Monitoria y análisis de Red con Nagios

Las redes de cómputo de las organizaciones, se vuelven cada vez más complejas y la exigencia de la operación es cada vez mas demandante. Las redes, cada vez mas, soportan aplicaciones y servicios estratégicos de las organizaciones. Por lo cual el análisis y monitoreo de redes se ha convertido en una labor cada vez mas importante y de carácter pro-activo para evitar problemas. Con el termino Monitoreo nos referimos a un sistema que constantemente monitorea una red de computadoras para detectar sistemas lentos o en mal funcionamiento y que notifica al administrador de la red en caso de falla vía correo electrónico, sms u otros medios.

Para prevenir errores en un sistema podemos utilizar una aplicación que se ocupe de estar “controlado y observando” el funcionamiento de la red, esto podemos realizarlo por medio de un software llamado Nagios. Nagios es un sistema de monitoreo de servidores, aplicaciones y redes. Comprueba clientes y servicios, especificados alertando en caso de problemas como caídas de servicio ó restauración de los mismos.

Nagios es un sistema de monitorización de equipos y de servicios de red, escrito en C y publicado bajo la GNU General Public License, el lenguage con el cual esta desarrollado nos asegura una rápida ejecución y su licencia que lo determina como Software Libre nos asegura que siempre tendremos actualizaciones disponibles y 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 que administran y conocer los problemas que ocurren en la infraestructura que administran 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 las cosas sucedan; 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 en base a ello como:

- *Conservar y almacene datos de la red para manejar reportes y tendencias*
- *Ver y analizar la red, así como el tráfico de la red a través del tiempo*
- *Monitorear el estado de la red en comparación a los reportes de análisis*
- *Generar reportes sustentados para justificar las necesidades de actualización de la red*

Para facilitar tareas de explotación de datos, hay diferentes aditivos como un visor de reportes integrados, en el cual se puede ver el histórico de actividad y performance de servicios, y además un visor de diagramas de red con el estado actual de cada equipo.

El mismo, esta constituido por un Núcleo que construye la interfaz de usuario y por

plugins los cuales representan los ojos y oídos de Nagios y por lo cual se encargan de recopilar información (bajo demanda). Los mismos pueden estar programados en diversos lenguajes como C, C++, Python, Perl, PHP, Java, Bash etc, ya que Nagios es independiente del lenguaje en el cual que 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.

Objetivos y necesidades

Conocer el estado de diferentes servicios brindados por un conjunto heterogéneo de dispositivos y equipos como servidores corriendo diferentes sistemas operativos, routers de los cuales dependen varios equipos, sistemas SCADA y sistemas de analizadores de humedad/temperatura/gases etc. Obtener información de los mismos como *estado en red*, *tiempo arriba*, *puertos abiertos*, *servicios* y *procesos corriendo*, *carga de CPU*, carga de *memoria física*, carga de *memoria virtual*, *espacio en disco*, *interfaces de red activas*.

Para poder tener esta información se debe establecer un control que asegure el mantenimiento de los dispositivos y se puedan efectuar acciones en forma preventiva o correctiva según corresponda en tiempo y forma ante eventuales anomalías de los servicios.

Es posible conocer los estados y datos de estos diferentes equipos para una posterior elaboración de reportes etc, elaborando una configuración personalizada de Nagios para cada caso en particular, por medio de testeo de paquetes de red, o haciendo uso de diferentes funciones que provee el protocolo SNMP (Simple Network Management Protocol) que nos permite gestionar y/o supervisar datos de diferentes elementos y componentes de la red como routers, switches, servidores etc y al ser un protocolo standard es posible monitorizar una amplia variedad de casos en escenarios con sistemas ó equipos diferentes.

Con lo cual podremos concluir si el sistema :

- Lleva a cabo eficazmente su finalidad
- Utiliza eficientemente los recursos.

Para :

- Optimizar los Procesos
- Reducir Costos Operativos
- Llevar a cabo un mejoramiento del Control de Calidad
  - Minimizar Tiempos ⇒ $$$$

Ya que podemos :

- Detectar de forma sistemática el uso de los recursos y los flujos de información dentro de una organización.
- Determinar qué información es crítica para el cumplimiento de su misión y objetivos, identificando necesidades, duplicidades, costos, valor y barreras, que obstaculizan flujos de información eficientes.
- Análizar de eficiencia del sistema.

- Verificar el cumplimiento de Normativas.
- Revisión de la gestión de recursos.

Con esto podemos elaborar informes, responder ante evaluaciones externas y documentar la evaluación para reflejar el desarrollo y los resultados de la misma pero además vamos a :

- Fortalecer bases de información para grupos y personas de apoyo que trabajan con los sistemas.

Puntos principales a tener en cuenta :

- Aplicación estable para entornos productivos.
- Licencia de código abierto.
- Debe monitorear equipos idenpendientemente de sistemas operativos : Windows, Linux, Unix, CISCO iOS.
- Generar alertas cuando se identifican incidencias.
- Los datos se deben poder exportar para su posterior análisis.
- El sistema debe poder trabajar tanto con agentes instalados en los equipos clientes como sin ellos.
- Poder generar complementos (plugs in) independientemente del lenguaje de programación o ubicación de los mismos.
- Documentación suficiente y clara disponible del sistema de monitoreo.
- El sistema debe tener una comunidad que lo respalde y preferentemente empresas.
- El sistema debe ser muy conocido o utilizado. Existencia de empresas clientes o usuarios a los que se puede referenciar.
- Actualizaciones regulares.

Como adicional debemos cubrir :

- Necesidades de Automatización, Instrumentación y Optimización de los procesos industriales.

Gestión proactiva para alcanzar los objetivos organizacionales

Diseñar e implementar procesos operativos y administrativos eficaces aplicables a la Gestión de la Red de TI.

La **Gestión de Problemas** puede ser:

- **Reactiva:** Analiza los incidentes ocurridos para descubrir su causa y propone soluciones a los mismos.
- **Proactiva:** Monitorea la calidad de la infraestructura TI. Analiza los Registros de Incidentes y configuraciones utilizando datos de otros procesos de Gestión del Servicio de TI para identificar tendencias o problemas significativos con el objetivo de prevenir incidentes incluso antes de que estos ocurran.

El meta a llegar para mejorar la eficiencia de un Área de TI, es poder ir adoptando una metodología de gestión proactiva de recursos.

Relevamiento y diseño del modelo

El relevamiento es la herramienta principal a desarrollar durante la fase de

Planificación de un sistema integral de monitoreo, para que este a su vez pueda dar datos fieles de como se ven afectados los procesos tanto operativos y de apoyo, como los procesos esenciales de producción y de negocio.

Infraestructura de equipamiento, software de base y comunicaciones

Para conocer la base donde se apoyaran los servicios de TI se deberá analizar lo siguiente :

- Software de base, desarrollo y comunicaciones.
- La capacidad del equipamiento para absorber, en situaciones de exigencia, las operaciones del proceso que están automatizadas y el crecimiento permitido para incorporar las faltantes.
- Las comunicaciones instaladas y sus características técnicas (señalización, protocolos, software de red, etc.). Se relevarán las redes LAN y las WAN, como así también las posibilidades de instalación de áreas no cubiertas en las que se ejecuten los procesos analizados.
- Las ubicaciones geográficas y direcciones de los distintos inmuebles en los que se realizan los procesos con indicación de distancias entre ellos.
  - Estado general de las instalaciones.
- Los costos por servicios informáticos y de comunicaciones (Ej.: mantenimiento de hardware o software, comunicaciones, licencias de software de base, alquiler, leasing o financiamiento de equipamiento, etc.

Software de aplicación

Se puede comenzar a partir de un un inventario del software o desarrollarlo contenga como mínimo :

- Denominación del sistema o subsistema.
- Antigüedad promedio.
- Cantidad de aplicaciones que lo integran.
- Desarrollo propio o de terceros, indicando si la Organización tiene capacidad de automantenimiento.
- Lenguaje de desarrollo.
- Esfuerzo de mantenimiento exigido.
- Objeto y alcance: eventos o actividades del proceso que automatiza.
- Productos que brinda
- Documentación funcional y técnica.
- Estimación en equivalencia de tiempo completo de la cantidad de personas afectadas al desarrollo, mantenimiento y operación del sistema.
- Costo de licencias y de mantenimiento, en caso de contratos con terceros.

Capacidad de gestión de los servicios informáticos

Capacidad de respuesta de los recursos humanos integrantes del servicio informático para afrontar los requerimientos de información que se plantean en la ejecución de los procesos estudiados.

Se constatará si existen normas, métodos, procedimientos y el nivel de retraso respecto de requerimientos de mantenimiento o nuevos sistemas.

Estimar los costos de los servicios informáticos

Los principales insumos para la determinación del costo de los servicios informáticos son:

- Contratos informáticos por alquiler, leasing o compra financiada de : equipamiento, software de base y comunicaciones.

Si la infraestructura es propiedad de la Organización se estimarán los costos de reposición para aplicar cuotas de amortización para un período de *n* años (plazo que se estima razonable para la depreciación de un sistema de información automatizado) y su antigüedad actual promedio.

- Contratos de alquiler de licencias de software aplicativo.
- Sueldos y costos operativos de los empleados del servicio informático.
- Insumos mensuales consumidos por el equipo humano de trabajo de TI (impresiones, tarjetas de credito corporativas, gastos de transporte etc).
- Servicios varios (aire acondicionado del datacenter, suministro eléctrio etc).

Cada uno de los ítems relevados deberá ser prorrateado conforme a la base de distribución que se considere más adecuada. Por ejemplo, para distribuir los costos de personal se puede utilizar el procedimiento equivalente de tiempo completo (ETC). Si una persona que trabaja 200 horas mensuales le dedica el 10% de su tiempo a mantener un sistema relacionado al proceso en estudio, estará afectando el ETC de 20 horas de trabajo. Luego, si su sueldo más leyes sociales es de **$XX** por mes, resultará que el costo proporcional de dicha actividad es de **$NN**.

En otras situaciones, como alquileres de inmuebles, la base de prorrateo puede ser metros cuadrados ocupados.

Identificar las soluciones informáticas desarrolladas por terceros

- Nivel de automatización de los procesos sometidos a análisis y grado de colaboración para una solución integral de gestión..
- Portabilidad de la solución respecto de plataformas de equipamientos y sistemas operativos.
- Infraestructura de la empresa y cantidad de productos instalados.
- Facilidad de instalación y requisitos de entrenamiento del personal que seguirá con el mantenimiento.
- Nivel de parametrización.
- Facilidad de uso.
- Calidad de la documentación disponible.
- Flexibilidad ante la necesidad de ajustes.
- Frecuencia de aparición de nuevas versiones.

Herramientas principales para el análisis

- Mapa de sistemas.
- Relevamiento de infraestructura informática y de comunicaciones.

    - Inventario de infraestructura informática y de comunicaciones (Activos de TI, Hardware, Software, Aplicaciones etc).
- Informe preliminar de soluciones informáticas desarrolladas por terceros.
- Informe de costos informáticos y de comunicaciones.

Herramientas y productos principales a utilizar

- Gestión de Inventario
- CMDB
- Sistema de administración de Proyectos

Análisis de costos y relación con Disponibilidad de Servicio - SLA

A la hora de contratar un servicio de Data Center por lo general se hace para suplir varias carencias que se pueden dar en la organización que si bien a veces esa carencia es de infraestructura y de inversión monetaria, aunque muchas otras veces es por la imposibilidad de disponibilizar tiempo y recursos humanos en el proyecto de implementación del mismo.

Cuando uno contrata un servicio de Data Center hay varios factores a tener en cuenta que deben ser provistos y desarrollados por el prestador del servicio y a la hora de ofrecer el mismo, esos factores suelen afectar directamente en su **costo** :

- Costo determinado por varios factores :
    - Disponibilidad
    - Redundancia
    - Escalabilidad
    - Eficiencia
- Servicios extras con interdependencia y relación de unos con otros :
    - Backup administrador por el proovedor
    - Servicio manos y ojos
    - Alertas proactivas 7×24
- Descuentos aleatorios

Todos estos factores deben estar esclarecidos ya que no solo puede existir la necesidad de asegurarle y garantizarle efectivamente esto a los utilizadores externos, sino también a los internos, ya que la Disponibilidad es el factor determinante de la contratación del servicio. Por lo general se establecen acuerdos de Disponibilidad de Servicio - SLA, con diferentes niveles y con un accionar y escalamiento determinado para cada caso.

Lo comentado anteriormente en la mayoría de los casos y por lógica y sentido común suele encuadrarse dentro de un marco legal por medio de un contrato y dentro del mismo suele haber cláusulas por multas monetarias debido a tiempos de indisponibilidad o mal funcionamiento del servicio. El proceso de costeo para realizar la calificación de incidentes y su posterior valorización monetaria debe estar detallado en una norma que regle dicho procedimiento definido en la contratación del servicio.

Por y para eso para nosotros es necesario como proovedores de servicio que somos y al estar abasteciendo a otras empresas con exigencias necesarias para el desarrollo de su actividad, **en este caso, la base material para correr su infraestructura informática** :

Que tengamos un registro minucioso y reportado de eventos ya que estos pueden afectarnos directamente de forma legal y monetaria en el desarrollo de nuestras actividades sino llevamos control de los mismos. Para a su vez así poder justificar y definir el costo y alcance de la arquitectura empresarial de la metodología de gestión de servicios de negocio poniendo foco en las necesidades del cliente.

Estimación de porcentaje de disponibilidad de servicio

Hay variadas formas de calcular la **disponibilidad de servicio** :

Ejemplo de cálculo SLA

D = A/B x 100

A es el número de horas en las cuales cada una de los servicios estuvieron disponibles. Este parámetro se verá disminuido con respecto a las horas que el sistema no funciona correctamente, es decir, cuando se ve afectado por cualquiera de los siguientes problemas:

- Caída de cualquiera de los servicios.
- Latencias superiores a **x** segundos.
- Degradación del servicio, alterando los tiempos de respuesta.
- Errores en la configuración.
- Fallas humanas provocadas.
- Cualquier otro factor que se considere pertinente.

B es el número de horas en que los servicios deberían estar disponibles, cuyo valor es setecientos veinte (720) horas mensuales;

D es el porcentaje de Disponibilidad.

El incumplimiento de la Disponibilidad acordada, puede llevar a penalizaciones previamente acordadas

Atención al Cliente Se debe prestar un servicio de soporte acordado mediante contrato. En caso de que el cliente reporte una indisponibilidad o fallas en el servicio, nosotros una vez reportada la falla y entregado el ticket se deberá de informar al cliente el seguimiento y solución que se le haya dado, en intervalos definidos. Todo esto

definidos en :

1. Acuerdos de niveles de servicio
    1. Tiempo de atención a fallas
    2. Niveles de Escalamiento – aseguramiento para los servicios
    3. Tiempo de atención al usuario.

Punto de restauración de servicio, se debe verificar con el Cliente por medio de **Pruebas de Aceptación del Servicio** a definir contractualmente que el Cliente contratante del servicio realizará a fin de acusar conformidad con la restauración de la disponibilidad del servicio.

Calculando el costo por caída de Servicio

Factores a tener en cuenta en un caso de ejemplo para una compañía muy dependiente de recursos de TI en el cual la indisponibilidad de servicio tiene mayor incidencia :

- **Coste de Inactividad de los empleados / Perdida de Productividad**
    - Básicamente es “Cuánto cuesta que nuestros empleados estén parados”, para ello, podemos utilizar el coste medio por hora de los empleados y lo multiplicaremos por el tiempo de inactividad y por el número de empleados afectados por la caída.

Coste Medio de Inactividad = Coste por hora de empleado * Nº de empleados afectados * Duración de la interrupción de servicio.

- **Pérdida de Operaciones**
    - Este concepto hace referencia al número de operaciones que no se llevan a cabo por la caída de sistemas, ya sean ventas, transacciones, pedidos o cualquier otra operación objeto del negocio de la compañía. Por ejemplo, si nuestra compañía se dedica a vender productos por internet podemos determinar en función de las estadísticas de venta diarias cuanto nos cuesta tener parado nuestro portal web.
- **Incumplimientos de normativa, acuerdos o SLA.**
    - Nuestra compañía puede tener contratos de prestación de servicios con otras empresas en las cuales se recogen penalizaciones por incumplimiento del mismo. Una caída de nuestro sistema puede afectar a estas empresas ocasionándolos un agravio y por ende reclamarnos una compensación económica. Por otro lado, algunas empresas por el sector al que se dedican o por el tipo de servicio que prestan, están sujetas a normativas o leyes que ante una parada del sistema pueden acarrearles sanciones o multas.
- **Impacto en marca, pérdida de confianza.**
    - Esto es algo muy relativo y tiene un impacto según la cultura de la zona donde se preste el servicio, pero puede ser un parámetro a considerar a la hora de

determinar el coste por caída de servicio. Muchas compañías emplean gran cantidad de recursos en campañas publicitarias, redes sociales, patrocinios etc., para afianzar y reforzar su marca en el mercado, una interrupción de servicio en los sistemas informáticos puede dañar su imágen o en algunos casos que el cliente desestime en ese momento utilizar sus productos o servicios.

Podemos determinar el costo en función del gasto económico que tiene que realizar la empresa para recuperar los niveles de confianza previos a la caída, pero para eso tenemos que establecer una ponderacíón de cada uno de los factores tenidos en cuenta para luego elaborar el cálculo de disponibilidad de servicio y el porcentaje de perjuicio a la metodología de provisión del servicio en la cuál se enmarca la empresa en cuestión.

Documentación de SLA

La documentación de SLA posee la vida y desarrollo de versiones del servicio. El documento posee revisión y firmas entre las partes que dan fé de la evolución y modificación de los acuerdos.

El documento debe contener las siguientes observaciones:

1. **Histórico**
    1. Ajuste de la revisión, fecha, descripción del histórico y autor de la revisión.
2. **Aprobaciones**
    1. Nombre, Fecha, título, firma y dirección de correo electrónico.
3. **Alcance**
    1. Audiencia
    2. Propósito
    3. Supuestos
    4. Contactos
4. **Garantías y recomendaciones y del detalle del servicio**
    1. Formatos de archivo
    2. Envío y Expectativas
        1. Tipo de archivo, frecuencia esperada
    3. Acciones de escalamiento
    4. Recursos para escalar
    5. Tiempo para la solución de problemas
    6. Histórico de desempeño
5. **Gestión de Problemas**
6. **Gestión del Desempeño**
7. **Funciones y Responsabilidades del Cliente**
8. **Terminación**

- Crear documentación de ayuda adicional
    - Información de contacto
        - Nombre función, teléfono, teléfono celular, etc

- Definiciones, términos, acrónimos;

Disposiciones legales

Además con los datos proporcionados por Nagios en la mayoría de los casos se puede obtener una explicación de cuál es el problema que está experimentando en su infraestructura, aunque sea aparentemente invisible, además esta operatoria ayuda en nuestra estrategia implementación de **BS 10012:2009 Data Protection - Specification for a Personal Information Management System (PIMS)**, la cual por ejemplo en Mexico es necesaria para cumplir los requerimientos de la **LFPDPPP** (LEY FEDERAL DE PROTECCIÓN DE DATOS PERSONALES EN POSESIÓN DE LOS PARTICULARES), ya que necesitamos capacidad de monitoreo para rastrear cualquier cambio en la información y poder llegar a establecer quién está utilizando cuales datos y cómo lo hace.

