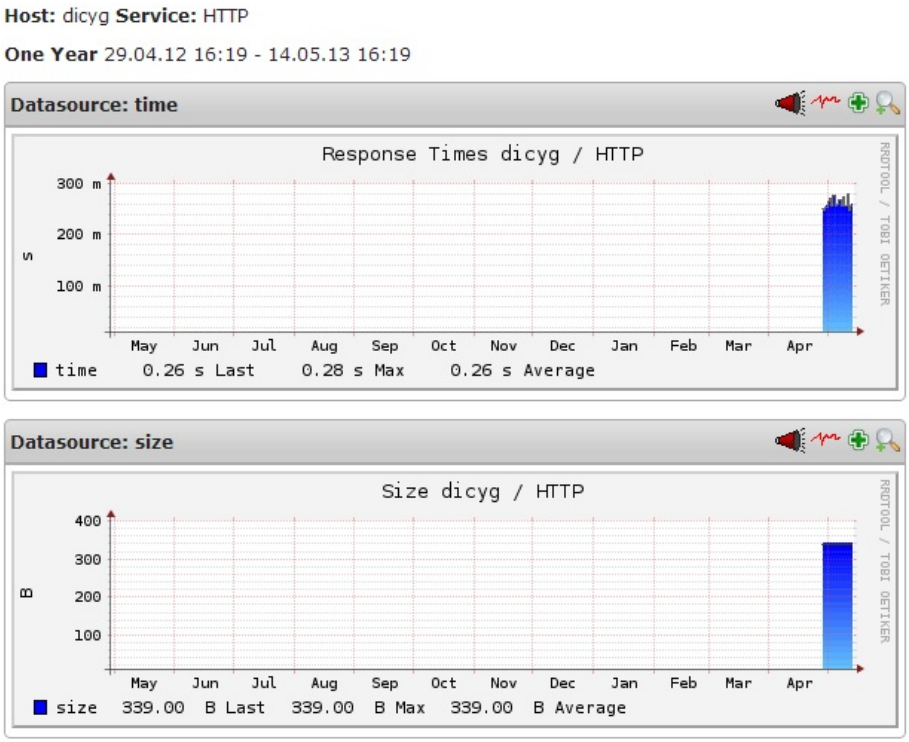


Figura 6.10: Ejemplo de gráfica del servicio PING en el servidor *dicyg*: último año

En la gráfica del servicio HTTP *time* se puede observar que hay picos, lo que significa que el servidor *dicyg* ha tardado tiempo en procesar el archivo durante la conexión HTTP establecida con el servidor *dicyg*. Esto puede resultar en que el servidor tiene mucha carga (hay muchas conexiones hacia el servidor *dicyg* en ese momento).

Una situación más que podría pasar es que en la gráfica *time* no se observe valor alguno, eso significaría que en el servidor *dicyg* el servidor *web* no esta funcionando.

## 6.3. Comprobación del Servicio MySQL

Para la monitorización del servicio MySQL establecemos una conexión a la Base de Datos y lo que obtenemos es un estado de OK en caso de que la conexión se establezca.

La información proporcionada por la conexión establecida es la siguiente:

- Versión del Sistema Operativo
- Hilos Conectados
- Tiempo de Actividad Ininterrumpida de la Base de Datos

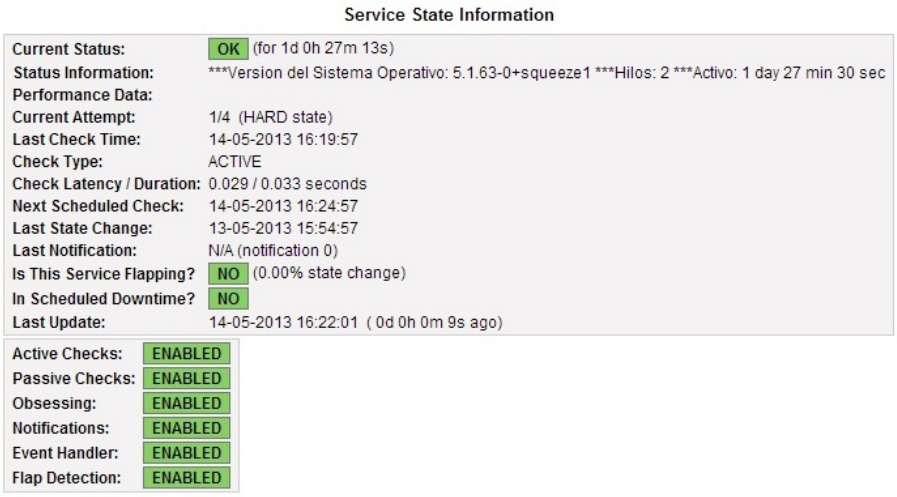


Figura 6.11: Ejemplo de comprobación del servicio MySQL en el servidor *dicyg*

# Conclusiones

*La conclusión es que sabemos muy poco y sin embargo es asombroso lo mucho que conocemos. Y más asombroso todavía que un conocimiento tan pequeño pueda dar tanto poder.*

Bertrand Russell (1872-1970) Filósofo, matemático y escritor británico.