Estrategias

Monitoreo Activo

Este tipo de monitoreo se realiza enviando paquetes desde sl sistema de monitoreo a los clientes que necesitamos monitorear. Ya sea un PING o pedidos a determinadas aplicaciones en los mismos.

- Ventajas
  - No hay que instalar un agente especializado en el cliente. En algunos casos solo SNMP. Es una opción para casos en los que no es posible instalar aplicaciones en los clientes
- Desventajas
  - Tiene métricas menos especificas por consiguiente se pueden realizar análisis menos detallados. Pueden ser afectadas por hechos que sucedan en la red..

Monitoreo Pasivo

Esta estrategia se basa en la obtención de datos a partir desde los clientes a monitorear hacia el sistema de monitoreo. Este enfoque bien planificado puede ser mas perfomante a lo que trafico de red se refiere comparándolo con la técnica de Monitoreo Pasivo.

- Ventajas
  - Información más específica y más detallada. Mayor flexibilidad para realizar monitoreos personalizable. Posibilidad de crear soluciones de monitoreo que controlen estados de servicios o métricas no estándares sobre aplicaciones o hardware. El control de las aplicaciones y servicios se realiza directamente en el nodo monitoreado. Mayor seguridad en la red ya que se manejan protocolos encriptación. Menor riesgo de detección de inactividades.
- Desventajas
  - Puede provocar mayor carga de actividad en el cliente. Se debe instalar el agente en todos los equipos que se van a monitorear.

Capas a chequear

- **Aspectos generales**

    - Monitoreo de objetos o cajas negras con agentes o sin agentes
    - Reportes estadísticos
- **Infraestructura / Funcionalidad de Hardware**
    - Uptime
    - Respuesta
    - **Infraestructura / Sistema Operativo**
        - Recursos
        - Procesos
        - Núcleo
        - **Servicios y aplicaciones**
            - Procesos
            - Tiempos de respuesta
            - Usuarios
- **Notificaciones y alertas en caso de incidente**

Que estrategía utilizar

Por ejemplo, utilizamos una estrategia de Monitoreo Activo

1. Si el servicio que tenemos depende de un enlace de red
    1. Si ese servicio de conectividad de red esta caído y es el único acceso al servicio :
        1. Podemos decir que el servicio esta caído por ser inaccesible, igualmente a la hora de elaborar un reporte se detallará la razón como caída de enlace de red.
    2. Si ese servicio de conectividad no es el único modo de acceder al servicio o solo se cayó un nodo de la red WAN, ya sea una Unidad Organizativa, como una sucursal etc.
        1. Solo se mostrará como caído ese nodo de red, ejemplo Sucursal 1. Pero el servicio seguira figurando correctamente.

Information Technology Infrastructure Library - ITIL

ITIL es un conjunto de Mejores Prácticas en la Gestión de Servicios de TI. Es una guía, y no un manual de cómo han de hacerse las cosas.

Situándonos en el marco ITIL (Biblioteca de Infraestructura de Tecnologías de Información) para la provisión de servicios TI con calidad, los tres objetivos claves de la Gestión de Servicios son:

- Alinear los servicios informáticos con las necesidades actuales y futuras
- Mejorar la calidad de los servicios informáticos entregados
- Reducir el coste a largo plazo del suministro de servicios

Su objeto de aplicación parte de estas premisas

- No se puede gestionar lo que no se puede controlar.

- No se puede controlar lo que no se puede medir.
- No se puede medir lo que no se puede definir.

¿Como encuandra Nagios en ITIL?

En ITIL, los consultores acompañan a las empresas a diseñar y/o implementar sus procesos. También realizan GAPs para evaluar cuan cerca se encuentra la organización de las actividades que se recomiendan en las mejores prácticas y se recomiendan posibles mejoras para acercarse.

Nagios cubre estas áreas de ITIL :

- Service Desk
- Incident Management
- Service Level Management
- Capacity Management
- IT Service Continuity Management
- Availability Management
- ICT infrastructure Management

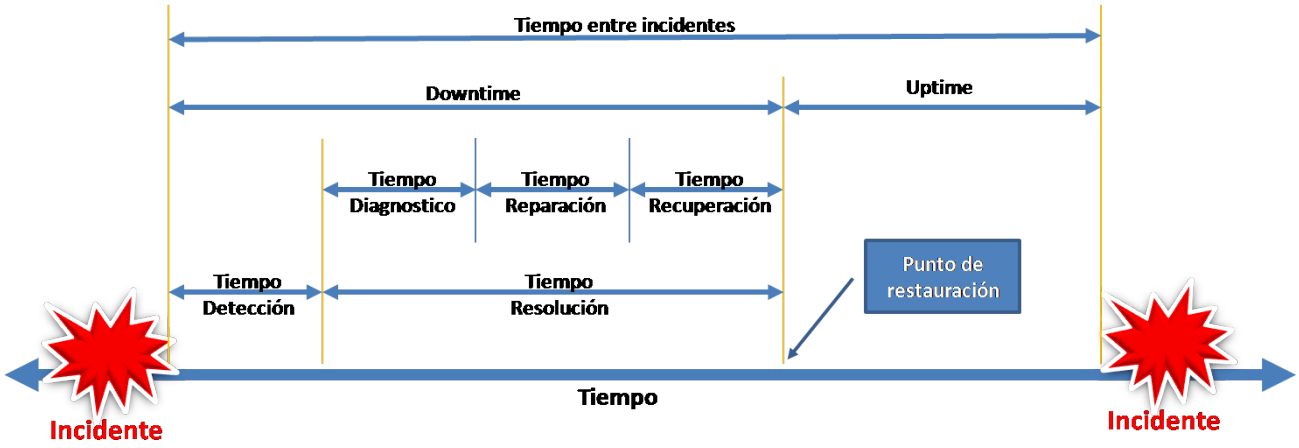
Gestión de la disponibilidad

- La disponibilidad “Availability Management” es un proceso del “Service Delivery”, definido en las especificaciones de ITIL.
  - Su meta es asegurar que el nivel de disponibilidad requerido esté proporcionado.
  - La supervisión y análisis de informes de la disponibilidad es una actividad clave para asegurar que los niveles del servicio se estén cumpliendo.
  - En la base de la gestión se debe supervisar contínuamente la disponibilidad de la Infraestructura, servicios y alertar a los administradores para iniciar los procedimientos oportunos.

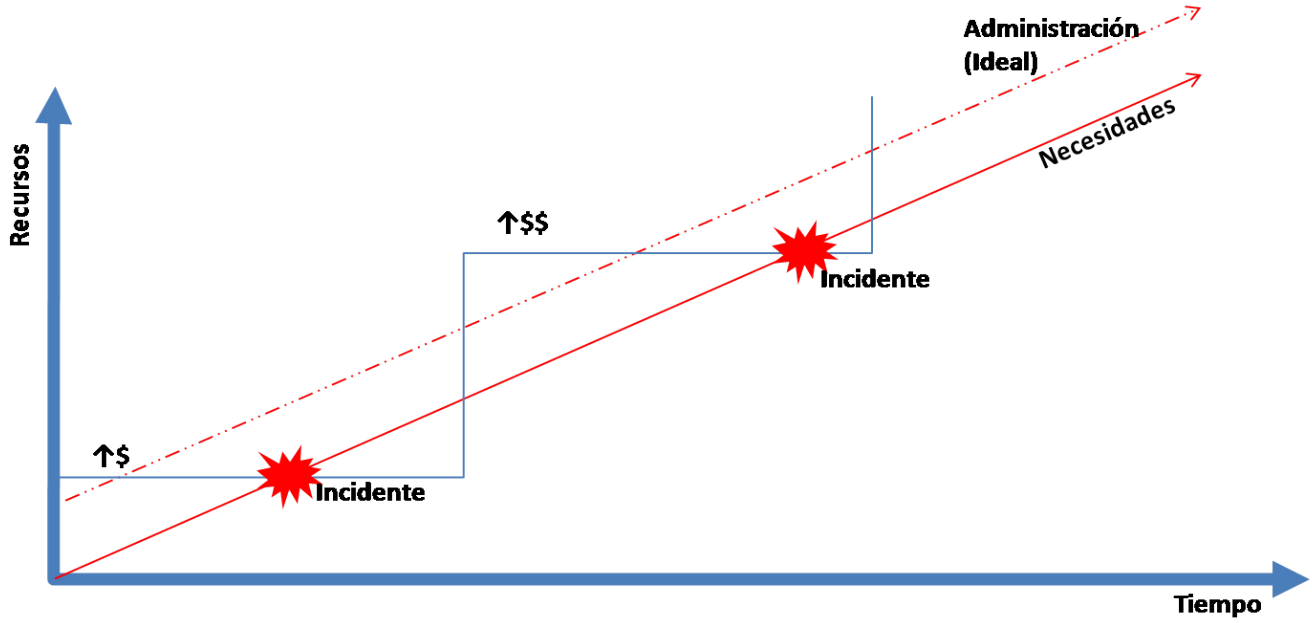
Tiempo resolución de Incidencias

- El tiempo de identificación de un problema mejora notablemente con la utilización de Nagios
  - Su meta es asegurar que el administrador 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 y performance ayudarán en la predicción de problemas y en identificar la necesidad de crecimientos.
  - Mejorando la Gestion de capacidad, estando directamente conectado con el processo de ITIL “Capacity Management”

Ciclo de un incidente



Administracion de la capacidad



Aplicando las normas ITIL

- Alinear los servicios de TI con las necesidades de la empresa (el negocio), actuales y futuras.
- Mejorar la calidad de los servicios de TI.
- Reducir los costos por la proveeduría de servicios de TI en el mediano y largo plazos.
- Mejora de rendimiento de la inversión de TI.
- Se mide el sistema de TI de la organización evaluando los procesos de Soporte Técnico y Entrega de Servicios comparándolos con las Mejores Prácticas.

Objetivos

Nuestras necesidades

Estan en orden de prioridad, pero todas van atadas de la mano una con otra

Solución de Monitoreo Integral

Registro y Correlación de Eventos, tanto en equipos de red como en servidores

Documentación Rápida, Participativa,Colaborativa Debe ser vía Web, con edición individual de secciones y versionado

Objetivos a largo plazo

CMDB Biblioteca para organizar el servicio de TI

- **Relaciones de muchos a muchos**
- **Inventario de equipos**
  - Detalle de responsable técnico
  - IP
  - Software Instalado
- **Base de IP**
  - Detalle de las subredes
  - Equipos asociados con sus ip y MAC si coresponde
- **Listado de reglas de ruteo indivuales por equipo, en caso de que existan**
- **Base de Certificados SSL**
  - Certificado con fecha de emisión y fecha de caducidad
  - Detalles tecnicos y equipos donde se utilizan los certificados
- **Base de Proovedores, Contratos y Contactos**
  - Contactos determinados por tipo, y detalles de como contactarlos
  - Detalles de un proovedor que presta servicios y le compramos equipos
  - Contratos detallados, con servicios y equipos asociados a los que aplica
- **Base de Usuarios y Contraseñas**
  - Nombre de Usuario, Contraseña y equipo u aplicación asociado
- **Necesario para enmarcar en ITIL**
  - Historico de modificaciónes
  - Inventario de los activos del Departamento de IT
  - Estructura del inventario y relaciones entre los diferentes elementos
  - Manejo de Licencias
  - Gestión de problemas, incidentes
    - Quejas y reclamos
  - Posibilidad de adjuntar documentación de cada item presente
  - SLA basados en contratos y horarios

Inventariado y gestion de Incidencias

¿Como terminamos de cubrir los puntos anteriores?

- Para la gestion de inventario y de incidencias presentadas con los equipos y servicios inventariados se puede implementar GLPI
  - Ademas se busca una herramienta que se integre y complemente al el sistema de ivnentario, como FusionInventory, que es una variante de OCS-Inventory, que se integra completamente con el Software GLPI

Descripción

Que se puede hacer con Nagios

- Monitorización de servicios de red (SMTP, POP3, HTTP, NTTP, ICMP, SNMP).
- Monitorización de los recursos de un host (carga del procesador, uso de los discos, logs del sistema) en varios sistemas operativos, incluso Microsoft Windows con el plugin NRPE_NT o también por medio del protocolo SNMP.
- Monitorización remoto, a través de túneles SSL cifrados o SSH.
- Diseño simple de plugins, que permiten a los usuarios desarrollar sus propios chequeos de servicios dependiendo de sus necesidades, usando sus herramientas preferidas (Bash, C++, Perl, Ruby, Python, PHP, C#, Java, etc.).
- Chequeo de servicios paralizados.
- Posibilidad de definir la jerarquía de la red, permitiendo distinguir entre host caídos y host inaccesibles.
- Notificaciones a los contactos cuando ocurren problemas en servicios o hosts, así como cuando son resueltos (Vía email, pager, Jabber, SMS o cualquier método definido por el usuario junto con su correspondiente complemento).
- Posibilidad de definir disparadores de eventos que se ejecuten al ocurrir una situación detectada dentro de un servicio o host para la resolución de problemas en forma proactiva.
- Rotación automática del archivo de registro.
- Soporte para implementar hosts de monitores redundantes.
- Interfaz web opcional, para observar el estado de la red actual, notificaciones, historial de problemas, archivos de registros, etc.
- Reportes y estadísticas del estado cronológico de disponibilidad de servicios y hosts.

Quien va a usar Nagios

- Administradores de Redes con alto conocimiento tecnico
- Operadores con minimo conocimiento tecnico de la situacion, o conocimiento puntual de algun servicio
  - Pudiedo avisar y ayudar a determinar posibles causas de efectos producidos
- Equipos de desarrollo
- Coordinadores de Mesas de Ayuda

- Areas relacionadas

Que se va a monitorear

Hay que definir o acotar que niveles y aspectos se van a monitorear

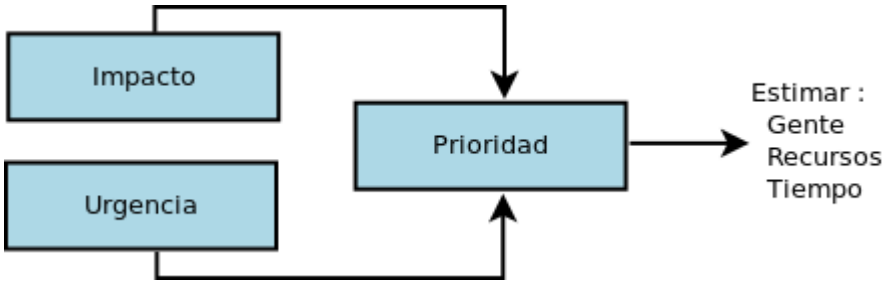
- Sistema Operativo
- Utilización de red
  - Trafico y consumo de ancho de banda
  - Tiempos de respuesta
- Servicios (SAP, Web, Bases de datos, DHCP)

Como impacta y para qué

- Mejora de productividad
- Antelacion de problemas
  - Conocer de donde proviene el problema
- Reporte y aviso de incidentes
  - Agilidad en su tratamiento
  - Compartir datos sobre la disponibilidad y SLA
- Mejor y mayor relacion e integracion de sectores adjuntos
- Detectar las brechas de seguridad
- Reducir y estimar periodos de inactividad y pérdidas empresariales
- Planificar actualizaciones de TI - **Capacity Planning**

Requerimientos Operativos

- Plan de acción bien diagramado
- Personal con conocimiento medio/avanzado en redes



impacto_urgencia.dia.gz

Recursos tecnológicos

- Hardware acorde a la inversion y criticidad de los datos
  - Si se van a monitorear >= 600 host y >= 2000 servicios hay que plantear bien esto, no sobrevaluar pero tampoco escatimar

Ejemplo un vez tuve que monitorear una red con 700 hosts y 2000 servicios que a su vez guardaba estadisticas completas en un MySQL dentro del mismo equipo para

luego generar informes SLA y mostrar datos en pantalla en una interfaz personalizada y disponia de un Intel Quad Core 2.8 GHz a 32 bits con un disco SATA de 320 GB y se quedo algo corto el hardware para los requerimientos

Pasos específicos de Nagios

Las tareas de la configuración son las siguientes:

- Definición de usuario para la ejecución de los *daemons*
- Configuración de apache para la visualización de la consola. Generalmente se accede a la consola de Nagios utilizando vínculos cifrados, por lo que es necesario crear los certificados correspondientes.
- Si el cliente posee un directorio donde centraliza todos sus usuarios, se podrá utilizar un usuario especialmente creado para consultar los perfiles de los usuarios que accederán a la consola de administración. Entre los directorios soportados se encuentra Microsoft Active Directory.
- Configuración de servidor de correo (MTA) y de SMS para el envío de todas las alertas que genera el sistema.
- Configuración de plugins básicos y de plugins específicos
- Definición de Servicios a monitorear
- Definición de dependencias de servicios (en aquellos servicios que dependan de la funcionalidad de otro servicio para poder ser monitoreados se deben configurar las dependencias, esto le indica a Nagios que no tiene que declarar fuera de línea un servicio si del que depende está caído).
- Definición de Commandos (los comandos son la forma en que se verifican el estado de los servicios declarados)
- Definición de HostGroups (los hostgroups son grupos de servidores que se monitorean, junto con el cliente de realiza la diagramación de estos grupos)
- Definición de Contacts (los contacts son personas o más bien direcciones de email/sms que integrarán los ContactGroups)
- Definición de ContactGroups (los contactgroups son grupos que se utilizarán para enviar las alertas que se generen en la monitoría)
- Creación de directivas (especificación de dónde se encuentran varias definiciones de Nagios como Hostgroups, ContactGroups, Contacts, etc.)
- Configuración de sistema de *parsing* de logs para producir las gráficas (históricas) de rendimiento de los servicios
- Adaptación de notificaciones (las notificaciones, el texto del correo electrónico, puede ser adaptado para una mejor identificación de problemas a los administradores)

Tareas

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 elijamos, si los tenemos empaquetados, o si los tenemos que compilar en instalar manualmente.

Paquete	Descripción	Sitio web
Perl	Interprete para el lenguaje de script Perl	http://www.perl.org
Net::SNMP	Modulo de Perl para consultas SNMP	http://search.cpan.org/dist/Net-SNMP
Crypt::DES	Modulo de Perl para encripción DES, necesario para consultas SNMPv3	http://search.cpan.org/~dparis/Crypt-DES/
Digest::HMAC	Keyed-Hashing for Message Authentication	http://search.cpan.org/dist/Digest-HMAC/
Digest::SHA1	Perl interface to the SHA-1 algorithm	http://search.cpan.org/dist/Digest-SHA1/
RRDTool	Utilitario para generación de gráficas de red y además su módulo de integración con el lenguaje Perl	http://oss.oetiker.ch/rrdtool
Zlib	Librería de compresión utilizada por las utilidades graficas	http://www.gzip.org/zlib/
LibJPEG	Librería para exportación jpg	http://www.ijg.org/
LibPNG	Librería para exportación png	http://www.libpng.org/pub/png/
Freetype2	Librería para procesamiento de fuentes	http://www.freetype.org/
Graphviz	Utilitario para generación de graficas	http://www.graphviz.org/
XFree86-libs	Librerías gráficas generales	http://koala.ilog.fr/lehors/xpm.html
Apache 2	Servidor Web	http://httpd.apache.org/
PHP	Interprete de lenguaje de script	http://www.php.net
MySQL	Sistema de base de datos	http://www.mysql.com
Postfix	SMTP para enviar mail	http://www.postfix.org/
GD	Librería para generación de formatos graficos	http://www.libgd.org/
Nagvis	Aditivo para la generación de diagramas dinamicos	http://www.nagvis.org/
PNP4Nagios	Aditivo para la generación	http://www.pnp4nagios.org/

	de gráficos estadísticos y reportes visuales	
NDO	Agregado para articular Nagios con MySQL	http://www.nagios.org
Plugins	Plugins de chequeo standard de Nagios	http://www.nagios.org
SNMP Plugins	Plugins para la integración de chequeos SNMP de Nagios	http://nagios.manubulon.com/
Nagios	Sitio de descarga oficial	http://www.nagios.org
NagiosQL	Herramienta visual de configuración de Nagios via Web	http://www.nagiosql.org/
Dokuwiki	Herramienta de documentación colaborativa	http://www.dokuwiki.org/
Syslog-Ng	Logueo de eventos del sistema	http://www.balabit.com/network-security/syslog-ng/
SNARE	Agente Syslog para clientes Windows	http://www.intersectalliance.com/projects/index.html
MK Livestatus	Aditivo para obtener los datos de Nagios en Vivo via Socket (muy útil para abandonar NDO)	http://mathias-kettner.de/checkmk_livestatus.html
Gnokii	Aplicación de interfaz para celulares y modems 3G, para la realización de llamadas y alertas SMS	http://gnokii.org
Thruk	Interfaz alternativa para Nagios, con muchas funciones extras, basado en MK Livestatus	www.thruk.org
Interfacetable_v3t	Plugin para detectar y chequear las interfaces de un Router	http://www.tontonitch.com/
Check TSM	Plugin para hacer chequeos generales en Tivoli TSM	https://github.com/osklil/nagios-misc
Check Iostat	Estadisticas y alertas en base a los datos de iostat	http://sysengineers.wordpress.com/2010/05/27/check_iostat-pl-version-0-9-7/
Cliente Oracle Basic SQL*Plus	Cliente de Oracle para realizar los chequeos	http://www.oracle.com/technetwork/database/features/instant-client/
Eventdb	Integración de chequeos de Syslog	https://www.netways.org/projects/eventdb
Highchart for Nagios	Gráficos de PNP4Nagios en AJAX	http://sourceforge.net/projects/highchartfornag/

Nota:

Hay algunos plugins que no estan mas disponibles en su sitio, aca los incluyo

nagios_lotus-blackberry.tgz

Descarga y compilación

En este apartado nos concentraremos en la descarga y compilación de los diferentes paquetes bajados en formato de *codigo fuente*.

Nagios

Para empezar deberemos descargar el *código fuente* del software Nagios desde su sitio web, en formato tar.gz, a la fecha la última versión es la 3.0.

Para ello descomprimiremos el paquete descargado y luego procederemos a *compilarlo*.

```
root@localhost # tar xvfzp nagios-3.0.6.tar.gz
root@localhost nagios-3.0.6 # ./configure

*** Configuration summary for nagios 3.0.6 12-01-2008 ***:

 General Options:
 -------------------------
        Nagios executable:  nagios
        Nagios user/group:  nagios,nagios
       Command user/group:  nagios,nagios
            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/rc.d/init.d
  Apache conf.d directory:  /etc/httpd/conf.d
             Mail program:  /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):  /bin/traceroute


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

root@localhost nagios-3.0.6 # make all

*** Compile finished ***

If the main program and CGIs compiled without any errors, you
can continue with installing Nagios as follows (type 'make'
without any arguments for a list of all possible options):

  make install
     - This installs the main program, CGIs, and HTML files

  make install-init
     - This installs the init script in /etc/rc.d/init.d

  make install-commandmode
     - This installs and configures permissions on the
       directory for holding the external command file
```

```
  make install-config
     - This installs *SAMPLE* config files in /usr/local/nagios/etc
       You'll have to modify these sample files before you can
       use Nagios.  Read the HTML documentation for more info
       on doing this.  Pay particular attention to the docs on
       object configuration files, as they determine what/how
       things get monitored!

  make install-webconf
     - This installs the Apache config file for the Nagios
       web interface


*** Support Notes *******************************************

If you have questions about configuring or running Nagios,
please make sure that you:

     - Look at the sample config files
     - Read the HTML documentation
     - Read the FAQs online at http://www.nagios.org/faqs

before you post a question to one of the mailing lists.
Also make sure to include pertinent information that could
help others help you.  This might include:

     - What version of Nagios you are using
     - What version of the plugins you are using
     - Relevant snippets from your config files
     - Relevant error messages from the Nagios log file

For more information on obtaining support for Nagios, visit:

       http://www.nagios.org/support/

************************************************************

Enjoy.

root@localhost nagios-3.0.6 # make fullinstall
root@localhost nagios-3.0.6 # make install-config
```

Para correr el *daemon del servicio nagios* y realizar tareas de administración y configuración sobre el, se debió crear el usuario **nagios** y el grupo **nagios** con privilegios de usuario normal del sistema, con su home ubicado en el directorio de instalación de nagios **/usr/local/nagios**, luego se le atribuyeron permisos de acceso y escritura para dicho usuario.

```
root@localhost # groupadd nagios
root@localhost # useradd nagios -d /usr/local/nagios -G nagios
root@localhost # chown nagios:nagios -R /usr/local/nagios
```

Para dejar corriendo el servicio deberemos ejecutar :

/etc/init.d/nagios start

Luego deberemos bajarnos el paquete de plugins básico de Nagios y descomprimirlo para luego compilarlo

```
root@localhost # tar xvfzp nagios-plugins-1.4.13.tar.gz
root@localhost nagios-plugins-1.4.13 # ./configure
root@localhost nagios-plugins-1.4.13 # make all
root@localhost nagios-plugins-1.4.13 # make install
```

```
root@localhost nagios-plugins-1.4.13 # make install-root
```

Con esto ya tendremos un Nagios instalado y funcional en modo básico.

PNP4Nagios

PNP4Nagios es un addon para Nagios que básicamente, nos genera gráficas con los resultados de los análisis de Nagios, para poder llevar un control más general de la monitorización de un determinado servidor o servicio en las últimas horas, días, semanas, meses o incluso años.

Una vez que tenemos instalado y configurado Nagios procederemos a descargar y descomprimir el paquete pnp4nagios,

```
root@localhost # tar xvfzp pnp-0.4.12.tar.gz
root@localhost pnp-0.4.12 # ./configure

*** Configuration summary for pnp 0.4.12 10-11-2008 ***

  General Options:
  -------------------------        -------------------
  Nagios user/group:               nagios nagios
  Install directory:               /usr/local/nagios
  HTML Dir:                        /usr/local/nagios/share/pnp
  Config Dir:                      /usr/local/nagios/etc/pnp
  Path to rrdtool:                 /usr/bin/rrdtool (Version 1.2.29)
  RRDs Perl Modules:               FOUND (Version 1.2029)
  RRD Files stored in:             /usr/local/nagios/share/perfdata
  process_perfdata.pl Logfile:     /usr/local/nagios/var/perfdata.log
  Perfdata files (NPCD) stored in: /usr/local/nagios/var/spool/perfdata/


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

root@localhost pnp-0.4.12 # make all
root@localhost pnp-0.4.12 # make fullinstall
```

NDOUtils

El generador de graficas **Nagvis** necesita acceder a los datos que **Nagios** genera, una de las formas de acceder a los mismos es que **Nagios** almacene sus datos dentro de una base de datos **MySQL** ya que por defecto lo hace en archivos de texto, para que **Nagios** pueda hacer eso, deberemos instalar el modulo **NDO** que viene dentro del paquete **NDOUtils** descargable via el sitio web de **Nagios**. Este módulo es el que se encarga de generar las consultas en formato **MySQL**, que son cargadas sobre un socket El proceso **NDO2DB** corriendo como daemon lee de ese socket y carga los datos en una base de datos **MySQL**.

```
root@localhost # tar xvfzp ndoutils-1.4b7.tar.gz
root@localhost ndoutils-1.4b7 # ./configure

*** Configuration summary for ndoutils 1.4b7 10-31-2007 ***:

 General Options:
 -------------------------
 NDO2DB user:    nagios
 NDO2DB group:   nagios
```

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

root@localhost ndoutils-1.4b7 # make
root@localhost ndoutils-1.4b7 # cd db
root@localhost db #./installdb -u nagios -p nagios -h localhost -d nagios
```

Hay 4 componentes principales que inician las utilidades NDO:

1. NDOMOD Event Broker Module (Modulo de evento corredor)
2. LOG2NDO Utility
3. FILE2SOCK Utility
4. NDO2DB Daemon

El NDOMOD Event Broker Module

Las utilidades NDO incluyen un Nagios Even Broker Module (NDOMOD.O) que exporta datos desde el demonio de nagios.

Asumiendo que nagios fue compilado con el Modulo Event Broker activado (esto es por default), usted puede configurar que nagios cargue el modulo NDOMOD en tiempo de ejecucion. Una vez que el modulo fue cargado por el daemon de nagios, este puede acceder a todos los datos y logicamente presente el el proceso de nagios que esta corriendo.

El modulo NDOMOD tiene designado exportar la configuracion, como informacion variada de eventos en tiempo de ejecucion que ocurre en el proceso de monitoreo, por el daemon de nagios. El modulo puede enviar esta informacion a un archivo estandar, a un Socket Unix de Dominio o un a socket TCP.

El NDOMOD escribe la info en un formato que el demonio NDO2DB puede entender.

Si el NDOMOD esta escrito para un archivo de salida, usted puede configurarlo para rotarlo periodicamente y/o procesarlo en otra maquina fisicamente (usando SSH, etc.) y envia este contenido al daemon NDO2DB usando la utilidad FILE2SOCK (que describiremos mas adelante).

La utilidad LOG2NDO

Esta es designada para permitir importar un historial de logs de nagios a una BD via el NDO2DB daemon (describiremos luego). La utilidad trabaja enviando archivos de logs históricos a un archivo estandar, un unix sock o un tcp sock en un formato que NDO2DB daemon entienda. El NDO2DB daemon puede luego usarlo para procesar la salida y almacenar en un archivo de log historico informandolo en una BD.

La utilidad FILE2SOCK

Esta utilidad es muy simple, solo lee de un archivo estandar (o STDIN) y escribe todo sobre un socket de dominio unix o un tcp socket. Estos datos son leidos y no son procesados por nada, antes de ser enviados al socket.

El demonio NDO2DB

La utilidad es diseñada para tomar los datos de salida de los componentes NDOMOD y LOG2NDO y almacenarlos en una BD MySQL o BD PostgreSQL.

Cuando este inicia, el daemon NDO2DB crea un socket y espera que los clientes se conecten. NDO2DB puede correr independientemente, bajo un demonio

multiproceso o bajo inetd (si esta usando un socket TCP).

Multiples clientes pueden conectarse al daemon NDO2DB y transmitir simultáneamente.

Instalación

Una vez compilado el modulo **NDO** procederemos a instalarlo manualmente

- **cp src/ndomod-3x.o /usr/local/nagios/bin/ndomod.o**
  - *Con esto copiaremos el modulo al directorio de ejecución de Nagios*
- **cp config/ndomod.cfg /usr/local/nagios/etc**
  - *De esta manera instalaremos la configuración inicial del modulo*

Igualmente lo haremos con el daemon **NDO2DB**

- **cp src/ndo2db-3x /usr/local/nagios/bin/ndo2db**
  - *Con esto copiaremos el daemon al directorio de ejecución de Nagios*
- **cp config/ndo2db.cfg /usr/local/nagios/etc**
  - *De esta manera instalaremos la configuración inicial del daemon*

Configuración

```
CREATE INDEX start_time_idx ON nagios_hostchecks (start_time);
CREATE INDEX start_time_idx ON nagios_servicechecks (start_time);
CREATE INDEX scheduled_time_idx ON nagios_timedeventqueue (scheduled_time);
CREATE INDEX start_time_idx ON nagios_hostchecks (start_time);
CREATE INDEX start_time_idx ON nagios_servicechecks (start_time);
CREATE INDEX scheduled_time_idx ON nagios_timedeventqueue (scheduled_time);
ALTER TABLE `nagios_configfilevariables` DROP INDEX `instance_id` ;
ADD INDEX `instance_id` ( `instance_id` , `configfile_id` );
```

MK Livestatus

La forma clásica de acceder a la informacion actual de sus hosts y servicios es mediante la lectura y análisis del archivo status.dat, que es creado por Nagios en una base regular. El intervalo de actualización se configura a través status_update_interval en nagios.cfg. Un valor típico es de 10 segundos. Si la instalación es cada vez más grande, usted podría tener que aumentar este valor con el fin de reducir al mínimo el uso de CPU y de E / S de disco. La interfaz web de Nagios utiliza status.dat para mostrar sus datos.

Analizar status.dat no es muy popular entre los desarrolladores de addons. Así que muchos utilizan otro enfoque: NDO. Este es un módulo de ORC que se carga directamente en el proceso de Nagios y envía todas las actualizaciones de estado a través de un socket UNIX a un proceso de ayuda. Eso crea sentencias SQL y actualizaciones de varias tablas en una base de datos MySQL o PostgreSQL. Este enfoque tiene varias ventajas sobre status.dat:

- Los datos se actualizan de inmediato, no sólo cada 10 o 20 segundos.
- Las solicitudes tienen acceso fácil a los datos a través de SQL. N analizador para status.dat es necesario.
- En las grandes instalaciones el acceso de los addons a los datos es más rápida que la lectura status.dat.

Lamentablemente, sin embargo, ha NDO también algunas deficiencias graves:

- Tiene una configuración compleja.
- Se necesita una (creciente) base de datos para ser administrado.
- Se alimenta de una parte significativa de sus Recursos de la CPU, solo con el fin de mantener la base estrictas disponibles.
- Limpieza periódica de la base de datos puede colgar Nagios.

El futuro

Desde la versión 1.1.0, Check_MK ofrece un enfoque totalmente nuevo para acceder a datos de estado y también histórico: Livestatus. Así como NDO, Livestatus hacer uso de la API de Nagios evento Broker y carga un módulo binario en su proceso de Nagios. Pero luego otros NDO, Livestatus no realiza escribir datos. En su lugar, se abre un socket en la que pueden consultar los datos a demanda.

La toma permite enviar una solicitud de los servicios u otros datos y obtener una respuesta inmediata. Los datos son directamente leídos de estructuras de datos internas de Nagios. Livestatus no crea su propia copia de esos datos. A partir de la versión 1.1.2 que también se pueden recuperar los datos históricos de los archivos de registro a través de Nagios Livestatus.

Esto es no sólo un enfoque increíblemente simple, si no también muy rápido. Algunas ventajas son:

- Otro entonces NDO, utilizando Livestatus no impone una carga mensurable de su CPU para nada. Sólo en el tratamiento de las consultas de una cantidad muy pequeña de la CPU es necesario. Pero eso ni siquiera se bloqueará Nagios.
- Livestatus produce cero / S de disco cuando quering datos de estado.
- Acceso a los datos es mucho más rápido que analizar status.dat o consultar una base de datos SQL.
- No se necesita configuración, base de datos no es necesaria. Ninguna administración es necesario.
- Livestatus escalas bastante bien a las grandes instalaciones, incluso más allá de 50,000 servicios.
- Livestatus le da acceso a los datos específicos de Nagios no se dispone de ningún otro método disponible acceder al estado - por ejemplo, la información del tiempo una multitud se encuentra actualmente en período de notificación.

En el mismo tiempo, ofrece a sus Livestatus propio lenguaje de consulta que es simple de entender, ofrece la mayoría de la flexibilidad de SQL e incluso más en algunos casos. Es un protocolo rápido, ligero y no necesita un cliente binario. Incluso, pueden obtener acceso a los datos sin ningún tipo de software especial de ayuda.

Proceso de compilación

```
root@linux# wget 'http://www.mathias-kettner.de/download/mk-livestatus-1.1.2.tar.gz'
root@linux# tar xzf mk-livestatus-1.1.2.tar.gz
root@linux# cd mk-livestatus-1.1.2
root@linux#
root@linux# ./configure
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking for g++... g++
```

```
checking for C++ compiler default output file name... a.out
checking whether the C++ compiler works... yes
checking whether we are cross compiling... no
checking for suffix of executables...
checking for suffix of object files... o
checking whether we are using the GNU C++ compiler... yes
checking whether g++ accepts -g... yes
...
configure: creating ./config.status
config.status: creating Makefile
config.status: creating src/Makefile
config.status: creating config.h
config.status: config.h is unchanged
config.status: executing depfiles commands
root@linux# make
g++ -DHAVE_CONFIG_H -I. -I..    -I../nagios -fPIC -g -O2 -MT livestatus_so-
AndingFil...
g++ -DHAVE_CONFIG_H -I. -I..    -I../nagios -fPIC -g -O2 -MT livestatus_so-
ClientQue...
g++ -DHAVE_CONFIG_H -I. -I..    -I../nagios -fPIC -g -O2 -MT livestatus_so-
Column.o ...
g++ -DHAVE_CONFIG_H -I. -I..    -I../nagios -fPIC -g -O2 -MT livestatus_so-
ColumnsCo...
g++ -DHAVE_CONFIG_H -I. -I..    -I../nagios -fPIC -g -O2 -MT livestatus_so-
ContactsC...
g++ -DHAVE_CONFIG_H -I. -I..    -I../nagios -fPIC -g -O2 -MT livestatus_so-
CustomVar...
g++ -DHAVE_CONFIG_H -I. -I..    -I../nagios -fPIC -g -O2 -MT livestatus_so-
CustomVar...
....
root@linux# make install
Making install in src
make[1]: Entering directory `/d/nagvis-dev/src/mk-livestatus-1.1.2/src'
make[2]: Entering directory `/d/nagvis-dev/src/mk-livestatus-1.1.2/src'
test -z "/usr/local/bin" || /bin/mkdir -p "/usr/local/bin"
  /usr/bin/install -c 'unixcat' '/usr/local/bin/unixcat'
test -z "/usr/local/lib/mk-livestatus" || /bin/mkdir -p "/usr/local/lib/mk-
livestatus"
 /usr/bin/install -c -m 644 'livestatus.so' '/usr/local/lib/mk-
livestatus/livestatus.so'
 ranlib '/usr/local/lib/mk-livestatus/livestatus.so'
/bin/sh /d/nagvis-dev/src/mk-livestatus-1.1.2/install-sh -d /usr/local/lib/mk-
livestatus
/usr/bin/install -c livestatus.o /usr/local/lib/mk-livestatus
rm -f /usr/local/lib/mk-livestatus/livestatus.so
make[2]: Leaving directory `/d/nagvis-dev/src/mk-livestatus-1.1.2/src'
make[1]: Leaving directory `/d/nagvis-dev/src/mk-livestatus-1.1.2/src'
make[1]: Entering directory `/d/nagvis-dev/src/mk-livestatus-1.1.2'
make[2]: Entering directory `/d/nagvis-dev/src/mk-livestatus-1.1.2'
make[2]: Nothing to be done for `install-exec-am'.
make[2]: Nothing to be done for `install-data-am'.
make[2]: Leaving directory `/d/nagvis-dev/src/mk-livestatus-1.1.2'
make[1]: Leaving directory `/d/nagvis-dev/src/mk-livestatus-1.1.2'
```

Despues tenemos que especificar que Nagios cargue el archivo objeto compilado *livestatus.o*, para eso debemos agregar a nagios.cfg:

```
broker_module=/usr/local/lib/mk-livestatus/livestatus.o /var/lib/nagios/rw/live
event_broker_options=-1
```

Y dentro del log de Nagios vamos a ver algo similar a esto

```
[1256144866] livestatus: Version 1.1.2 initializing. Socket path: '/var/lib
```

```
/nagios/rw/live'
[1256144866] livestatus: Created UNIX control socket at /var/lib/nagios/rw/
live
[1256144866] livestatus: Opened UNIX socket /var/lib/nagios/rw/live
[1256144866] livestatus: successfully finished initialization
[1256144866] Event broker module '/usr/local/lib/mk-livestatus/livestatus.o'
initializ
ed successfully.
[1256144866] Finished daemonizing... (New PID=5363)
[1256144866] livestatus: Starting 10 client threads
[1256144866] livestatus: Entering main loop, listening on UNIX socket
```

Opciones del módulo

Opción	Valor por default	Que significa
debug	0	Set this to 1 in order to make Livestatus log each query it executes in nagios.log
max_cached_messages	500000	Livestatus' access to Nagios logfiles caches messages in-memory. Here you can set the maximum number of cached messages. Each message takes about 250 bytes (in the current implementation)
max_response_size	104857600	Livestatus constructs each response in-memory before sending it to the clients. In order to avoid a crash in case of extensive queries, the maximum response size is limited. The default limit is 100 MB
num_client_threads	10	Livestatus needs one thread for each concurrent client connection. A fixed number of threads is created when Nagios starts
thread_stack_size	65536	This parameter sets the size of the stack of each client thread. In versions before 1.1.4, the stack size was set to 8 MB (pthread default). The new default value is 64 KB. A small stack reduces virtual memory usage and also save CPU ressources. A too small value will probably crash your Nagios process, though. You have been

		warned
query_timeout	10000	This value is in ms. In order to avoid being hung by broken clients, Livestatus imposes a limit on the time for reading the query from the client. A value of 0 disables the timeout
idle_timeout	300000	This value is in ms. Livestatus is waiting at most that much time for the next query. A value of 0 disables the timeout

Ejemplo de como agregar opciones

```
broker_module=/usr/local/lib/mk-livestatus/livestatus.o /var/run/nagios/rw/live
debug=1
```

Ejemplo de como dejar MK Livestatus escuchando en un socket tcp para consultarlo por red, por ejemplo por un sistema de reportes al estilo Jasper Reports o alguna interfaz alternativa como Thruk.

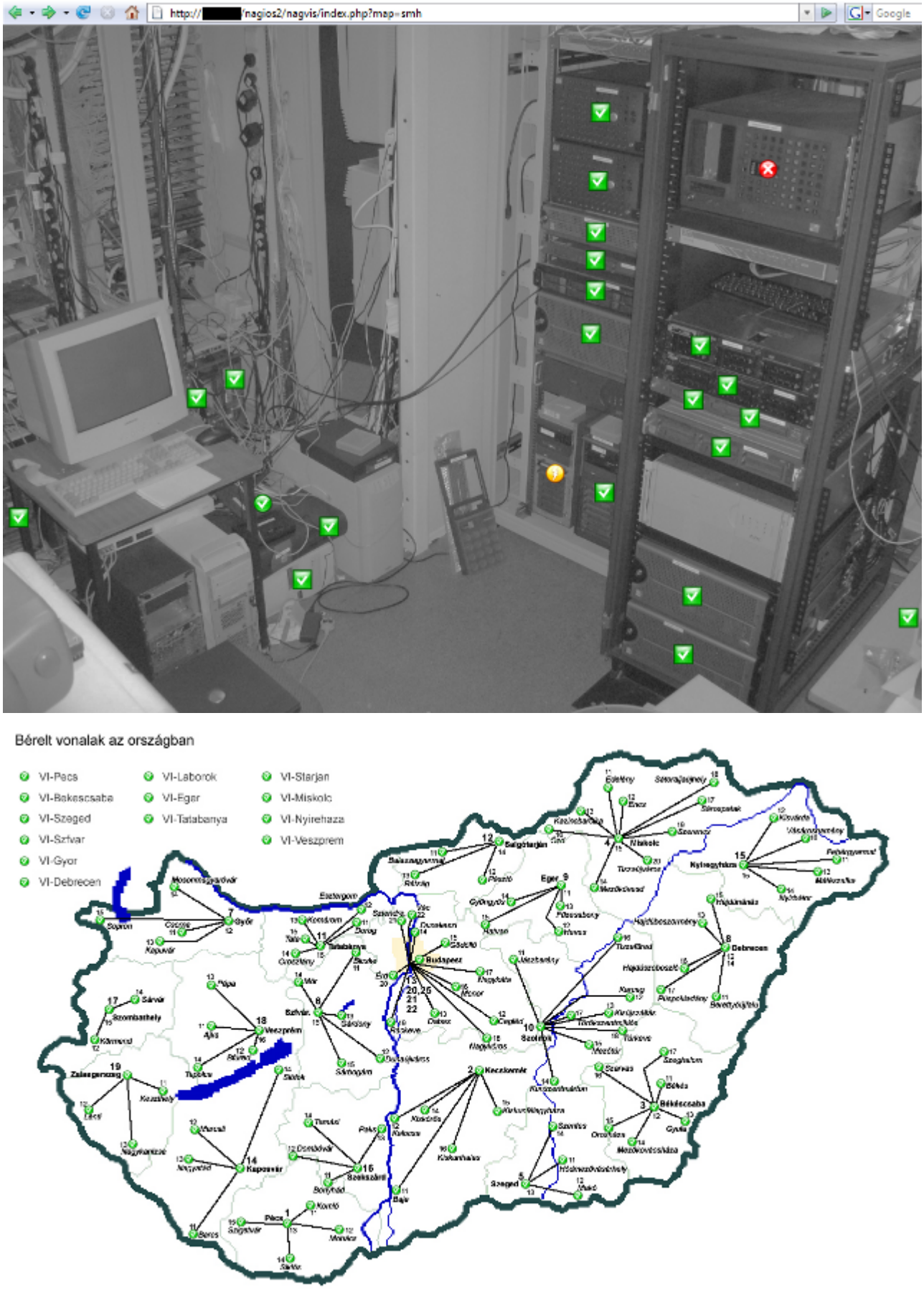
```
service livestatus
{
      type        = UNLISTED
      port        = 6557
      socket_type = stream
      protocol    = tcp
      wait        = no
# limit to 100 connections per second. Disable 3 secs if above.
      cps             = 100 3
# set the number of maximum allowed parallel instances of unixcat.
# Please make sure that this values is at least as high as
# the number of threads defined with num_client_threads in
# etc/mk-livestatus/nagios.cfg
        instances       = 500
# limit the maximum number of simultaneous connections from
# one source IP address
        per_source      = 250
# Disable TCP delay, makes connection more responsive
      flags           = NODELAY
      user        = nagios
      server            = /usr/local/nagios/bin/unixcat
      server_args     = /usr/local/nagios/var/rw/live
# configure the IP address(es) of your Nagios server here:
#     only_from       = 127.0.0.1 10.0.20.1 10.0.20.2
      disable           = no
}
```

Nagvis

Nagvis es un addon para Nagios, con el cual podemos tener gráficos a modo de diagrama estructural de red, dinámicos, con lo cual podemos conocer el estado actual de la red mirando un gráfico amigable al usuario final para que pueda tomar acciones y/o informar sobre el estado de los mismos a personal responsable.

Se pueden utilizar imágenes propias de fondo (denominadas mapas) y luego integrarle iconos representativos de las máquinas y servicios de la red que

muestren el estado actual de los mismos.



Deberemos bajar el paquete Nagvis desde su sitio web, y descomprimirlo y copiarlo a un directorio visible desde la interfaz web de Nagios.

```
root@localhost # tar xvzf nagvis-1.3.x.tar.gz
```

```
root@localhost # mv nagvis /usr/local/nagios/share
```

Para instalar la configuración por defecto deberemos dentro de /usr/local/nagios/share/nagvis renombrar o mover el archivo etc/nagvis.ini.php-sample a etc/nagvis.ini.php.

Para su correcto funcionamiento necesitaremos establecer una serie particular de permisos

```
root@localhost # chmod 664 /usr/local/nagios/share/nagvis/etc/nagvis.ini.php
root@localhost # chmod 775 /usr/local/nagios/share/nagvis/nagvis/images/maps
root@localhost # chmod 664 /usr/local/nagios/share/nagvis/nagvis/images/maps/*
root@localhost # chmod 775 /usr/local/nagios/share/nagvis/etc/maps
root@localhost # chmod 664 /usr/local/nagios/share/nagvis/etc/maps/*
root@localhost # chmod 775 /usr/local/nagios/share/nagvis/var
root@localhost # chmod 664 /usr/local/nagios/share/nagvis/var/*
```

Nota sobre la generación del Automap

Para una correcta generación de la característica Automap de Nagvis, primero es necesario configurar todos los *parents hosts* para poder diagramar en tiempo real la topología de red. Ademas deberemos tomar los iconos de la directiva *statusmap_image* que especificamos en el anexo *hostextinfo*, dicho icono se almacena en *nagios/share/images/logos*, en formato gd2 entendible por el binario statusmap.cgi para mostrarlo via web, pero no es entendible por nagvis siendo que este lo busca en base al parametro *statusmap_image* entonces aprovechando que dicho icono esta en formato gd2 y en formato png copiaremos en archivo en formato png al directorio *nagvis/images/shapes* pero cambiandole la extensión de png a gd2, y entonces Nagvis lo vera como un gd2 siendo que realmente es un PNG y debido a eso sera capas de mostrarlo.

Nota Nagvis 1.5 y template integrada con PNP

Modificar el archivo *nagvis/share/userfiles/templates/default.hover.html* de la siguiente manera, agregando solo las dos lineas donde especificamos el tag img.

```
<table class="hover_table">
      <tr><th colspan="2">[lang_obj_type] ([lang_last_status_refresh]:
[last_status_refresh])</th></tr>
      <tr><td class="label"><label>[lang_name]</label></td><td>[obj_name]</td></tr>
      <!-- BEGIN service -->
      <tr><td
class="label"><label>[lang_service_description]</label></td><td>[service_description
]</td></tr>
      <!-- END service -->
      <tr><td
class="label"><label>[lang_alias]</label></td><td>[obj_alias]</td></tr>
      <!-- BEGIN host -->
      <tr><td colspan="2" style="text-align:center;"><img
src="/nagios/pnp/index.php?host=[obj_name]&srv=_HOST_&source=1&view=0&display=image"
with="300px" height="94px"></td></tr>
      <tr><td class="spacer" colspan="2"></td></tr>
      <tr><td class="label[obj_state]"><label>[lang_state]
([lang_state_type])</label></td><td class="state[obj_state]">[obj_state]
([obj_state_type])[obj_in_downtime][obj_acknowledged]</td></tr>
      <tr><td
class="label"><label>[lang_output]</label></td><td>[obj_output]</td></tr>
      <tr><td
class="label"><label>[lang_perfdata]</label></td><td>[obj_perfdata]</td></tr>
      <tr><td
class="label"><label>[lang_current_attempt]</label></td><td>[obj_current_check_attem
pt]/[obj_max_check_attempts]</td></tr>
```

```
      <tr><td
class="label"><label>[lang_last_check]</label></td><td>[obj_last_check]</td></tr>
      <tr><td
class="label"><label>[lang_next_check]</label></td><td>[obj_next_check]</td></tr>
      <tr><td
class="label"><label>[lang_last_state_change]</label></td><td>[obj_last_state_change
]</td></tr>
      <!-- END host -->
      <tr><td class="spacer" colspan="2"></td></tr>
      <tr><td
class="label[obj_summary_state]"><label>[lang_summary_state]</label></td><td
class="state[obj_summary_state]">[obj_summary_state] [obj_summary_in_downtime]
[obj_summary_acknowledged]</td></tr>
      <tr><td
class="label"><label>[lang_summary_output]</label></td><td>[obj_summary_output]</td>
</tr>
      <!-- BEGIN service -->
      <tr><td colspan="2" style="text-align:center;"><img
src="/nagios/pnp/index.php?
host=[obj_name]&srv=[service_description]&source=1&view=0&display=image"
with="300px" height="94px"></td></tr>
      <tr><td
class="label"><label>[lang_perfdata]</label></td><td>[obj_perfdata]</td></tr>
      <tr><td
class="label"><label>[lang_current_attempt]</label></td><td>[obj_current_check_attem
pt]/[obj_max_check_attempts]</td></tr>
      <tr><td
class="label"><label>[lang_state_type]</label></td><td>[obj_state_type]</td></tr>
      <tr><td
class="label"><label>[lang_last_check]</label></td><td>[obj_last_check]</td></tr>
      <tr><td
class="label"><label>[lang_next_check]</label></td><td>[obj_next_check]</td></tr>
      <tr><td
class="label"><label>[lang_last_state_change]</label></td><td>[obj_last_state_change
]</td></tr>
      <!-- END service -->
      <!-- BEGIN childs -->
      <tr><td class="spacer" colspan="2"></td></tr>
      <tr><td colspan="2">
            <table width="100%">
                  <tr>
                  <!-- BEGIN servicegroup --><td
class="label"><label>[lang_child_name1]</label></td><!-- END servicegroup --><td
class="label"><label>[lang_child_name]</label></td><td
class="label"><label>[lang_state]</label></td><td
class="label"><label>[lang_output]</label></td>
                  </tr>
                  <!-- BEGIN loop_child -->
                  <tr><!-- BEGIN servicegroup_child --><td>[obj_name1]</td><!-- END
servicegroup_child --><td>[obj_name]</td><td
class="state[obj_summary_state]">[obj_summary_state][obj_summary_in_downtime]
[obj_summary_acknowledged]</td><td>[obj_summary_output]</td></tr>
                  <!-- END loop_child -->
            </table>
      </td></tr>
      <!-- END childs -->
</table>
```

MySQL

MySQL es uno de los Sistemas Gestores de Bases de Datos Relacional multihilo y multiusuario, más populares,

Compilacion

Debemos descargar el paquete de su web oficial y proceder a compilarlo.

```
root@localhost # groupadd mysql
root@localhost # useradd -g mysql mysql
root@localhost # gunzip < mysql-VERSION.tar.gz | tar -xvf -
root@localhost # cd mysql-VERSION
root@localhost # ./configure --prefix=/usr/local/mysql
root@localhost # make
root@localhost # make install
root@localhost # cp support-files/my-medium.cnf /etc/my.cnf
root@localhost # cd /usr/local/mysql
root@localhost # bin/mysql_install_db --user=mysql
root@localhost # chown -R root  .
root@localhost # chown -R mysql var
root@localhost # chgrp -R mysql .
root@localhost # bin/mysqld_safe --user=mysql &
```

Establecer la contraseña del usuario root

Por defecto, el usuario root no tiene asignada una contraseña y esto no es nada recomendable, así que vamos a establecer una. Utilizamos el comando:

mysqladmin -u root password '*contraseñadificil*'

Cambiar ‘loquesea’ por la contraseña que desemos establecer, pero es importante no olvidarse de teclear las comillas simples.

Comprobamos la nueva contraseña abriendo una sesión en mysql:

mysql -u root -p

Nos pedirá la contraseña, la tecleamos, y si todo es correcto entraremos en la interfaz del cliente de MySQL, podemos teclear algún comando de mysql para interactuar con el servidor, por ejemplo:

mysql> show databases;

Nos mostrará las bases de datos que existan en el servidor, normalmente y si acabamos de instalar, aparecerán las bases de datos mysql y test.

Deberemos crear una base de datos llamada Nagios

mysql>create database nagios;

```
Query OK, 1 row affected (0.00 sec)
mysql>show databases;

+----------+
| Database |
+----------+
| mysql    |
| nagios   |
| test     |
+----------+
3 rows in set (0.01 sec)
```

Ahora deberemos crear un usuario con privilegios de SELECT, INSERT, UPDATE, DELETE

```
CREATE USER nagios IDENTIFIED BY 'nagios123';
```

Y darle privilegios sobre la base de datos nagios

```
mysql> GRANT ALL ON nagios.* TO nagios@localhost IDENTIFIED BY "nagios";
Query OK, 0 rows affected (0.00 sec)