# Conclusiones

Mediante el uso de la herramienta Nagios se logró una comprobación constante de los servicios (HTTP y MySQL) y de la disponibilidad del servidor de la DICYG cumpliendo los objetivos planteados en la presente tesis.

Nagios es una herramienta que se usa para monitorizar *hosts*, servicios, procesos y características importantes de la red que se utilizan día a día en Internet y/o redes internas, comprobando el estado de los dispositivos y/o características de la red que para el administrador de la red sean importantes. Con Nagios se puede obtener la información de interés con el objetivo de prevenir fallas o caídas de servicios y servidores, y en caso de tener alguna incidencia reaccionar de manera oportuna para solucionar el problema.

Para lograr la monitorización de los servicios y la disponibilidad del servidor se realizaron las respectivas configuraciones en la herramienta Nagios para cada uno de los servicios y *hosts*.

Definiendo cada uno de los servicios y cómo iban a ser tratados en caso de fallo.

Para eficientar la prevención de incidencias y hacer más sencillo el trabajo de monitorización Nagios proporcionó información y gráficas sobre el comportamiento de los servicios y disponibilidad del servidor *dicyg*, de esta manera se pudo comprobar el correcto funcionamiento de los servicios y disponibilidad del servidor *dicyg* respectivamente, todo lo anterior para prevenir interrupciones y que en caso de presentarse una, reaccionar de manera oportuna.

De acuerdo a la configuración realizada a la herramienta Nagios, en caso de incidencia, ésta notificó al administrador de la red vía correo electrónico para que éste realizara las acciones necesarias para corregir los fallos.

Nagios es capaz de ejecutar un *script* con una serie de comandos, por lo que en caso de interrupción en los servicios de HTTP y MySQL Nagios intenta restablecer el servicio y en caso de que en las comprobaciones el estado de alguno de los servicio llegue a *CARITICAL HARD* notifica al administrador de la red sobre la interrupción. En caso de que el servicio se restablezca se le notifica al administrador.

Cuando Nagios no es capaz de restablecer los servicios se recomienda seguir el Plan de Contingencia planteado para el restablecimiento oportuno de los servicios.

Con este proyecto se asegura una reacción oportuna para solucionar los fallos que se presenten en los servicios y el servidor utilizados por la DICYG con los cuales proporciona servicios a la comunidad universitaria, personal académico y alumnos.

### *Retos y Recomendaciones*

- Agregar como modo de notificación el envío de mensajes SMS al administrador.
- Como continuación al proyecto que se presenta en esta tesis, se podría hacer extensivo a todos los equipos usados en la DICYG.
- Mejorar la seguridad del proyecto.
- El planteamiento de un Plan de Contingencia generalizado para la DICYG.

# Glosario

**-A-**

**Agujero de Seguridad** es una vulnerabilidad de un sistema de información que permite, mediante su explotación, violar la seguridad del sistema.

**-B-**

**Backbone** se refiere a las principales conexiones en una instalación de red de área local que sigue la normativa de cableado estructurado.

**-D-**

**Datagrama** Agrupamiento lógico de información enviada como unidad de capa de red a través de un medio de transmisión sin establecer previamente un circuito virtual. Los datagramas IP son las unidades principales de información de la Internet. Los términos trama, mensaje, paquete y segmento también se usan para describir agrupamientos de información lógica en las diversas capas del modelo de referencia OSI y en varios círculos tecnológicos.

**Default** en informática, se refiere a un ajuste o un valor que se asigna automáticamente a una aplicación, programa de software o dispositivo, la cual no requirió de la intervención del usuario.

**Demonio** es un tipo especial de proceso informático no interactivo, se ejecuta en segundo plano en vez de ser controlado directamente por el usuario.

**Disposiciones Legales** hipótesis normativas dentro de un marco legal.

**DNS** Domain Name Server (en español: Servidor de Nombres de Dominio). Su función más importante, es traducir (resolver) nombres inteligibles para las personas en identificadores asociados con los equipos conectados a la red, esto con el propósito de poder localizar y direccionar estos equipos mundialmente.

**DSA** *Digital Signature Algorithm* (en español Algoritmo de Firma Digital), este algoritmo como su nombre lo indica, sirve para firmar y no para cifrar información.

**-F-**

**FDDI** *Fiber Distributed Data Interface* (en español: Interfaz de Datos Distribuida por Fibra) es un conjunto de estándares ISO y ANSI para la transmisión de datos en redes de computadoras de área extendida o local (LAN) mediante cable de fibra óptica.

**Firewall** (en español cortafuegos) es una parte de un sistema o una red que está diseñada para bloquear el acceso no autorizado, permitiendo al mismo tiempo comunicaciones autorizadas.