mysql> quit
```

Tecleamos exit o quit para salir del programa cliente.

Paquetes

La instalacion de **MySQL**, en el caso de tenerlo empaquetado en nuestra distribucion Linux, es bastante simple

CentOS y RedHat

Para instalar el paquete mysql haremos uso de la utilidad de sistema **up2date** o **yum**

up2date mysql-server

yum install mysql-server

Y para dejar el servicio corriendo en segundo plano

service mysqld start

Para configurar el arranque del servicio

Lo habitual será que cuando arranque o se pare nuestro servidor tambien se inicie o detenga el MySQL, para ello deberemos ejecutar:

chkconfig –level 35 mysqld on

Esto activa el demonio mysqld en los runlevel 3 y 5, y lo detiene en el resto. Si queremos comprobar el estado del servicio podemos utilizar lo siguiente:

chkconfig –list mysqld

Debian

Para instalar mysql en Debian, deberemos ejecutar :

apt-get install mysql-server mysql-common mysql-client

Para arrancar el servicio y dejarlo corriendo :

/etc/init.d/mysql start

Y para setear su arranque por defecto :

update-rc.d -f mysql defaults

Nota SuSE

Para setear el arranque de un servicio en SuSE Linux deberemos ejecutar el comando *insserv -d nombre_servicio* con el parametro *-d* estariamos indicando una opcion similar a *defaults* como en Debian.

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 con la información correspondiente para hacerlo.

En esta etapa se hace dicha instalación siguiendo las mejores prácticas de instalación de Apache poniendo énfasis en su seguridad.

Apache es un software libre servidor HTTP de código abierto para plataformas Unix

(BSD, GNU/Linux, etc.), Windows, Macintosh y otras, es el mas usado en internet.

PHP es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas, en nuestro caso nos servira realizar una interpretación extra de los datos de Nagios via web.

CentOS

Para instalar los paquetes desde CentOS ejecutaremos :

yum install httpd php php-common

Para dejar el servicio corriendo :

/etc/init.d/httpd start

Debian

Para instalar los paquetes de Debian, deberemos ejecutar :

apt-get install apache2 php5

Para dejar el servicio corriendo :

/etc/init.d/apache2 start

Configuración

Ahora nos referiremos a la configuración de los elementos instalados para su posterior *articulación* y funcionamiento en *conjunto*.

Implementación

Para el correcto funcionamiento de Nagios, y asegurar escalabilidad con orden, se debe seguir una estructura de configuración y tener previamente planteados temas como:

- Definición una estructura de archivos y directorios acorde a la situación, haciéndolo a su vez mas entendible para su posterior administración
- Configurar Apache para su permitir su acceso via web por HTTP o HTTPS
- En la mayoría de los equipos a monitorear mientras fuera posible instalar y dejar corriendo los servicios de SNMP
- Configurar servicio de envío de emails
- Definir grupos de contactos a los cuales se les enviarían los avisos de notificaciones, dependiendo de que hosts o servicio se trate.
- Definir grupos de hosts y servicios, al tenerlos agrupados y verlos mas facilmente

A continuación se detalla como llevar a cabo dichos pasos necesarios para su implementación.

En el servidor

Configuraciones necesarias en el servidor de monitoreo.

Nagios

Algunos puntos basicos previos a la instalacion :

- **PATH** Esta es la ruta de instalación. Por defecto es /usr/local/nagios
- **Usuario** Usuario que va a usar nagios par ejecutarse. Debe crearse con adduser especificarle el PATH de Nagios como *su directorio home de inicio*, usualmente deberemos llamarlo nagios y debe estar dentro del grupo nagios
- **Grupo** Grupo de usuario que va a usar Nagios. Este grupo tendrá permisos sobre todos los ficheros y directorios de Nagios. Por defecto es nagios. Puede crearse con groupadd.
- **URL** Nagios utiliza una interfaz web para ejecutarse. Esta URL determina cual va a ser el directorio virtual que debe usar para instalarse. Por defecto /nagios, es decir, las peticiones irán dirigidas a http://host/nagios

Estructura de archivos

Una vez que compilamos e instalamos el paquete Nagios nos termina quedando una nomenclatura de directorios como la siguiente :

- **bin**
  - *Aqui se almacenan los binarios ejecutables*
- **etc**
  - *Guarda la configuracion de Nagios*
- **libexec**
  - *Se almacenan los plugins que efectuaran los chequeos a monitorear*
- **sbin**
  - *Dentro de este directorio se mantienen los ejecutables CGI de la interfaz web*
- **share**
  - *Organiza el contenido web a mostrar, iconos, html, php etc*
- **var**
  - *Guarda los datos de ejecucion del monitoreo, estado de servicios, hosts, y logs*

bin

Dentro de este directorio encontramos los ejecutable principales, como el binario ***nagios*** que es el que se ejecuta como proceso en segundo plano, el objeto ***ndomod.o*** que es el modulo que se encarga de traducir las estadisticas de nagios en formato de consultas *MySQL*, y ***ndo2db*** que el proceso en segundo plano que se encarga conectarse con la base de datos para posteriormente ejecutar esas consultas.

etc

Este directorio guarda la configuración de Nagios, sus componentes, hosts/servicios a chequear, comandos de ejecucion, contactos de notificación, intervalos de chequeos. Dentro de el hay diferentes subdirectorios y archivos.

libexec

Alli se contienen lo ejecutables de los plugins que efectuan los chequeos, SNMP, SAP, Oracle, SSH, que pueden ser binarios, scripts en Perl, PHP, Shell, Java, etc.

sbin

Aqui se almacenan los ejecutables cgi que se ejecutaran para la visualizacion por

web de la consola Nagios.

share

Aqui encontramos el contenido web, imagenes, logos, los aditivos como PNP, Nagvis y los datos que necesitan para funcionar estos.

var

Aqui se guardan los datos internos de Nagios, estadisticas de los chequeos, informacion de ejecucion actual, archivos de sockets, registros de logs, colas de ejecución de chequeos.

Archivos de configuracion nagios/etc

cgi.cfg

- *Definir archivo de configuracion principal de Nagios*
  - main_config_file=/usr/local/nagios/etc/nagios.cfg
- *Ruta donde se ubican los archivos a mostrar via web*
  - physical_html_path=/usr/local/nagios/share
- *Ruta del url a donde ubicar Nagios desde el navegador*
  - url_html_path=/nagios
- *Mostrar o no el icono de ayuda en la interfaz web*
  - show_context_help=0
- *Mostrar objetos pendientes de chequeo*
  - use_pending_states=1
- *Usar autenticacion para acceder a Nagios*
  - use_authentication=1
- *Tener usuario logueado por default (no recomendado, dejar comentado)*
  - #default_user_name=guest
- *Usuarios con acceso permitido para ver la informacion de objetos (separados por comas)*
  - authorized_for_system_information=nagiosadmin
- *Usuarios con acceso permitido para ver la informacion de configuracion (separados por comas)*
  - authorized_for_configuration_information=nagiosadmin
- *Usuarios con acceso permitido ejecucion de comandos nagios (separados por comas)*
  - authorized_for_system_commands=nagiosadmin
- *Usuarios permitidos a ver informacion 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 refresco para la interfaz web en segundos*
  - refresh_rate=90

htpasswd.users

- Archivo con passwords encriptadas de los usuarios que se autentificaran por HTTP

nagios.cfg

- Archivo de configuracion principal de Nagios, aqui se especifican los directorios de trabajo y se incluyen los archivos de configuracion extra a utilizar por Nagios
- Con diversos parametros :
  - **log_file** se especifica el archivo de log a utilizar por Nagios
  - **cfg_file** se especifica un archivo de configuracion extra a incluir en la ejecucion de Nagios
  - **cfg_dir** se especifica un directorio con archivos de configuracion extra a incluir recursivamente en la ejecucion de Nagios
  - **log_archive_path** path donde se alojaran los archivos de log
  - **use_syslog** integracion con syslog

ndo2db.cfg

- Archivo de configuracion del daemon que se encarga de introducir las consultar generadas por el modulo ndomod

ndomod.cfg

- Modulo de Nagios que se encarga de traducir la informacion de ejecucion de Nagios en consultas MySQL, disponiendolas por medio de un socket

resource.cfg

- Archivo de configuracion donde se definen macros de ejecucion

objects/

- Directorio de archivos generales de configuracion

objects/commands.cfg

- Definicion de comandos de ejecucion por default, con los alias que queremos usar

objects/contacts.cfg

- Definicion de contactos de notificacion

objects/localhost.cfg

- Plantilla inicial para el chequeo del host local

objects/printer.cfg

- Plantilla de ejemplo de chequeo de impresoras por SNMP

objects/switch.cfg

- Plantilla de ejemplo de chequeo de switch por SNMP

objects/templates.cfg

- Plantillas generales de host, contactos, y servicios

objects/timeperiods.cfg

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

objects/windows.cfg

- Plantilla de ejemplo de chequeo de equipos Windows

services/

- Aqui vamos a definir los servicios que usaremos en los chequeos. Se define la métrica o el servicio a monitorizar y el host/grupo de hosts sobre el que se ejecuta

var/rw/

- Alli se encuentra un archivo special de socket que realiza la comunicacion de los comando y ordenes de la interfaz web hacia nagios, como cambiar horarios de chequeo, deshabilitar notificaciones etc.
  - El archivo que alli 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 asi poder enviar ordenes desde la interfaz web CGI. Esta es una característica avanzada de Nagios es que permite vía web la ejecución de ciertas tareas más allá del propio conjunto de CGI's que vienen de serie, como por ejemplo la

caída o el reinicio del propio Nagios, etcétera. Para poder ejecutar este tipo de comandos es necesario también configurar el sistema de una forma un tanto especial. No hay que olvidar que al configurar Nagios de este modo se está permitiendo desde la web 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 fichero 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 el chequeo para buscar comandos externos, con tanta frecuencia como sea posible por el sistema y buscará los comandos en el archivo *nagios.cmd*.

En el siguiente gráfico detalla la organización recomendada de la configuración de Nagios.

diagrama_nagios.dia.gz

Si por ejemplo tenemos estos dos casos :

Nagios un país multiples provincias o localidades

- **Buenos Aires**
  - Lanús
  - Florencio Varela

    - CABA
    - Olavarria
    - Bahia Blanca
- **Santa Fe**
    - Rosario
- **Neuquén**
    - Zapala
    - San Martin

En este caso, simplemente deberemos tener templates para hosts y servicios, que se dividan por tipo. Si es un servidor, equipo de red, scada etc.

Nagios múltiples países y multiples provincias o localidades

- **Argentina**
    - Buenos Aires
    - Neuquen
    - Tucuman
- **Brasil**
    - Rio
    - Cajati
- **Paraguay**
    - Ciudad del este
    - Villa hayes
- **Portugal**
    - Lisboa

En este caso algo mas complejo, deberemos tener templates para hosts divididos para cada país, templates para servicios divididos para cada país, templates para contactos divididos para cada país.

Siempre debemos utilizar templates para todo, porque cuando debemos hacer un cambio en masa es mucho mas simple y con menos posibilidad a errores.

SNMP Traps

Una trap es generado por el agente snmp en el dispositivo a monitorear para reportar ciertas condiciones y cambios de estado en un procesp

- Se “cae” un servicio
- Hay un problema de memoria o de hardware
- La carga de procesos excede un límite
- Se llena una partición de disco

En debian para instalar el manejador de traps SNMP deberemos ejecutar los siguiente :

```
apt-get install snmptt
```

/etc/snmp/snmptt.ini

```
mode = daemon
log_system_enable = 1
unknown_trap_log_enable = 1

mysql_dbi_enable = 1
mysql_dbi_host = localhost
mysql_dbi_port = 3306
mysql_dbi_database = snmptt
mysql_dbi_table = snmptt
mysql_dbi_table_unknown = snmptt_unknown
mysql_dbi_table_statistics = snmptt_statistics
mysql_dbi_username = snmptt
mysql_dbi_password = mytrap
```

/etc/snmp/snmptt.conf

```
EVENT linkDown .1.3.6.1.6.3.1.1.5.3 "Status Events" Critical
```

/etc/snmp/snmptrapd.conf

```
disableAuthorization yes
traphandle default /usr/sbin/snmptthandler
```

Configuración de permisos :

```
usermod -a -G nagios snmptt
```

/etc/default/snmpd

```
TRAPDRUN=yes
```

Schema MySQL

```
CREATE DATABASE snmptt;
USE snmptt;

DROP TABLE snmptt;
CREATE TABLE snmptt (
id INT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
eventname VARCHAR(50),
eventid VARCHAR(50),
trapoid VARCHAR(100),
enterprise VARCHAR(100),
community VARCHAR(20),
hostname VARCHAR(100),
agentip  VARCHAR(16),
category VARCHAR(20),
severity VARCHAR(20),
uptime  VARCHAR(20),
traptime VARCHAR(30),
formatline VARCHAR(255));

USE snmptt;

DROP TABLE snmptt_unknown;
CREATE TABLE snmptt_unknown (
trapoid VARCHAR(100),
enterprise VARCHAR(100),
community VARCHAR(20),
hostname VARCHAR(100),
agentip  VARCHAR(16),
```

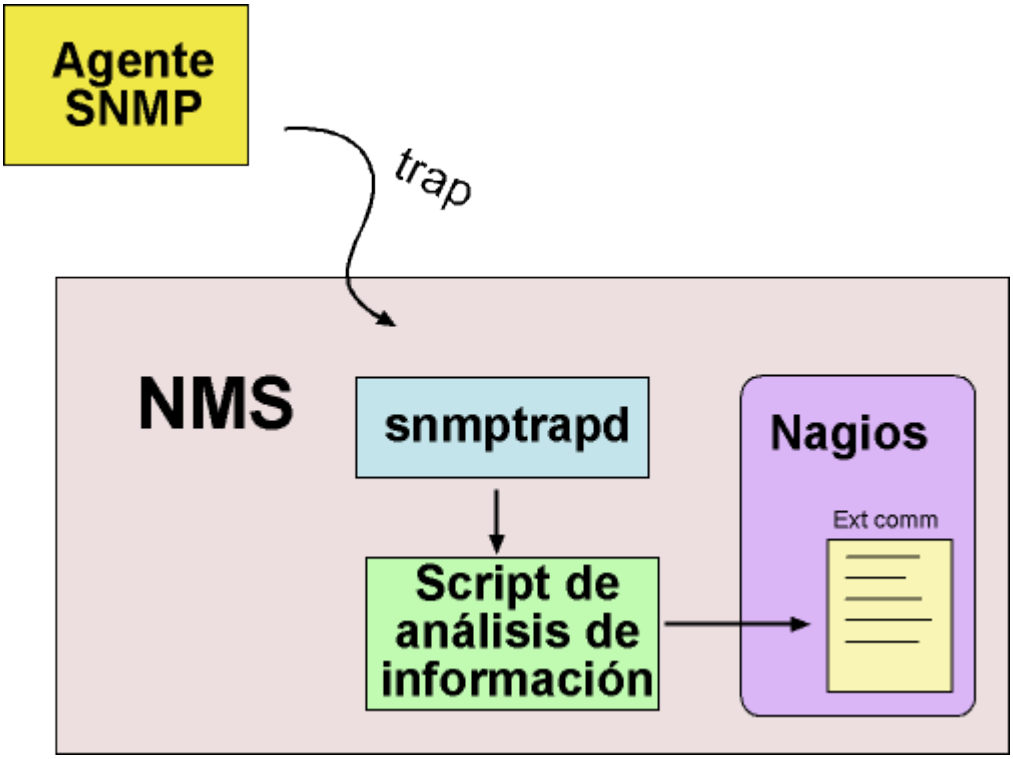
```
uptime  VARCHAR(20),
traptime VARCHAR(30),
formatline VARCHAR(255));

USE snmptt;

DROP TABLE snmptt_statistics;
CREATE TABLE snmptt_statistics (
stat_time VARCHAR(30),
total_received BIGINT,
total_translated BIGINT,
total_ignored BIGINT,
total_unknown BIGINT);
```

Reiniciamos los servicios SNMP :

```
/etc/init.d/snmptt
/etc/init.d/snmpd
```



Ref.: http://exchange.nagios.org/directory/Addons/SNMP/Nagios-SNMP-Trap-Interface-%28NSTI%29/details

Apache

Para permitir y tener una correcta visualizacion de la consola web de Nagios, procederemos a realizar una configuración personalizada dentro del servidor web Apache.

Deberemos crear un archivo de configuracion preferentemente con el nombre nagios.conf para tenerlo de una manera mejor organizada, y deberá ir dentro del directorio de donde el Apache obtiene su configuración, el mismo dependerá del método de instalación elegido o con que distribución de Linux estemos trabajando.

- **CentOS** /etc/httpd/conf.d
- **Debian** /etc/apache2/site-available *y luego crear un link simbolico a ese archivo dentro de* /etc/apache2/site-enabled

- **SuSE** /etc/apache2/vhosts.d

Archivo de configuración por defecto de Nagios

```
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>
```

Configuración SSL

La configuración HTTPS con SSL se realiza por defecto en la instalación del paquete Debian y CentOS, no asi en SuSE para ellos deberemos realizar configuraciones extra.

Primero procederemos a crear las claves SSL

```
localhost:~ # openssl genrsa -des3 -out server.3des-key 1024
localhost:~ # openssl rsa -in server.3des-key -out server.key
localhost:~ # openssl req -new -key server.key -x509 -out server.crt -days 365
localhost:~ # chmod 600 server.key
localhost:~ # rm server.3des-key
localhost:~ # mv server.crt /etc/apache2/ssl.crt
localhost:~ # mv server.key /etc/apache2/ssl.key
```

*Dentro del archivo **/etc/sysconfig/apache2** deberemos agregarle el siguiente FLAG de arranque para especificar la utilización del protocolo SSL dentro del servidor web Apache.*

```
APACHE_SERVER_FLAGS="-D SSL"
```

A continuación mostramos un archivo de configuración de Apache mas elaborado con autenticación por ActiveDirectory

```
<VirtualHost *:443>
SSLEngine on
SSLCertificateFile /etc/apache2/ssl.crt/server.crt
```

```
SSLCertificateKeyFile /etc/apache2/ssl.key/server.key
ScriptAlias /nagios/cgi-bin "/usr/local/nagios/sbin"
<Directory "/usr/local/nagios/sbin">
    AuthType Basic
    AuthName "Nagios Access"
    AuthLDAPEnabled On
    AuthLDAPURL ldap://192.168.1.9:389/OU=Usuarios,DC=test,DC=com,DC=ar?
SamAccountName?sub?(&(objectClass=user)
(memberOf=CN=Monitoreo,OU=Tech,OU=Grupos,DC=test,DC=com,DC=ar))
    AuthLDAPBindDN usuario_bind@test.com.ar
    AuthLDAPBindPassword *****
    Options All
    Order allow,deny
    Allow from all
    SSLRequireSSL
    AllowOverride None
    Require valid-user
</Directory>

Alias /nagios "/usr/local/nagios/share"
<Directory "/usr/local/nagios/share">
    AuthType Basic
    AuthName "Nagios Access"
    AuthLDAPEnabled On
    AuthLDAPURL ldap://192.168.1.9:389/OU=Usuarios,DC=test,DC=com,DC=ar?
SamAccountName?sub?(&(objectClass=user)
(memberOf=CN=Monitoreo,OU=Tech,OU=Grupos,DC=test,DC=com,DC=ar))
    AuthLDAPBindDN usuario_bind@test.com.ar
    AuthLDAPBindPassword *****
    Options All
    Order allow,deny
    Allow from all
    SSLRequireSSL
    AllowOverride None
    Require valid-user
</Directory>
```

El usuario Bind debe ser un usuario solo con privilegios necesarios para saber que usuarios existen (ver documentación LDAP), y su contraseña debe estar almacenada en texto plano en la configuración, y el archivo de configuración debe tener los debidos permisos para que no sean leídos por otros usuarios.

En la directiva **AuthLDAPURL**, especificamos

- IP del servidor de autenticacion
- DN donde buscar
  - OU=Usuarios,DC=test,DC=com,DC=ar
- Que buscar, en este caso usuarios
  - objectClass=user
- Que sean miembros del grupo Monitoreo a su vez dentro del grupo Tech
  - memberOf=CN=Monitoreo,OU=Tech,OU=Grupos,DC=test,DC=com,DC=ar

En caso de tener requerimientos mas exhaustivos con respecto a la autenticación, como restringir el acceso por ip se puede agregar a cada campo <Directory> los siguientes parámetros.

```
  Order deny,allow
  Deny from all
  Allow from 127.0.0.1
```

Con esto restringimos todo el acceso, y lo vamos abriendo IP x IP.

En el caso por ejemplo de que usemos otro sistema de autenticacion como CAS, deberemos compilar el modulo de apache Auth_CAS, si estammos usando Debian podemos bajarnos los archivos *libapache2-mod-auth-cas_1.0.8.orig.tar.gz libapache2-mod-auth-cas_1.0.8-3.diff.gz*, ya que a la fecha no estan en la rama estable, si no en la inestable, pero tenemos los fuentes originales y el diff para parchear esos fuentes y ejecutar los scripts que lo convierten en paquete debian. Antes deberemos tener instalados los paquetes *debhelper* y *dh-make* en el sistema.

```
# tar xvfzp libapache2-mod-auth-cas_1.0.8.orig.tar.gz
# gzip -dc libapache2-mod-auth-cas_1.0.8-3.diff.gz | patch -p0
patching file libapache2-mod-auth-cas-1.0.8/debian/libapache2-mod-auth-cas.install
patching file libapache2-mod-auth-cas-1.0.8/debian/watch
patching file libapache2-mod-auth-cas-1.0.8/debian/README.Debian
patching file libapache2-mod-auth-cas-1.0.8/debian/copyright
patching file libapache2-mod-auth-cas-1.0.8/debian/compat
patching file libapache2-mod-auth-cas-1.0.8/debian/control
patching file libapache2-mod-auth-cas-1.0.8/debian/rules
patching file libapache2-mod-auth-cas-1.0.8/debian/changelog
patching file libapache2-mod-auth-cas-1.0.8/debian/auth_cas.conf
patching file libapache2-mod-auth-cas-1.0.8/debian/libapache2-mod-auth-cas.postinst
patching file libapache2-mod-auth-cas-1.0.8/debian/libapache2-mod-auth-cas.docs
patching file libapache2-mod-auth-cas-1.0.8/debian/auth_cas.load
patching file libapache2-mod-auth-cas-1.0.8/debian/libapache2-mod-auth-cas.lintian-
overrides
patching file libapache2-mod-auth-cas-1.0.8/debian/libapache2-mod-auth-cas.dirs
patching file libapache2-mod-auth-cas-1.0.8/debian/README.source
patching file libapache2-mod-auth-cas-1.0.8/debian/patches/00list
patching file libapache2-mod-auth-cas-1.0.8/debian/patches/10_ssl_libs.dpatch
libapache2-mod-auth-cas-1.0.8# debian/rules binary
...
dpkg-deb: building package `libapache2-mod-auth-cas' in `../libapache2-mod-auth-
cas_1.0.8-3_i386.deb'.
```

Con eso tenemos el paquete libapache2-mod-auth-cas listo para funcionar, para instalarlo deberemos ejecutar el comando *dpkg -i libapache2-mod-auth-cas_1.0.8-3_i386.deb*, luego ejecutar el comando *a2enmod auth_cas*, con lo cual habilitamos ese modulo en la ejecucion del servicio Apache.

En el archivo de configuración del site de Nagios, deberemos incluir las siguientes lineas.