**FTP** (siglas en inglés de *File Transfer Protocol*, 'Protocolo de Transferencia de Archivos') es un protocolo de red para la transferencia de archivos entre sistemas conectados a una red TCP (*Transmission Control Protocol*), basado en la arquitectura cliente-servidor. Desde un equipo cliente se puede conectar a un servidor para descargar archivos desde él o para enviarle archivos, independientemente del sistema operativo utilizado en cada equipo.

**-H-**

**Host** se refiere a las computadoras conectadas a una red, que proveen y utilizan servicios de ella.

**HTTP** *Hypertext Transfer Protocol* (en español Protocolo de Transferencia de Hipertexto) es el protocolo usado en cada transacción de la World Wide Web. Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor.

**-I-**

**ICMP** *Internet Control Message Protocol* (en español: Protocolo de Mensajes de Control de Internet) es el sub protocolo de control y notificación de errores del Protocolo de Internet (IP). Se usa para enviar mensajes de error, indicando por ejemplo que un servicio determinado no está disponible o que un *router* o *host* no puede ser localizado.

**Interoperabilidad** El Instituto de Ingenieros Eléctricos y Electrónicos (IEEE) define interoperabilidad como la habilidad de dos o más sistemas o componentes para intercambiar información y utilizar la información intercambiada.

**-L-**

**Lenguaje de Programación** se define como un idioma artificial diseñado para expresar procesos que pueden ser llevadas a cabo por máquinas como las computadoras.

**Link Simbólico** en sistemas unix o linux, indica un acceso a un directorio o archivo que se encuentra en un lugar distinto dentro de la estructura de directorios. Una modificación realizada utilizando el *link* se reflejará en el archivo original. Y si se elimina el *link simbólico*, no se eliminará el archivo original.

**Localhost** es un nombre reservado que tienen todas las computadoras o dispositivos. El nombre *localhost* es traducido como la dirección IP de *loopback* 127.0.0.1 en IPv4.

**-M-**

**Macro** es una serie de instrucciones que se almacenan para que se puedan ejecutar de manera secuencial mediante una sola llamada u orden de ejecución. Permite la automatización de tareas repetitivas.

**Marco Legal** proporciona las bases sobre las cuales las instituciones construyen y determinan el alcance y naturaleza de la participación política. En el marco legal regularmente se encuentran en un buen número de provisiones regulatorias y leyes interrelacionadas entre sí.

**Modem** es un dispositivo que sirve para enviar una señal llamada moduladora mediante otra señal llamada portadora.

**-N-**

**NEB** *Nagios Event Broker* es la forma en la que Nagios pone disponible información interna a librerias externas.

**Networking** se define como el diseño, la creación o la utilización de una red informática.

**Nodo** En redes de computadoras cada una de las máquinas es un nodo, y si la red es Internet, cada servidor constituye también un nodo.

**-P-**

**Paquete de Datos** Un paquete de datos es una unidad fundamental de transporte de información en las redes de computadoras. El término datagrama es usado a veces como sinónimo.

**Ping** es una utilidad que comprueba el estado de la comunicación del host local con uno o varios equipos remotos de una red TCP/IP por medio del envío de paquetes ICMP. Mediante esta utilidad puede diagnosticarse el estado, velocidad y calidad de una red determinada.

**Plugin** Un complemento es una aplicación que se relaciona con otra para aportarle una función nueva y generalmente muy específica. También se conoce como *plug-in*, *add-on*, conector o extensión.

**POP** un *point-of-presence* es un punto de acceso desde un lugar hacia Internet.

**Protocolo** es un conjunto de reglas y normas que permiten que dos o más entidades de un sistema de comunicación se comuniquen entre ellos para transmitir información por medio de cualquier tipo de variación de una magnitud física. También se puede definir como las reglas o el estándar que define la sintaxis, semántica y sincronización de la comunicación, así como posibles métodos de recuperación de errores.

**Puerto** es una forma genérica de denominar a una interfaz a través de la cual los diferentes tipos de datos se pueden enviar y recibir. Dicha interfaz puede ser de tipo físico, o puede ser a nivel de *software*, en cuyo caso se usa frecuentemente el término puerto lógico.

**PDU** *Protocol Data Unit* (en español Unidad de Datos del Protocolo). La forma que adopta una sección de datos en cualquier capa se denomina PDU. Durante la encapsulación, cada capa encapsula las PDU que recibe de la capa anterior de acuerdo con el protocolo que utilice. En cada etapa del proceso, una PDU tiene un nombre distinto para reflejar su nuevo aspecto.

**-R-**

**RSA** *Rivest, Shamir y Adleman* es un sistema criptográfico de clave pública. Algoritmo utilizado para cifrar y firmar información digitalmente.

**RTT (*Round Trip Times*)** es el tiempo que tarda un paquete de datos enviado desde un emisor en volver a este mismo emisor habiendo pasado por el receptor de destino.