```
CASAllowWildcardCert On
#CASCookiePath /tmp/mod_auth_cas/
CASDebug On
CASValidateServer Off
CASValidateDepth 9
CASLoginURL https://servidorcas/cas/login
CASValidateURL https://servidorcas/cas/proxyValidate
CASTimeout 7200
CASIdleTimeout 7200
```

En la directiva CASCookiePath, especificamos un directorio temporal donde alojar las cookies, por default en debian es */var/cache/apache2/mod_auth_cas*

Luego en cada entrada Directory eliminamos todas las entradas de autenticacion y las cambiamos a solo esta *AuthType CAS*.

Deberemos reiniciar el servicio Apache para que los cambios tengan efecto.

Configuración de Email

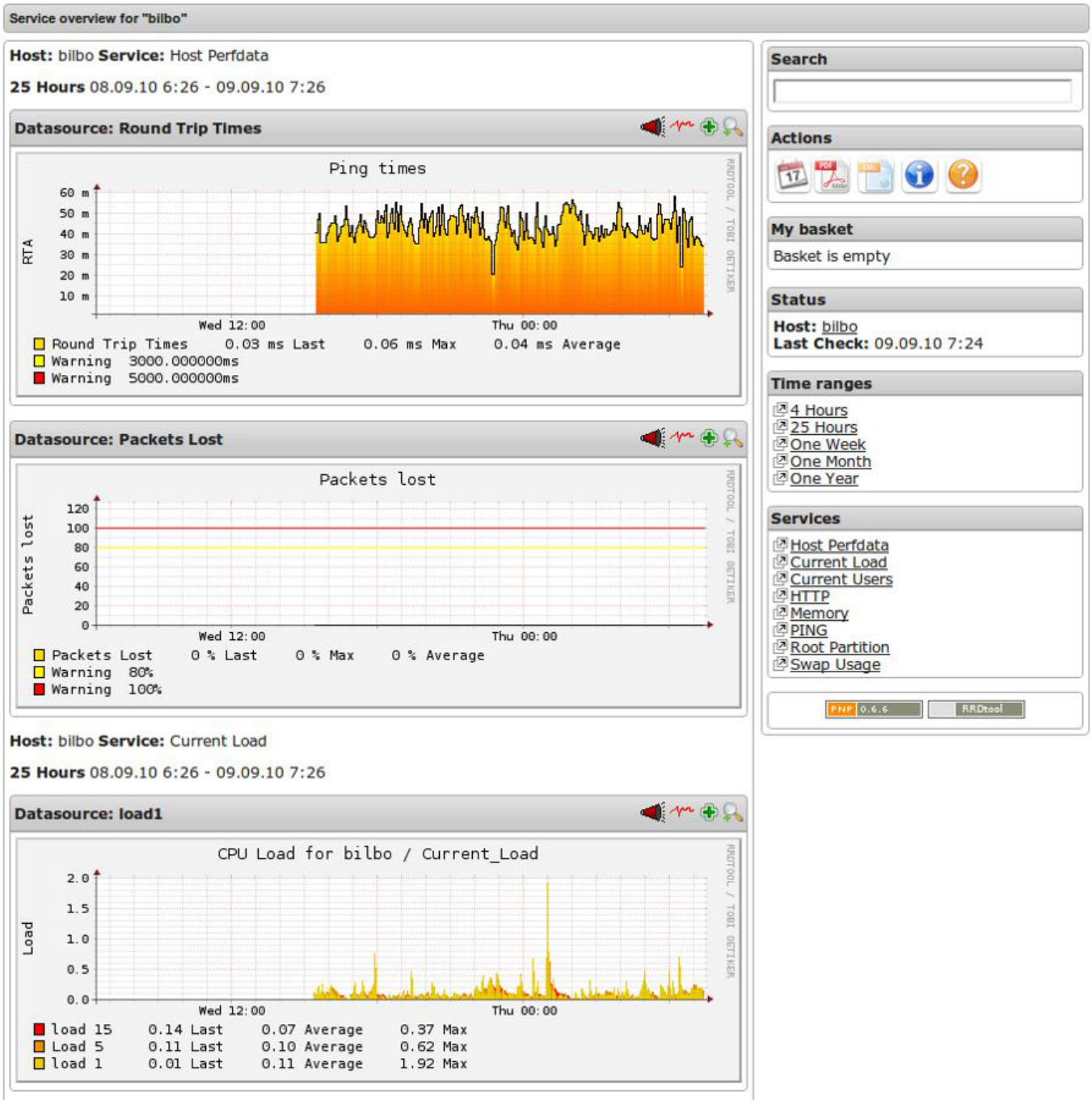
Para el envío de notificaciones por parte de Nagios es necesario configurar un MTA (Agente de Transporte de Correos), por lo general suele estar previamente configurado el Sendmail en el caso de que nos basemos en un CentOS, o un Postifx tratándose de un SuSE o Debian.

En caso de que utilizemos el paquete de Postfix disponible en la distribución, y debamos utilizar otro servidor saliente SMTP dentro de la red, deberemos configurar el relayhost apuntando al servidor SMTP de la red local ej 192.168.1.110, quedando el archivo /etc/postfix/main.cf con las siguientes entradas de configuración.

```
queue_directory = /var/spool/postfix
command_directory = /usr/sbin
daemon_directory = /usr/lib/postfix
mail_owner = postfix
myhostname = nagios
mydomain = midominio.net
inet_interfaces = $myhostname
unknown_local_recipient_reject_code = 550
debug_peer_level = 2
debugger_command =
         PATH=/bin:/usr/bin:/usr/local/bin:/usr/X11R6/bin
         xxgdb $daemon_directory/$process_name $process_id & sleep 5
sendmail_path = /usr/sbin/sendmail
newaliases_path = /usr/bin/newaliases
mailq_path = /usr/bin/mailq
setgid_group = maildrop
html_directory = /usr/share/doc/packages/postfix/html
manpage_directory = /usr/share/man
sample_directory = /usr/share/doc/packages/postfix/samples
readme_directory = /usr/share/doc/packages/postfix/README_FILES
mail_spool_directory = /var/mail
canonical_maps = hash:/etc/postfix/canonical
virtual_maps = hash:/etc/postfix/virtual
relocated_maps = hash:/etc/postfix/relocated
transport_maps = hash:/etc/postfix/transport
sender_canonical_maps = hash:/etc/postfix/sender_canonical
masquerade_exceptions = root
masquerade_classes = envelope_sender, header_sender, header_recipient
program_directory = /usr/lib/postfix
inet_interfaces = 192.168.1.4
masquerade_domains =
mydestination = $myhostname, localhost.$mydomain
defer_transports =
disable_dns_lookups = no
relayhost = 192.168.1.110
content_filter =
mailbox_command =
mailbox_transport =
smtpd_sender_restrictions = hash:/etc/postfix/access
smtpd_client_restrictions =
smtpd_helo_required = no
smtpd_helo_restrictions =
strict_rfc821_envelopes = no
smtpd_recipient_restrictions = permit_mynetworks,reject_unauth_destination
smtp_sasl_auth_enable = no
smtpd_sasl_auth_enable = no
smtpd_use_tls = no
smtp_use_tls = no
alias_maps = hash:/etc/aliases
mailbox_size_limit = 0
message_size_limit = 10240000
```

Siendo los parámetros : **inet_interfaces** *la ip propia por cual salir, si tenemos una sola placa de red, pondremos esa ip*, **relayhost** *es el host que nos realizará el envío SMTP*.

PNP4Nagios



Para la ejecución de almacenamiento de gráficas deberemos configurar ciertos comandos que obtengan los resultados de la ejecución de comandos y servicios, para ellos deberemos agregar y/o modificar en la configuración de Nagios ciertos parámetros :

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 de process-service-perfdata.

```
enable_environment_macros=1
service_perfdata_command=process-service-perfdata
```

```
host_perfdata_command=process-host-perfdata


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

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

Modo Masivo

En el modo masivo, Nagios escribe los datos a un fichero temporal en un formato predefinido. Este fichero se procesa mediante process_perfdata.pl a intervalos regulares. Nagios controla el arranque y ejecución periódica del recolector de datos.

.

commands.cfg

```
define command{
       command_name    process-service-perfdata-file
       command_line    /usr/local/nagios/pnp4nagios/libexec/process_perfdata.pl
--bulk=/usr/local/nagios/pnp4nagios/var/service-perfdata
}

define command{
       command_name    process-host-perfdata-file
       command_line    /usr/local/nagios/pnp4nagios/libexec/process_perfdata.pl
--bulk=/usr/local/nagios/pnp4nagios/var/host-perfdata
}
```

Modo Masivo con NPCD

Se vuelca la informacion de perfdata en una cola para luego ser procesada por un proceso en segundo plano, lo cual libera gran carga del CPU.

nagios.cfg

```
process_performance_data=1
#
# service performance data
#
service_perfdata_file=/usr/local/nagios/var/service-perfdata
service_perfdata_file_template=DATATYPE::SERVICEPERFDATA\tTIMET::$TIMET$\tHOSTNAME::
$HOSTNAME$\tSERVICEDESC::$SERVICEDESC$\tSERVICEPERFDATA::
$SERVICEPERFDATA$\tSERVICECHECKCOMMAND::$SERVICECHECKCOMMAND$\tHOSTSTATE::
$HOSTSTATE$\tHOSTSTATETYPE::$HOSTSTATETYPE$\tSERVICESTATE::
$SERVICESTATE$\tSERVICESTATETYPE::$SERVICESTATETYPE$\tSERVICEOUTPUT::$SERVICEOUTPUT$
service_perfdata_file_mode=a
service_perfdata_file_processing_interval=15
service_perfdata_file_processing_command=process-service-perfdata-file

#
# host performance data starting with Nagios 3.0
#
host_perfdata_file=/usr/local/nagios/var/host-perfdata
host_perfdata_file_template=DATATYPE::HOSTPERFDATA\tTIMET::$TIMET$\tHOSTNAME::
$HOSTNAME$\tHOSTPERFDATA::$HOSTPERFDATA$\tHOSTCHECKCOMMAND::
$HOSTCHECKCOMMAND$\tHOSTSTATE::$HOSTSTATE$\tHOSTSTATETYPE::
```

```
$HOSTSTATETYPE$\tHOSTOUTPUT::$HOSTOUTPUT$
host_perfdata_file_mode=a
host_perfdata_file_processing_interval=15
host_perfdata_file_processing_command=process-host-perfdata-file
```

commands.cfg

```
define command{
        command_name    process-service-perfdata-file
        command_line    /bin/mv /usr/local/nagios/var/service-perfdata
/usr/local/nagios/var/spool/perfdata/service-perfdata.$TIMET$
 }

define command{
        command_name    process-host-perfdata-file
        command_line    /bin/mv /usr/local/nagios/var/host-perfdata
/usr/local/nagios/var/spool/perfdata/host-perfdata.$TIMET$
 }
```

Y luego ejecutar :

```
/etc/init.d/npcd start
```

Datos de configuración

- **service_perfdata_file**
  - Ruta al archivo temporal que debe contener los datos de rendimiento.
- **service_perfdata_file_template**
  - Formato del archivo temporal. Los datos se definen utilizando Nagios macros.
- **service_perfdata_file_mode**
  - Opción “a” especifica que los datos se insertan como anexo.
- **service_perfdata_file_processing_interval**
  - El intervalo de procesamiento es de 15 segundos
- **service_perfdata_file_processing_command**
  - El comando que habrá de ejecutarse durante dicho intervalo.

Luego se deja ejecutando en segundo plano el demonio npcd para procesar la cola de mensajes.

En modo masivo con NPCD se puede deshabilitar la opcion *enable_environment_macros* para ahorrar carga de CPU, ya que en este modo esta ya no es requerida.

Si en algún momento se llega a colgar el demonio NPCD o si por algo falla el procesamiento de los archivos de perfdata, reiniciamos NPCD y va a reprocesar los archivos en el spool de perfdata para volver a incluir la información faltante en RRD.

Interfaz web

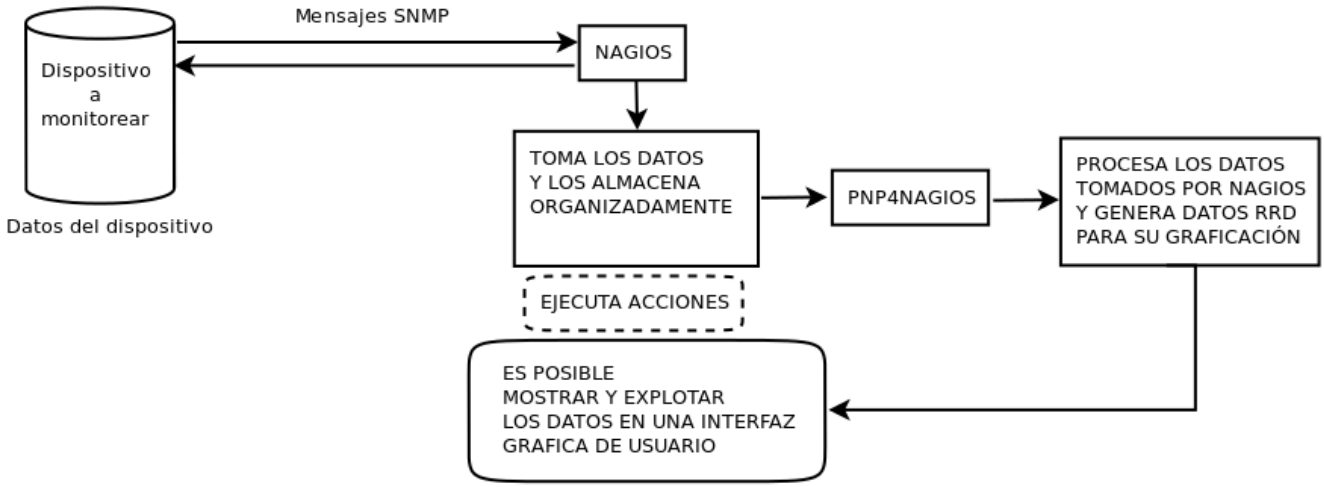
Luego en el template que vayamos a utilizar deberemos agregar estas directivas :

Para los hosts

```
action_url /pnp4nagios/index.php/graph?host=$HOSTNAME$
```

Para los servicios

```
action_url /pnp4nagios/index.php/graph?host=$HOSTNAME$&srv=$SERVICEDESC$
```



nagios_pnp.dia.gz

En los clientes

Podemos elegir monitorear los clientes por medio del protocolo ICMP, SNMP o de mecanismos como ver si tenemos x puerto abierto, o ciertos mensajes de respuesta, aqui se comentan los protocolos mas comunes para monitorear estados de equipos (ICMP/SNMP) o servicios (SNMP).

ICMP

El protocolo ICMP (*Internet Control Message Protocol*) puede ser considerado como parte de la capa IP. La especificación completa de este protocolo se encuentra en RFC 792. Aunque sus mensajes son encapsulados en datagramas IP como cualquier otro protocolo de capa superior, su uso corresponde a mensajes especiales de la propia capa de red, aunque también pueden acceder a él las propias aplicaciones (por ejemplo el programa ping).

Algunos ejemplos de uso de ICMP son: obtención de máscaras de red (solicitud y respuesta), obtención de marcas de tiempo (solicitud y respuesta), condiciones de error del tipo “port unreachable” o “se necesita fragmentar el datagrama habiéndose solicitado la no-fragmentación“, petición y respuesta de eco para comprobar la actividad de otro host, información sobre el estado de las comunicaciones en una red, etc.

Algunas consultas ICMP

Los mensajes de query incluyen tras la cabecera 2 bytes de identificación, para que el cliente distinga las respuestas dirigidas a él de las dirigidas a otros procesos, y 2 btes de número de secuencia, para que el cliente pueda identificar 'requests' con 'replays'. La generación de replys suele correr a cargo del kernel (no es un proceso de usuario). Los mensajes ICMP de 'query' más importantes son:

- ***Petición/respuesta de máscara***: Aunque normalmente se usa DHCP, también es posible para el host hacer una petición durante el proceso de arranque para obtener la máscara y una petición RARP para obtener su IP. La petición de máscara suele enviarse en difusión. Un servidor de máscaras se encarga de contestar.
- ***Petición/respuesta de marca de tiempo***: Se utiliza para conocer el tiempo que un host tarda en procesar un mensaje y contestarlo o para sincronizar relojes entre hosts. Los tiempos se ponen en milisegundos desde las 0h UTC (Tiempo Coordinado Universal). Se proveen tres campos, a parte de los anteriormente citados para este tipo de mensajes. 'Marca de tiempo de generación', a rellenar por el emisor, y 'Marca de tiempo de recepción', y 'Marca de tiempo de transmisión', a rellenar por el receptor. Los significados de los campos resultan evidentes.
- ***Solicitud/anuncio de router***: Se trata de una forma de inicializar la tabla de enrutamiento. Tras el arranque, un host envía en broadcast o multicast un mensaje de solicitud de router. Uno o más routers responderán con mensajes de anuncio de router. Esto lo podrán hacer periódicamente permitiendo a los hosts actualizar sus tablas de enrutamiento.
- ***Petición/respuesta de eco***: Se utiliza para conocer si un host está en red. El mensaje ICMP contiene unos datos OPCIONALES cuyo contenido es irrelevante. Muchos SO implementan este mensaje mediante el programa ping. En las implementaciones UNIX de ping el campo identificador suele asociarse con el ID del proceso que envía el request. Esto permite identificar las respuestas si hubiera varias instancias de ping corriendo en el mismo host. En el campo de secuencia, se suele llevar un contador que permite ver si se han perdido, duplicado o desordenado paquetes, cosa típica en la capa IP sobre la que se encapsulan los mensajes ICMP.

Veamos un ejemplo de la salida del programa ping.

```
[deivid@localhost ~]$ ping www.virtuamanager.com
PING virtuamanager.com (213.194.149.188) 56(84) bytes of data.
64 bytes from virtuamanager.com (213.194.149.188): icmp_seq=1 ttl=56 time=42.6 ms
64 bytes from virtuamanager.com (213.194.149.188): icmp_seq=2 ttl=56 time=41.8 ms
64 bytes from virtuamanager.com (213.194.149.188): icmp_seq=3 ttl=56 time=41.6 ms
64 bytes from virtuamanager.com (213.194.149.188): icmp_seq=4 ttl=56 time=43.7 ms
64 bytes from virtuamanager.com (213.194.149.188): icmp_seq=5 ttl=56 time=42.7 ms
64 bytes from virtuamanager.com (213.194.149.188): icmp_seq=6 ttl=56 time=42.2 ms
64 bytes from virtuamanager.com (213.194.149.188): icmp_seq=7 ttl=56 time=46.2 ms

--- virtuamanager.com ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6005ms
rtt min/avg/max/mdev = 41.666/43.038/46.270/1.477 ms
```

ALGUNOS ICMP'S DE ERROR

Ya se ha dicho que el formato de los mensajes de error tan sólo incluye, además de la cabecera, una copia de la cabecera IP del datagrama que generó el error y los 8 primeros bytes del datagrama. En algunos de los ejemplos que siguen veremos la razón de ser de este formato:

- ***Destination unreachable***: Este tipo de mensajes de error se generan cuando por alguna razón, el datagrama no pudo alcanzar su destino (puerto, host, red inalcanzable, necesidad de fragmentar con bit DF activado...). La forma de comunicar este error a la capa superior es copiando la cabecera IP en el propio

mensaje de error. Gracias al campo 'protocolo' el módulo ICMP que lo reciba sabrá asociarlo con el protocolo adecuado. Supongamos que se produce un 'port unreachable' en el envío de un datagrama UDP. La cabecera IP permite identificar que el error se produjo en el protocolo UDP, y además esto sirve para interpretar los 8 bytes adicionales, que pertenecerán a la cabecera UDP. Estos contendrán los puertos origen y destino (ver tema UDP). El puerto de origen puede servir para asociar el error con un determinado proceso de usuario (ej. puerto 2924 asociado a un cliente tftp), mientras que el destino nos indica qué puerto originó el error 'port unreachable'.

Otro ejemplo, al intentar hacer ping al otro equipo de la red, que está apagado, se obtiene lo siguiente:

```
[deivid@localhost ~]$ ping 192.168.1.2
PING 192.168.1.2 (192.168.1.2) 56(84) bytes of data.
From 192.168.1.100 icmp_seq=2 Destination Host Unreachable
From 192.168.1.100 icmp_seq=3 Destination Host Unreachable
From 192.168.1.100 icmp_seq=4 Destination Host Unreachable
From 192.168.1.100 icmp_seq=6 Destination Host Unreachable
From 192.168.1.100 icmp_seq=7 Destination Host Unreachable
From 192.168.1.100 icmp_seq=8 Destination Host Unreachable

--- 192.168.1.2 ping statistics ---
9 packets transmitted, 0 received, +6 errors, 100% packet loss, time 8002ms
, pipe 3
```

Se han producido mensajes ICMP de error en respuesta a mensajes ICMP de query, lo cual SI está permitido, como ya se ha dicho.

- ***Source quench***: Este tipo de mensaje es generado por un router con problemas de congestión para avisar al host fuente de que reduzca el flujo de transmisión. Es muy poco usado, normalmente los routers se limitan a tirar paquetes y a intentar solucionar la congestión mediante algoritmos de enrutamiento.
- ***Parameter problem***: Si un router intermedio o el host destino encuentran un problema en la cabecera IP que le obliga a desecharlo, puede enviar un mensaje ICMP de este tipo al host fuente. Si el código es cero, un puntero marca el lugar de la cabecera en el que se produjo el error.
- ***Redirección***: Son enviados por un router al remitente de un datagrama IP cuando éste debería haber sido enviado a través de otro router.
- ***Time Exceeded***: Son enviados al remitente de un datagrama IP cuando se cumple el tiempo de vida máximo del paquete. Esto puede ocurrir porque el campo TTL del datagrama se haya puesto a 0 en un router, o bien el tiempo de reensamblado de fragmentos se haya cumplido en el destino. Basándose en este mensaje, y variando el campo TTL de un mensaje de petición de eco es posible obtener la ruta a un destino. Si el TTL es insuficiente, se recibirá un mensaje de tiempo excedido por cada router donde al recalcular el TTL valga 0, y un mensaje de respuesta de eco cuando el TTL sea suficiente para llegar al destino. El comando `tracert` de windows se basa en este algoritmo.

SNMP (Simple Network Management Protocol)

Protocolo Simple de Administración de Red o SNMP es un protocolo de la capa de aplicación que facilita el intercambio de información de administración entre dispositivos de red. Es parte de la familia de protocolos TCP/IP. SNMP permite a los administradores supervisar el desempeño de la red, buscar y resolver sus problemas, y planear su crecimiento. Podemos conocer datos internos de

dispositivos a monitorear, ej uso de CPU, Memoria, Disco, Uptime etc.

Se basa entre gestor y agentes.

Los agentes se centran en recopilar cierta información y mantenerla organizada para que el gestor pueda acceder a ella cuando lo necesite. El agente también puede mandar notificaciones al gestor sobre problemas o información relevante sin que el gestor se lo solicite.

El gestor controla y actúa sobre los elementos de la red, controlando la información que recopilan los agentes se pueden tomar decisiones y actuar sobre la red para mejorar los aspectos que se necesiten. Para ello se basa en tres elementos:

- **SNMP**: Es el protocolo entre el gestor y el elemento de red, se suele utilizar UDP
- **SMI**: (Structure of Management Information): conjunto de estructuras y esquemas de identificación para definir las MIB´s
- **MIB**: (Management Information Base): especifica qué variables mantienen los elementos de la red (agentes)

SMI (Structure of Management Information)

SMI es el conjunto de estructuras y esquemas para definir elementos de las MIB. Dichas estructuras están formateadas en ASN.1 (Abstract Synstax Notation One), indicando cómo debe ser el nombre, la sintaxis y el método de codificación de los datos para su transmisión por la red.

Un objeto en SMI consta de los siguientes campos:

- Objeto: Literal (Object descriptor) con la etiqueta que identifica el objeto
- Sintaxis: Puede ser alguno de los siguientes tipos ASN.1:
  - SimpleSyntax (INTEGER, OCTET STRING, OBJECT IDENTIFIER, NULL)
  - ApplicationSyntax (NetworkAddress (direcciones ip de 32 bits), Counter (enteros no negativos de tamaño 32bits), Gauge (contadores que mantienen el maximo valor alcanzado, pueden decrementarse en algun momento), TimeTicks (centésimas de segundo desde el evento deseado), Opaque (información arbitraria))
  - Otros tipos (RFC 1155).
- Acceso: Puede ser: read-only, read-write, write-only, not-acessible. SMIv2 incluye además read-create, y deja de usar write-only
- Status: Puede ser: mandatory (en SMIv2 se indica como current), que tienen que ser implementadas por cualquier versión de la MIB que incluya ese modulo, optional, que pueden faltar sin que eso cause ningún problema (en vez de optional, SMIv2 incluye deprecated, para objetos que ya no se usan, y que por tanto, no tienen por qué estar implementados), u obsolete, que han dejado de mantenerse y revisarse.
- Descripción: Definición textual de la semántica del tipo de objeto. No es codificable mediante una computadora, es una descripción para programadores y administradores que pueden leerla para entender el funcionamiento de la MIB.

La definición termina indicando bajo qué nodo del árbol de OIDs (ver la siguiente sección) debe situarse y con qué número a efectos de identificación

MIB: (Management Information Base)

La información requerida de un sistema se almacena en las MIBs usando una estructura jerárquica que contiene los identificadores de objeto descritos mediante ASN.1.

Esta jerarquía de árbol contiene los elementos que pueden ser de tipo escalar o tablas de datos. Estos objetos cuelgan en el árbol de la MIB de la rama correspondiente a la organización que mantiene dicha estructura. Se nombra a los ejemplares de la MIB mediante su identificador de objeto que es una cadena de enteros que representa como cuelga el objeto del nodo raíz. Por ejemplo IP es el (1.3.6.1.2.1.4.)

Esta es la estructura de la MIB-II (RFC 1213) y sus diferentes subgrupos. Estas son algunas de las variables que se mantienen en cada grupo:

- Grupo system: Descripción de la entidad, identificador, tiempo desde arranque, nombre del administrador, localización física, servicios ofrecidos
- Grupo interfaces: Número de interfaces del sistema
- Grupo at: Número de interfaz, Dirección física, Dirección IP
- Grupo ip: Si el sistema hace forward, valor del TTL, número de datagramas recibidos y enviados, errores, datagramas con protocolo no válido, etc.
- Grupo icmp: cuatro contadores generales: número total de mensajes ICMP de entrada y salida con o sin errores y 22 contadores para los diferentes mensajes ICMP
- Grupo tcp: Algoritmo de retransmisión, timeout en milisegundos, número de conexiones TCP, número de transiciones entre los diferentes estados de TCP, número de segmentos recibidos y enviados, número de segmentos retransmitidos, con error y con el flag RST activado.
- Grupo udp: Número de datagramas enviados y recibidos, datagramas sin proceso receptor
- Grupo egp: Número de mensajes de encaminamiento recibidos con y sin error, número de mensajes generados en el sistema, estado del sistema.

Funcionamiento del protocolo

Como sigue el paradigma gestor-agente la los comandos son para solicitar, modificar o devolver información de la MIB, así como envíar notificaciones (traps). Las operaciones disponibles en SNMPv1 son:

- Get-request : Solicitar el valor de una o varias variables
- Get-next-request : Solicitar el valor de la siguiente variable o grupo de variables. Se utiliza para recorrer tablas, celda a celda.
- Set-request : Dar valor a una variable
- Get-response : Devolver el valor de una o varias variables
- Trap : El agente notifica al gestor de alguna situación especial sucede en el elemento de red. Reinicialización, fallo y recuperación en un enlace

Posteriormente, a partir de la versión 2 se introdujo la operación de get-bulk-request, que permite realizar varios get-next seguidos sin tener que hacer varias peticiones. También se introdujo la operación de inform, para que distintos gestores puedan intercambiarse notificaciones.

SNMPv1

Ejemplo de configuración SNMPv3 para un sistema típico Linux/UNIX/AIX/Solaris etc

```
rocommunity public 192.168.1.4
syscontact Sysadmin (root@localhost)
syslocation Server Room

view all included .1
access notConfigGroup "" any noauth exact systemview none none
access notConfigGroup "" any noauth exact all none none
```

Donde rocommunity es una comunidad de solo lectura, de nombre public y que solo se le permita el acceso desde la ip 192.168.1.4, el nombre de la comunidad define la dificultad de adivinar su acceso, por eso es recomendable que tenga un nombre similar a una contraseña.

SNMPv3

Configuramos un usuario que se llama nagios con autenticacion, en la siguiente linea lo creamos con una clave almacenada en texto plano en este archivo y luego encriptada en MD5 y DES

Ejemplo de configuración SNMPv3 para un sistema típico Linux/UNIX/AIX/Solaris etc

```
rouser  nagios auth
createUser nagios MD5 "ClaV3RfsTtD2" DES
```

Debemos reiniciar el servicio SNMP y luego podemos hacer algo como esto

snmpget -v 3 -u nagios -l authNoPriv -a MD5 -A ClaV3RfsTtD2 127.0.0.1 sysUpTime.0

TIPS

NOTA: En Debian deberemos editar el archivo */etc/default/snmpd*, modificando la siguiente linea

De:

```
SNMPDOPTS='-Lsd -Lf /dev/null -u snmp -I -smux -p /var/run/snmpd.pid 127.0.0.1'
```

A :

```
SNMPDOPTS='-Lsd -Lf /dev/null -u snmp -I -smux -p /var/run/snmpd.pid'
```

Para que el servicio se accesible por red.

A veces por causa de tener montados filesystems remotos NFS o CIFS, cuando tenemos retrasos en la conexion con ellos se puede colgar el servicio SNMP, para prevenir esto deberemos agregar la siguiente opción en el archivo de configuración *snmpd.conf*.

```
skipNFSInHostResources yes
```

Agregar variables a SNMP

Por ejemplo si queremos obtener desde SNMP la información de un script

/usr/local/bin/var.sh

```
echo "hola"
echo "chau"
```

En el snmpd.conf le agregamos la siguiente configuración:

```
exec .1.3.6.1.4.1.2021.555 /bin/sh /usr/local/bin/var.sh
```

Si corremos esta consulta **snmpwalk -v 1 -c public localhost .1.3.6.1.4.1.2021.555** Obtendremos :

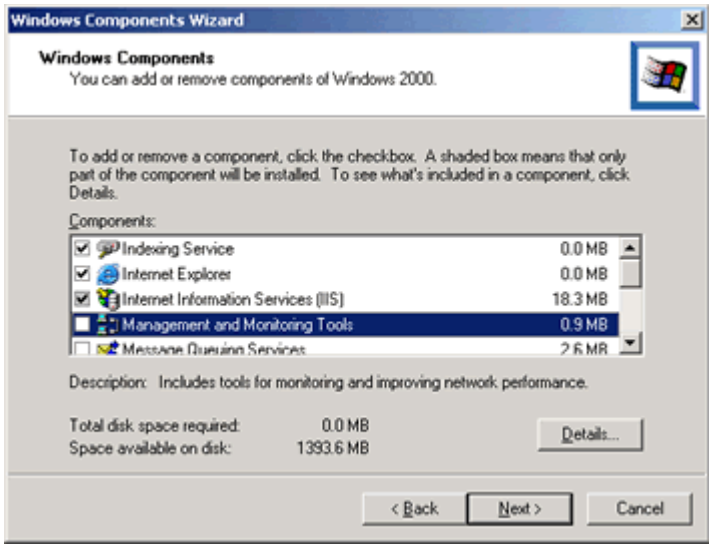
```
UCD-SNMP-MIB::ucdavis.555.1.1 = INTEGER: 1
UCD-SNMP-MIB::ucdavis.555.2.1 = STRING: "/bin/sh"
UCD-SNMP-MIB::ucdavis.555.3.1 = STRING: "/tmp/bar.sh"
UCD-SNMP-MIB::ucdavis.555.100.1 = INTEGER: 0
UCD-SNMP-MIB::ucdavis.555.101.1 = STRING: "hola"
UCD-SNMP-MIB::ucdavis.555.101.2 = STRING: "chau"
UCD-SNMP-MIB::ucdavis.555.102.1 = INTEGER: 0
UCD-SNMP-MIB::ucdavis.555.103.1 = ""
```

SNMPv1 en Windows

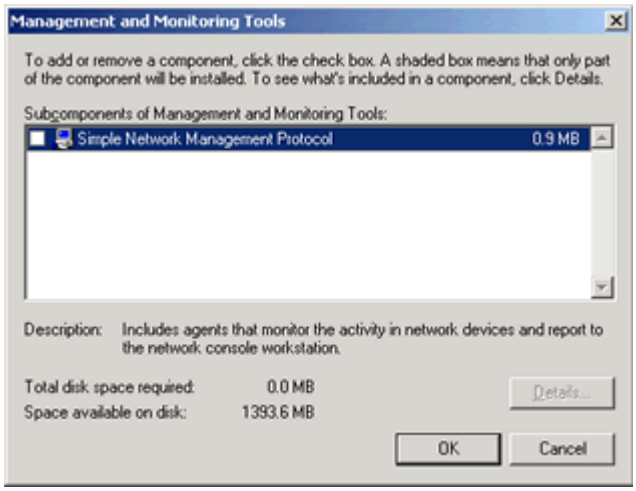
En los equipos Windows se configurara una community RO (read only) denominada public, acceso para la ip que corresponda al servidor que tiene acceso al SNMP, en este caso nuestro nagios.

Para ello nos situamos en el panel de control dentro de agregar programas en la sección de componentes.

Una vez dentro nos situamos dentro de Herramientas de administración y supervisión o Management and Monitoring tools

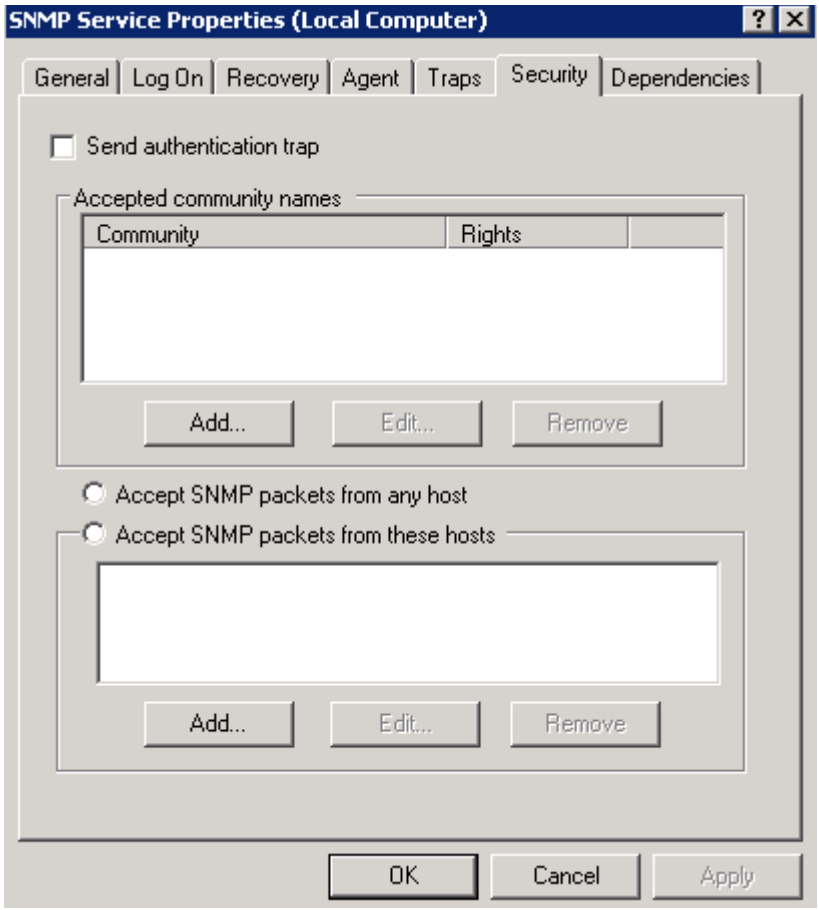


Dentro de ella nos situamos sobre la opcion SNMP y la tildamos

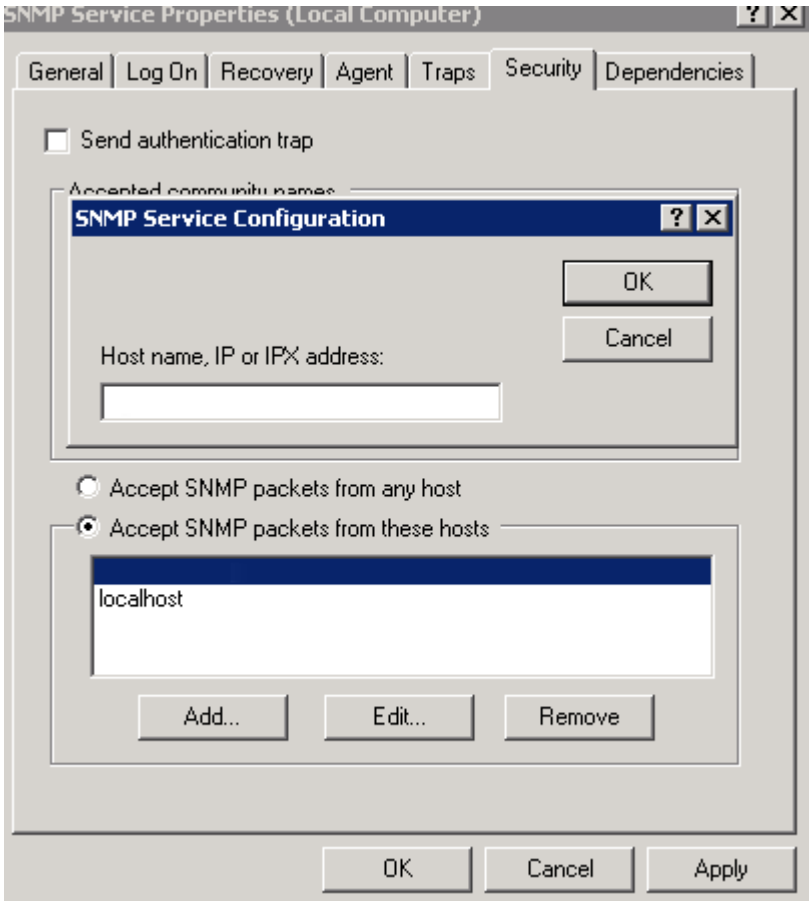


Luego en la Administración de Servicios, nos dirigiremos a las propiedades del servicio SNMP