**-S-**

**Script** es un programa usualmente simple, que por lo regular se almacena en un archivo de texto plano.

**Segmento** se le llama a los paquetes de bits que constituyen las unidades de datos del protocolo TCP

**Servidor Web** es un programa informático que procesa una aplicación del lado del servidor realizando conexiones bidireccionales y/o unidireccionales y síncronas o asíncronas con el cliente, generando o cediendo una respuesta en cualquier lenguaje o aplicación del lado del cliente.

**SFTP** *SSH File Transfer Protocol* (también conocido como SFTP o *Secure File Transfer Protocol*) es un protocolo del nivel de aplicación que proporciona la funcionalidad necesaria para la transferencia y manipulación de archivos sobre un flujo de datos fiable.

**Socket** se llama *socket* cuando dos programas pueden intercambiar cualquier flujo de datos, generalmente de manera fiable y ordenada.
También es usado para referir a una interfaz de programación de aplicaciones (API) para la familia de protocolos de Internet TCP/IP, provista usualmente por el sistema operativo. Los *sockets* de Internet constituyen el mecanismo para la entrega de paquetes de datos provenientes de la tarjeta de red a los procesos apropiados. Un *socket* queda definido por un par de direcciones IP local y remota, un protocolo de transporte y un par de números de puerto local y remoto.

**Spam** se llama *spam*, correo basura o mensaje basura a los mensajes no solicitados, no deseados o de remitente no conocido (correo anónimo), habitualmente de tipo publicitario, generalmente enviados en grandes cantidades (incluso masivas) que perjudican de alguna o varias maneras al receptor.

**Spyware** es un software que recopila información de un ordenador y después transmite esta información a una entidad externa sin el conocimiento o el consentimiento del propietario del ordenador.

**-T-**

**Tasa de Actualización** Es una medida de tiempo en la cual una pagina *web* ralizara una actualización de la información que muestra.

**Template** Plantilla

**TFTP** *Trivial File Transfer Protocol* (Protocolo de Transferencia de Archivos Trivial) es un protocolo de transferencia muy simple semejante a una versión básica de FTP. TFTP a menudo se utiliza para transferir pequeños archivos entre computadoras en una red.

**TIC** Se encargan del estudio, desarrollo, implementación, almacenamiento y distribución de la información mediante la utilización de *hardware* y *software* como medio de sistema informático.

**Tunneling** consiste en encapsular un protocolo de red sobre otro (protocolo de red encapsulador) creando un túnel dentro de una red de computadoras. El establecimiento de dicho túnel se implementa incluyendo una PDU determinada dentro de otra PDU con el objetivo de transmitirla desde un extremo al otro del túnel sin que sea necesaria una interpretación intermedia de la PDU encapsulada. De esta manera se encaminan los paquetes de datos sobre nodos intermedios que son incapaces de ver en claro el contenido de dichos paquetes. El túnel queda definido por los puntos extremos y el protocolo de comunicación empleado, que entre otros, podría ser SSH.

**-U-**

**UDP** es un protocolo del nivel de transporte basado en el intercambio de datagramas (Encapsulado de capa de transporte Modelo TCP/IP). Permite el envío de datagramas a través de la red sin que se haya establecido previamente una conexión, ya que el propio datagrama incorpora suficiente información de direccionamiento en su cabecera. Tampoco tiene confirmación ni control de flujo, por lo que los paquetes pueden adelantarse unos a otros; y tampoco se sabe si ha llegado correctamente, ya que no hay confirmación de entrega o recepción.

**-V-**

**VPN** *Virtual Private Network* (en español Red Privada Virtual), es una tecnología de red que permite una extensión segura de la red local sobre una red pública o no controlada como Internet. Permite que la computadora en la red envíe y reciba datos sobre redes compartidas o públicas como si fuera una red privada con toda la funcionalidad, seguridad y políticas de gestión de una red privada.

**-W-**

**World Wide Web** Red informática mundial o (Internet) es un sistema de distribución de información basado en hipertexto. Con un navegador *web*, un usuario visualiza sitios *web* compuestos de páginas *web* que pueden contener texto, imágenes,vídeos u otros contenidos multimedia, y navegar a través de esas páginas usando hiperenlaces (*links*).

# Anexos

```
##############################################################
#
# CGI.CFG - CONFIGURACIÓN DEL ARCHIVO CGI PARA NAGIOS 3.4.3
#
# Ultima Modificación: 03-06-2013
#
##############################################################

# MAIN CONFIGURATION FILE
main_config_file=/nagios/etc/nagios.cfg

# PHYSICAL HTML PATH
physical_html_path=/nagios/share

# URL HTML PATH
url_html_path=/nagios
...

# AUTHENTICATION USAGE
use_authentication=1
...

# SYSTEM/PROCESS INFORMATION ACCESS
authorized_for_system_information=nagiosadmin

# CONFIGURATION INFORMATION ACCESS
authorized_for_configuration_information=nagiosadmin

# SYSTEM/PROCESS COMMAND ACCESS
authorized_for_system_commands=nagiosadmin

# GLOBAL HOST/SERVICE VIEW ACCESS
authorized_for_all_services=nagiosadmin
authorized_for_all_hosts=nagiosadmin

# GLOBAL HOST/SERVICE COMMAND ACCESS
authorized_for_all_service_commands=nagiosadmin
authorized_for_all_host_commands=nagiosadmin
...

# REFRESH RATE
refresh_rate=90
...
```

```
##############################################################################
# COMMANDS.CFG - DEFINICIÓN DE COMANDOS PARA NAGIOS 3.4.3
#
# Ultima Modificación: 03-06-2013
#
##############################################################################

########
### COMANDOS DE NOTIFICACIÓN
########

# Definición del comando para notificar por correo debiso a falla en un host
(notify-host-by-email)
define command{
    command_name    notify-host-by-email
    command_line    /usr/bin/printf "%b" "***** Nagios *****\n\nNotification Type:
    $NOTIFICATIONTYPE$\nHost: $HOSTNAME$\nState: $HOSTSTATE$\nAddress: $HOSTADDRESS$\nInfo:
    $HOSTOUTPUT$\n\nDate/Time: $LONGDATETIME$\n" | /usr/bin/mail -s "** $NOTIFICATIONTYPE$
    Host Alert: $HOSTNAME$ is $HOSTSTATE$ **" $CONTACTEMAIL$
    }

# Definición del comando para notificar por correo debiso a falla en un host
(notify-service-by-email)
define command{
    command_name    notify-service-by-email
    command_line    /usr/bin/printf "%b" "***** Nagios *****\n\nNotification Type:
    $NOTIFICATIONTYPE$\n\nService: $SERVICEDESC$\nHost: $HOSTALIAS$\nAddress:
    $HOSTADDRESS$\nState: $SERVICESTATE$\n\nDate/Time: $LONGDATETIME$\n\nAdditional
    Info:\n\n$SERVICEOUTPUT$\n" | /usr/bin/mail -s "** $NOTIFICATIONTYPE$ Service Alert:
    $HOSTALIAS$/$SERVICEDESC$ is $SERVICESTATE$ **" $CONTACTEMAIL$
    }

########
### COMANDO DE COMPROBACIÓN DE HOST
########

# Definición de comando para comprobar la disponibilidad de un host (check-host-alive)
define command{
        command_name    check-host-alive
        command_line    $USER1$/check_ping -H $HOSTADDRESS$ -w 3000.0,80% -c 5000.0,100% -p 5
        }

########
### COMANDOS DE COMPROBACIÓN DE SERVICIOS
########

# Definición de comando para comprobar el servicio HTTP (check_http)
define command{
        command_name    check_http
        command_line    $USER1$/check_http -I $HOSTADDRESS$ $ARG1$
        }

# Definición de comando para comprobar el servicio PING (check_ping)
define command{
        command_name    check_ping
        command_line    $USER1$/check_ping -H $HOSTADDRESS$ -w $ARG1$ -c $ARG2$ -p 5
        }
```

```
# Definición de comando para comprobar el servicio MySQL (check_mysql.pl)
define command{
        command_name    check_mysql
        command_line    perl $USER1$/check_mysql.pl $HOSTADDRESS$ $ARG1$ $ARG2$
        }

########
### COMANDOS DE DATOS DE RENDIMIENTO (PERFORMANCE DATA)
########

# Definición de comando para procesar los datos de rendimiento de un host
(process-host-perfdata)
define command{
    command_name    process-host-perfdata
    command_line    /usr/bin/perl /usr/local/pnp4nagios/libexec/process_perfdata.pl -d
    HOSTPERFDATA
    }

# Definición de comando para procesar los datos de rendimiento de un servicio
(process-service-perfdata)
define command{
    command_name    process-service-perfdata
    command_line    /usr/bin/perl /usr/local/pnp4nagios/libexec/process_perfdata.pl
    }

########
### COMANDOS EVENT HANDLER
########

# Definición de comando para reiniciar MySQL en la maquina local
define command{
        command_name    reinicia-mysql
        command_line    /bin/sh $USER1$/event_handler $SERVICESTATE$ $SERVICESTATETYPE$
        $SERVICEATTEMPT$
}

# Definición de comando para reiniciar MySQL en el servidor jesus
define command{
        command_name    reinicia-jesus
        command_line    /bin/sh $USER1$/eventhandler_jesus $SERVICESTATE$ $SERVICESTATETYPE$
        $SERVICEATTEMPT$
}
```

```
##############################################################################
# CONTACTS.CFG - DEFINICIÓN DE CONTACTOS Y GRUPOS DE CONTACTOS
#
# Ultima Modificación: 03-06-2013
#
##############################################################################

########
### CONTACTOS
########

# Definición del contacto nagiosadmin

define contact{
        contact_name                    nagiosadmin
        use                             generic-contact
        alias                           Nagios Admin
        email                           ulises@mail.com
        }

# Definición del contacto ulises
define contact{
        contact_name                    ulises
        use                             generic-contact
        alias                           Ulises Admin
        email                           ulises2@mail.com
        }

########
### GRUPO DE CONTACTOS
########

# Definición del grupo de contactos admins

define contactgroup{
        contactgroup_name       admins
        alias                   Nagios Administrators
        members                 nagiosadmin,ulises
        }
```

```
##############################################################################
# LOCALHOST.CFG - ARCHIVO DE CONFIGURACIÓN PARA LA MONITORIZACIÓN
#
# Ultima Modificación: 03-06-2013
#
##############################################################################

########
### DEFINICIÓN DE HOSTS
########

# Definición del localhost
define host{
        use                     linux-server
        host_name               localhost
        alias                   localhost
        address                 127.0.0.1
        check_command           check-host-alive
        process_perf_data       1
        }

# Definición del host dicyg
define host{
        use                     linux-server
        host_name               dicyg
        alias                   test1
        address                 192.168.1.1
        check_command           check-host-alive
        process_perf_data       1
        }

# Definición del host jesus
define host{
        use                     linux-server
        host_name               jesus
        alias                   test2
        address                 192.168.1.2
        check_command           check-host-alive
        process_perf_data       1
        }

# Definición del host gateway
define host{
        use                     linux-server
        host_name               gateway
        alias                   test3
        address                 192.168.1.254
        check_command           check-host-alive
        process_perf_data       1
        }
```

```
# Definición del host 110.inverso.unam.mx
define host{
        use                     linux-server
        host_name               110.inverso.unam.mx
        alias                   test4
        address                 192.168.1.3
        check_command           check-host-alive
        process_perf_data       1
        }

########
### DEFINICIÓN DE UN GRUPO DE HOSTS
########

# Definición de un grupo de hosts (Linux)
define hostgroup{
        hostgroup_name  linux-servers
        alias           Linux Servers
        members         localhost
        }

########
### DEFINICIÓN DE SERVICIOS
########

# Definición del servicio PING en localhost
define service{
        use                             local-service,srv-pnp
        host_name                       localhost
        service_description             PING
        check_command                   check_ping!100.0,20%!500.0,60%
        }

# Definición del servicio PING en dicyg
define service{
        use                             local-service,srv-pnp
        host_name                       dicyg
        service_description             PING
        check_command                   check_ping!100.0,20%!500.0,60%
        }

# Definición del servicio PING en jesus
define service{
        use                             local-service,srv-pnp
        host_name                       jesus
        service_description             PING
        check_command                   check_ping!100.0,20%!500.0,60%
        }

# Definición del servicio PING en gateway
define service{
        use                             local-service,srv-pnp
        host_name                       gateway
        service_description             PING
        check_command                   check_ping!100.0,20%!500.0,60%
        }
```

```
# Definición del servicio PING en 110.inverso.unam.mx
define service{
        use                             local-service,srv-pnp
        host_name                       110.inverso.unam.mx
        service_description             PING
        check_command                   check_ping!100.0,20%!500.0,60%
        }

# Definición del Servicio MySQL en jesus
define service{
        use                             local-service
        host_name                       jesus
        service_description             MYSQL
        check_command                   check_mysql!<user>!<contraseña>
        event_handler                   reinicia-jesus
}

# Definición del Servicio MySQL en localhost
define service{
        use                             local-service
        host_name                       localhost
        service_description             MYSQL
        check_command                   check_mysql!<user>!<contraseña>
        event_handler                   reinicia-mysql
}

# Definición del Servicio MySQL en dicyg
define service{
        use                             local-service
        host_name                       dicyg
        service_description             MYSQL
        check_command                   check_mysql!<user>!<contraseña>
}

# Definición del Servicio HTTP en localhost
define service{
        use                             local-service,srv-pnp
        host_name                       localhost
        service_description             HTTP
        check_command                   check_http
    }

# Definición del Servicio HTTP en dicyg
define service{
        use                             local-service,srv-pnp
        host_name                       dicyg
        service_description             HTTP
        check_command                   check_http
        }

# Definición del Servicio HTTP en jesus
define service{
        use                             local-service,srv-pnp
        host_name                       jesus
        service_description             HTTP
        check_command                   check_http
        }
```

```
##############################################################################
#
# NAGIOS.CFG - ARCHIVO PRINCIPAL DE CONFIGURACIÓN DE NAGIOS 3.4.3
#
# Ultima Modificación: 03-06-2013
#
##############################################################################

# LOG FILE
log_file=/usr/local/nagios/var/nagios.log

# OBJECT CONFIGURATION FILE(S)
cfg_file=/nagios/etc/objects/commands.cfg
cfg_file=/nagios/etc/objects/contacts.cfg
cfg_file=/nagios/etc/objects/timeperiods.cfg
cfg_file=/nagios/etc/objects/templates.cfg
# Definitions for monitoring the local (Linux) host
cfg_file=/nagios/etc/objects/localhost.cfg
...

# RESOURCE FILE
resource_file=/nagios/etc/resource.cfg
...
# NAGIOS USER
nagios_user=<usuario_definido>

# NAGIOS GROUP
nagios_group=<grupo_definido>

# EXTERNAL COMMAND OPTION
check_external_commands=1

# EXTERNAL COMMAND CHECK INTERVAL
command_check_interval=-1

# EXTERNAL COMMAND FILE
command_file=/nagios/var/rw/nagios.cmd
...

# NOTIFICATIONS OPTION
enable_notifications=1

# EVENT HANDLER USE OPTION
enable_event_handlers=1

# PROCESS PERFORMANCE DATA OPTION
process_performance_data=1

# HOST AND SERVICE PERFORMANCE DATA PROCESSING COMMANDS
host_perfdata_command=process-host-perfdata
service_perfdata_command=process-service-perfdata
...
```

```
##############################################################################
#
# RESOURCE.CFG - ARCHIVO RESOURCE PARA NAGIOS 3.4.3
#
# Ultima Modificación: 03-06-2013
#
##############################################################################

# Configuración de $USER1$ para que sea el path de los plugins
$USER1$=/nagios/libexec

# Configuración de $USER2$ para que sea el path de los event handlers
#$USER2$=/nagios/libexec/eventhandlers

# Almacenamiento de algunos usrios y contraseñas
#$USER3$=someuser
#$USER4$=somepassword
```

```
##############################################################################
# TEMPLATES.CFG - DEFINICIÓN DE PLANTILLAS
#
# Ultima Modificación: 03-06-2013
#
##############################################################################

########
### PLANTILLAS DE CONTACTOS
########

# Plantilla de Contacto (No es un contacto real)
define contact{
        name                            generic-contact
        service_notification_period     24x7
        host_notification_period        24x7
        service_notification_options    w,u,c,r,f,s
        host_notification_options       d,u,r,f,s
        service_notification_commands   notify-service-by-email
        host_notification_commands      notify-host-by-email
        register                        0
        }

########
### PLANTILLAS DE HOSTS
########

# Plantilla de host (No es un host real)
define host{
        name                            generic-host
        notifications_enabled           1
        event_handler_enabled           1
        flap_detection_enabled          1
        failure_prediction_enabled      1
        process_perf_data               1
        retain_status_information       1
        retain_nonstatus_information    1
        notification_period             24x7
        register                        0
        }

# Plantilla de host con diferentes caracteristicas
define host{
    name                        linux-server
    use                         generic-host
    check_period                24x7
    check_interval              5
    retry_interval              1
    max_check_attempts          10
    check_command               check-host-alive
    notification_period         24x7
    notification_interval       120
    notification_options        d,u,r
    contact_groups              admins
    register                    0
    }
```

```
########
### PLANTILLAS DE SERVICIOS
########

# Plantilla de servicio (No es un servicio real)
define service{
        name                            generic-service
        active_checks_enabled           1
        passive_checks_enabled          1
        parallelize_check               1
        obsess_over_service             1
        check_freshness                 0
        notifications_enabled           1
        event_handler_enabled           1
        flap_detection_enabled          1
        failure_prediction_enabled      1
        process_perf_data               1
        retain_status_information       1
        retain_nonstatus_information    1
        is_volatile                     0
        check_period                    24x7
        max_check_attempts              3
        normal_check_interval           10
        retry_check_interval            2
        contact_groups                  admins
        notification_options            w,u,c,r
        notification_interval           60
        notification_period             24x7
         register                       0
        }


# Plantilla de servicio con caracteristicas diferentes (No es un servicio real)
define service{
        name                            local-service
        use                             generic-service
        max_check_attempts              4
        normal_check_interval           5
        retry_check_interval            1
        register                        0
    }

########
###    TEMPLATES PNP4NAGIOS
########

define host {
   name       host-pnp
   action_url /pnp4nagios/index.php/graph?host=$HOSTNAME$&srv=_HOST_
   register   0
}

define service {
   name       srv-pnp
   action_url /pnp4nagios/index.php/graph?host=$HOSTNAME$&srv=$SERVICEDESC$
   register   0
}
```

```
##############################################################################
# TIMEPERIODS.CFG - DIFINICIONES DE PERIODOS DE TIEMPO
#
# Ultima Modificación: 03-06-2013
#
##############################################################################

#24 horas, 7 días de la semana
define timeperiod{
        timeperiod_name 24x7
        alias           24 Hours A Day, 7 Days A Week
        sunday          00:00-24:00
        monday          00:00-24:00
        tuesday         00:00-24:00
        wednesday       00:00-24:00
        thursday        00:00-24:00
        friday          00:00-24:00
        saturday        00:00-24:00
        }

# Horario laboral de 9am - 5pm
define timeperiod{
    timeperiod_name workhours
    alias       Normal Work Hours
    monday      09:00-17:00
    tuesday     09:00-17:00
    wednesday   09:00-17:00
    thursday    09:00-17:00
    friday      09:00-17:00
    }

# Sin definición (No monitorizar)
define timeperiod{
    timeperiod_name none
    alias       No Time Is A Good Time
    }
```

# Referencias


[1] *IEEE Sección México - Historia.* http://www.ieee.org.mx/IEEE/IEEE_Seccion_Mexico_-_Historia.html , [Consultado el: 2013-05-07].

[2] *Nagios - Documentation.* http://nagios.org/documentation, [Consultado el: 2013-05-07].

[3] *Paradigma TI México.* http://www.paradigmait.com.mx/monitoreo.html, [Consultado el: 2013-05-07].

[4] *SSH Overview - Table of Contents.* http://www.vandyke.com/solutions/ssh_overview/index.html, [Consultado el: 2013-05-07].

[5] *The ISO Story.* http://www.iso.org/iso/home/about/the_iso_story.htm, [Consultado el: 2013-05-07].

[6] *Configurar Postfix para enviar correo desde Gmail.* http://hluisgarcia.es/wordpress/2012/07/configurar-postfix-para-enviar-correo-desde-gmail/, [Consultado el: 2013-05-09].

[7] *Event Handlers.* http://nagios.sourceforge.net/docs/nagioscore/3/en/eventhandlers.html, [Consultado el: 2013-05-09].

[8] *Nagios Core - Features.* http://assets.nagios.com/datasheets/nagioscore/Nagios, [Consultado el: 2013-05-09].

[9] *Nagios Core Version 3.x Documentation.* http://nagios.sourceforge.net/docs/nagioscore-3-en.pdf, [Consultado el: 2013-05-09].

[10] *PandoraFMS - Características.* http://pandorafms.com/downloads/hojaproductoES.pdf , [Consultado el: 2013-05-09].

[11] *Postfix Feature Overview.* http://www.postfix.org/features.html, [Consultado el: 2013-05-09].

[12] *RFC - Simple Mail Transfer Protocol.* http://datatracker.ietf.org/doc/rfc2821/?include_text=1, [Consultado el: 2013-05-09].

[13] *Servidor de Correo.* http://servilinux.galeon.com/, [Consultado el: 2013-05-09].

[14] *The Postfix Home Page.* http://www.postfix.org/start.html, [Consultado el: 2013-05-09].

[15] *Zenoss Features - General.* http://swik.net/zenoss/Features , [Consultado el: 2013-05-09].

[16] *Zenoss Features - Hardware Requeriments.* http://community.zenoss.org/docs/DOC-7387 , [Consultado el: 2013-05-09].

[17] *ANSI: Historical Overview.* http://www.ansi.org/about_ansi/introduction/history.aspx?menuid=1# [Consultado el: 2013-05-27].

[18] *EIA - Electronics Industries Association.* http://redes-utp-007.blogspot.mx/2012/04/eia-electronics-industry-association.html, [Consultado el: 2013-05-27].

[19] *Object Definitions.* http://nagios.sourceforge.net/docs/nagioscore/3/en/objectdefinitions.html#host, [Consultado el: 2013-05-30].

[20] *Historia IEEE.* http://www.ewh.ieee.org/r8/germany/ias-pels/m_erlangen/IEEEHistory.pdf, [Consultado el: 2013-10-27].

[21] *Instituto Federal de Telecomunicaciones - Quienes Somos.* http://www.ift.org.mx/iftweb/informacion-general/, [Consultado el: 2013-10-27].

[22] *Página de la DICYG - Quienes Somos.* http://dicyg.fi-c.unam.mx:8080/Site/quienes-somos, [Consultado el: 2013-10-27].

[23] CAYUQUEO, Sergio. *Manual:Nagios.* http://wiki.cayu.com.ar/doku.php?id=manuales:nagios, [Consultado el: 2013-05-07].

[24] CISCO Systems. *Inc., Academia de Networking de Cisco Systems: guía del primer año CCNA 1 y 2.* Pearson Education, 2004.

[25] GÓMEZ, Álvaro, SUÁREZ, Carlos y. *Sistemas de Información: Herramientas prácticas para la gestión.* Alfaomega, 2002.

[26] GÓMEZ LÓPEZ, Julio GIL MONTOYA, Francisco VILLAR FERNÁNDEZ, Eugenio MÉNDEZ CIRERA, Francisco. *Administración Avanzada de Sistemas Informáticos.* Alfaomega, 2010.

[27] MACÍAS RÍOS, Ma. Eugenia. *Apuntes Administración de Redes.* UNAM, FI, 2011.

[28] POYATO, Chelo COLL, Francisco MORENO, David. *Recomendaciones de Seguridad: Definición de una Política de Seguridad.*

[29] VIGUERAS VILLASEÑOR, Marco Antonio. *Apuntes Redes de Datos.* UNAM, FI, 2011.

[30] WEINMAN, William. *El Libro de CGI.* Prentice-Hall Hispanoamericana, 1996.