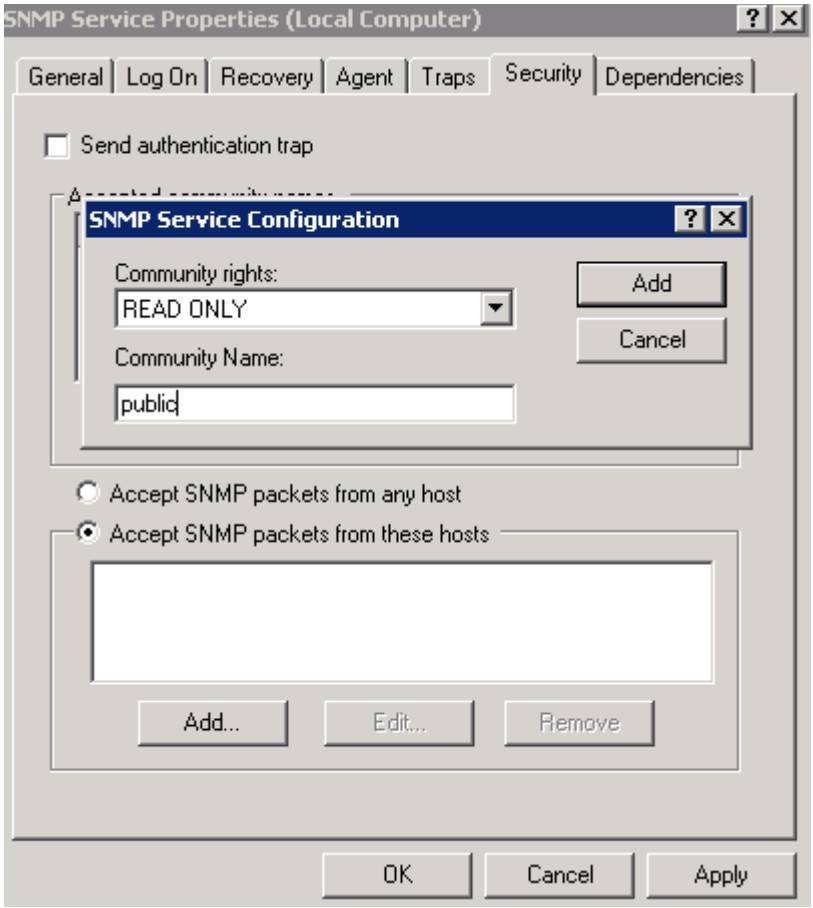
Dentro de ella nos encontraremos con un cuadro de dialogo



Configuraremos el accesso a la ip del servidor con acceso SNMP



Y luego la comunidad de aceso a los datos SNMP



Luego de realizados los cambios de configuración procederemos a reiniciar el servicio para que estos surtan efecto.

A veces en Windows es necesario configurar el servicio de Firewall para permitir las

consultas por medio SNMP.

Ref.: http://support.microsoft.com/kb/324263/es

NRPE

NSClient++

```
[/settings/external scripts/scripts]
check_pagefile="C:\Program Files\Nagios Plugin Collection\check_pagefile.exe"
```

En el Monitoreo

Creando directivas

Debemos crear algunas entradas de configuración para especificar donde encontramos los servicios, grupos, contactos etc, las mismas debemos incluirlas en nuestro archivo de configuración nagios.cfg

```
# Directorio con la configuración de grupos de Hosts de los Servidores
cfg_dir=/usr/local/nagios/etc/hostgroups
# Directorio con la configuración de grupos de servicios de los Servidores
cfg_dir=/usr/local/nagios/etc/servicegroup
# Directorio con la configuración de contactos
cfg_dir=/usr/local/nagios/etc/contacts
# Directorio con la configuración de grupos de contacto
cfg_dir=/usr/local/nagios/etc/contactgroups
# Directorio con la configuración de servicios
cfg_dir=/usr/local/nagios/etc/services
# Directorio con la configuración de los comandos
cfg_dir=/usr/local/nagios/etc/commands
# Directorio con la configuración de los equipos a monitorear
cfg_dir=/usr/local/nagios/etc/servers
```

Con la directiva cfg_dir el indicamos Nagios que tome como configuración los archivos con extencion “cfg” encontrados en tal directorio.

Configuración de alertas

Agregando Grupos de contacto

Para que el Nagios envíe notificaciones sobre el estado de los servicios es necesario definir grupos a los cuales enviárselas, y dentro de ellos estarán los miembros a cuales enviarlos

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

```
   define contactgroup{
        contactgroup_name               admin
        alias                           Administrators
        members                         admin-sap,admin-windows
    }
```

```
   define contactgroup{
        contactgroup_name               {nombre del grupo contacto}
        alias                           {descripcion}
        members                         {miembros del grupo}
    }
```

contactgroup_name

Nombre que se le asignara al grupo de contacto

alias

Descripción corta para identificar al grupo

members

Miembros del grupo separados por comas

Se deberá crear el archivo {nagios-dir}/etc/contactgroups/{nombregrupodecontacto.cfg} con las entradas correspondientes anteriormente explicadas.

Agregando Contactos

Para recibir las notificaciones de Nagios es necesario generar 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                    admin
        alias                           Administrador Nagios
        contactgroups                   admin
        service_notification_period     24x7
        host_notification_period        24x7
        service_notification_options    w,u,c,r
        host_notification_options       d,u,r
        service_notification_commands   notify-by-email
        host_notification_commands      host-notify-by-email
        email                           root@localhost
    }


   define contact{
        contact_name                    {nombre del contacto}
        alias                           {descripcion del contacto}
        contactgroups                   {grupo de contactos al cual pertenece}
        service_notification_period     {priodo de tiempo de notificaciones de
servicios}
        host_notification_period        {priodo de tiempo de notificaciones de hosts}
        service_notification_options    {opciones de notificacion por servicio}
        host_notification_options       {opciones de notificacion por host}
        service_notification_commands   {comando de notificacion a utilizar por
servicio}
        host_notification_commands      {comando de notificacion a utilizar por host}
        email                           {direccion de email del contacto}
    }
```

contact_name

Nombre literal a asignarle al contacto

alias

Alias descriptivo del contacto, ej Administrador de Routers

contactgroups

Grupos de contactos a los cuales pertenece

service_notification_period

Periodo de tiempo en el cual recibir notificaciones sobre el estado de los servicios

host_notification_period

Periodo de tiempo en el cual recibir notificaciones sobre el estado de los hosts

service_notification_options

Opciones de cuando realizar una notificación sobre el estado de un servicio

host_notification_options

Opciones de cuando realizar una notificación sobre el estado de un host

service_notification_commands

Comando para realizar la notificación del estado del servicio, podemos definir múltiples comandos

host_notification_commands

Comando para realizar la notificacion del estado del host, podemos definir múltiples comandos

email

Email perteneciente al contacto en el cual recibira las notificaciones por email. Para que esto funcione se debe tener correctamente configurado el mail delivery local.

Se deberá crear el archivo {nagios-dir}/etc/contacts/{nombredecontacto.cfg} con las entradas correspondientes anteriormente explicadas.

Alertas en formato HTML

Podemos estilizar nuestras alertas por email de nagios con colores, símbolos eh imágenes.

Aquí adjunto dos scripts descargados de nagiosexchange que están modificados por mi :

nagios_send_host_mail.pl.gz nagios_send_service_mail.pl.gz

Alertas por SMS

Podemos configurar alertas SMS de forma muy simple con el software Gnokii.

Ejemplo de archivo de configuración de Gnokii

```
[global]
port = /dev/ttyUSB2
model = AT
initlength = default
connection = serial
use_locking = yes
serial_baudrate = 19200
smsc_timeout = 10

[connect_script]
TELEPHONE = 12345678
[disconnect_script]
[fake_driver]
[flags]
```

```
[gnokii]
[gnokiid]
[logging]
debug = on
rlpdebug = off
xdebug = off
```

Modo de uso :

echo “Hola mundo” | sudo gnokii –sendsms 1155667798

Definición como comando de notificación, (podemos agregarlo como comando extra al contacto, para que ademas del comando común de notificación por email tambien se ejecute este para el envío de la alerta por sms).

Notemos que lo ejecutamos por ssh, es para una situación hipotética de que el modem USB 3G de nuestro proovedor de telefonía celular este enchufado en otra maquina de nuestra red.

```
define command {
 command_name    host-notify-by-sms
 command_line    ssh nagios@10.15.99.11 "echo "Host '$HOSTNAME$' esta $HOSTSTATE$
$HOSTOUTPUT$" | sudo gnokii --sendsms $CONTACTPAGER$"
}


define command {
 command_name    service-notify-by-sms
 command_line    ssh nagios@10.15.99.11 "echo 'Servicio $SERVICEDESC$ en Host
$HOSTNAME$ IP $HOSTADDRESS$ State $SERVICESTATE$ Info $SERVICEOUTPUT$' | sudo gnokii
--sendsms $CONTACTPAGER$"
}
```

nagios_sms.svg.gz

Agregando Comandos

En Nagios los encargados de recabar los datos del monitoreo, de mostrar alertas, de todas las tareas, son los comandos.

Los mismos se dividen en comandos de performance y en comandos de chequeo, los primeros son utilizados para algunos casos en particular.

Los comandos de chequeo no traen datos de los equipos a monitorear, como consumo de CPU, Memoria, Disco, procesos corriendo, puertos abiertos etc, es decir

todos los datos necesarios de la monitoria.

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

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

```
define command{
        command_name check_snmp_mem
        command_line $USER1$/check_snmp_mem.pl -H $HOSTADDRESS$ $ARG1$ -w $ARG2$ -c
$ARG3$ $ARG4$
        }


define command{
        command_name {nombre del comando}
        command_line {datos de  ejecucion}
        }
```

command_name

El nombre que el comando tendra para nuestra configuración personal de Nagios

command_line

Modo del cual Nagios ejecutara el comando en cuestión, con su ruta física y argumentos Lo que vemos en entre signos $ son variables internas de nagios, llamadas macros, las mas comunes son:

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

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

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

Podemos definir nuestros propios macros seteando variables en el archivo resource.cfg

Se deberán agregar al archivo {nagios-dir}/etc/commands.cfg las entradas correspondientes anteriormente explicadas.

Agregando Grupos de Servicios

Los grupos de servicio se utilizan para denotar un variedad de servicios sobre otros, debemos tener asignado aunque sea un servicio a ese grupo por que si no de lo contrario el Nagios mostrara un error al arranque, para eso lo asignamos en la propiedad *servicegroups* de un servicio en particular. Ejemplo de una entrada de grupo de servicios:

```
define servicegroup{
        servicegroup_name       lotus_response
        alias                   Lotus Reponse Services
        }


define servicegroup{
        servicegroup_name       {nombre corto del grupo de servicio}
        alias                   {alias descriptivo completo del grupo}
        }
```

Se deberá crear el archivo {nagios-dir}/etc/servicegroup/ {nombregrupodeservicios.cfg} con las entradas correspondientes anteriormente

explicadas.

Agregando Servicios

A continuacion se muestra una tipica entrada de configuración de un servicio

```
define service {
        use                             windows
        host_name                       srv1,srv2
        hostgroup_name                  servidores-windows
        service_description             Verification disco F:
        servicegroups                   storage
        is_volatile                     0
        check_period                    24x7
        max_check_attempts              3
        normal_check_interval           5
        retry_check_interval            1
        contact_groups                  windows
        notification_interval           240
        notification_period             24x7
        notification_options            c,r
        check_command                   check_snmp_storage!^F!60!90!-C public!-r
}

define service {
        use                             {template de servicio a utilizar}
        host_name                       {hosts que ejecutan dicho servicio}
        hostgroup_name                  {grupos de host que ejecutan ese servicio}
        service_description             {descripcion del servicio}
        servicegroups                   {grupo al cual pertenece}
        is_volatile                     {si el servicio es volatil}
        check_period                    {periodo de tiempo para el chequeo}
        max_check_attempts              {maximo de intentos de chequeo}
        normal_check_interval           {intervalo de tiempo a programar los
chequeos}
        retry_check_interval            {intervalo de tiempo para un re-chequeo}
        contact_groups                  {grupo de contacto};
        max_check_attempts              {maxima cantidad de chequeos}
        notification_interval           {intervalo de tiempo entre notificaciones}
        notification_period             {priodo de tiempo de notificaciones}
        notification_options            {cuando enviar notificaciones}
        check_command                   {comando de chequeo con sus argumentos}
}
```

use

Template de servicio a utilizar

host_name

Nombre del o los host a los cuales esta asignado dicho servicio

hostgroup_name

Nombre del grupo de host en los cuales esta asignado dicho servicio, es útil para cuando se vuelve tedioso poner uno por uno los nombres de los hosts a los cuales se asigna el servicio

service_description

Alias descriptivo del nombre del servicio

contact_groups

Grupos de contacto a los que enviar las notificaciones

max_check_attempts

Maxima cantidad de chequeos a efectuar por Nagios, antes de enviar un OK como resultado

normal_check_interval

Intervalo de tiempo antes de programar un nuevo chequeo del servicio

retry_check_interval

Intervalo de tiempo antes de realizar un re-chequeo del servicio

notification_interval

Esta directiva se utiliza para definir el número de las “unidades del tiempo” para esperar antes de re-notificar a un contacto que este servidor todavía está abajo o inalcanzable. A menos que se haya cambiado la directiva interval_length del valor prefijado de 60, este número significará minutos. Si este valor se establece a 0, Nagios re-no notificará contactos sobre los problemas para este host - solamente una notificación del problema será enviada.

notification_period

Periodo de tiempo en el cualse envia notificacion y notificacion

notification_options

Esta directiva indica a Nagios en que momentos debe enviar notificaciones de estado

- d = DOWN cuando el host esta caido
- u = UNREACHABLE cuando el host no es visible o es inalcanzable
- r = RECOVERY (OK) cuando el host se recupero
- f = FLAPPING cuando es de estado cambiante
- n = NONE no enviar notificaciones

check_command

Comando para efectuar el chequeo de dicho servicio

Se deberá crear el archivo {nagios-dir}/etc/services/{nombreservicio.cfg} con las entradas correspondientes anteriormente explicadas.

Dependencia del estado del servicio por el estado de otro servicio

En caso en los cuales el estado de un servicio dependa de la disponibilidad o el estado de otro, se pueden definir dependencias. Una entrada a modo ejemplo puede ser:

```
define servicedependency{
      host_name               Host A
      service_description           Service A
      dependent_host_name           Host B
      dependent_service_description Service D
      execution_failure_criteria    u
      notification_failure_criteria n
      }


define servicedependency{
      host_name               {host donde se ejecuta el servicio dependiente}
      service_description           {servicio dependiente}
      dependent_host_name           {host donde se ejecuta el servicio del cual se
```

```
depende}
      dependent_service_description {servicio del cual se depende}
      execution_failure_criteria    {criterio para establecer el estado}
      notification_failure_criteria {notificar segun x estado}
      }
```

host_name

Nombre del o los host dentro de los cuales se ejecuta el servicio dependiente

service_description

Descripcion del servicio dependiente, debe ser igual a la entrada que aparece en la configuracion del servicio.

dependent_host_name

Host donde se esta ejecutando el servicio del cual se depende

dependent_service_description

Nombre descriptivo que corresponde al servicio del cual se depende, debe ser igual al de su configuracion de servicio

execution_failure_criteria

Criterios para definir el estado del servicio

notification_failure_criteria

En base a que estado realizar las notificaciones, si esta caido, si esta ok o no realizar notificaciones

Informacion extendida de servicio

En algunos casos podemos agregar un link informativo u externo haciendo referencia al servicio en ejecution

```
define serviceextinfo{
      host_name         linux2
      service_description     Carga del sistema Linux
      notes             Carga sistema
      notes_url         http://localhost/cargalinux.pl?
host=linux2&service=Carga+Sistema
      icon_image        carga.png
      icon_image_alt          Alertas de carga
      }


define serviceextinfo{
      host_name         {nombre del host}
      service_description     {nombre descriptivo del servicio}
      notes             {nota descriptiva sobre la informacion extra}
      notes_url         {url donde se encuentra la informacion extra}
      icon_image        {imagen de icono}
      icon_image_alt          {texto alternativo de la imagen}
      }
```

host_name

Host donde se ejecuta dicho servicio

service_description

Nombre descriptivo del servicio al cual corresponde la informacion extra

notes

Descripcion sobre “que es” o “a que corresponde” la información extra a mostrar

notes_url

Url donde encontrar la informacion extra

icon_image

Icono a referenciar el link de la información extra

icon_image_alt

Texto alternativo del icono

Agregando Grupos de Hosts

Los host en nagios de puede agrupar mediante grupos y asi tener un listado aparte que los diferencia de los demas. por ejemplo tener por un lado los servidores SAP con Oracle y por otro los servidores Lotus, o Linux y Windows por separado etc.

Un archivo tipo de grupos de host se presenta a continuacion

```
define hostgroup {
    hostgroup_name                  ramallo
    alias                           Equipos de Ramallo
    members                         ramallo,slnra01,srvrmlofs
}


define hostgroup {
    hostgroup_name                  {nombre del grupo}
    alias                           {alias descriptivo}
    members                         {host miembros}
}
```

hostgroup_name

Nombre del grupo de hosts

alias

Alias descriptivo del grupo

members

Host que son miembros del grupo, debemos ingresar el host_name de cada uno separado por comas ”,“

Se deberá crear el archivo {nagios-dir}/etc/hostgroups/{nombregrupodehosts.cfg} con las entradas correspondientes anteriormente explicadas.

Agregando Hosts

Para configurar un host con o sin SNMP previamente instalado y configurado como lo indicado anteriormente, para su posterior monitoreo. Se debe crear una entrada en la configuracion de Nagios.

Un tipico archivos hosts.cfg

```
  define host{
        use                             servidores
        host_name                       servidorsap2
        hostgroup_name                  servidores-linux
        alias                           SAP SERVER
        address                         192.168.10.84
        parents                         buenos_aires
```

```
        contact_groups                  linux;
        max_check_attempts              10
        notification_interval           120
        notification_period             24x7
        notification_options            d,u,r
    }


  define host{
        use                             {template-host}
        host_name                       {nombre-host}
        hostgroup_name                  {grupos al que pertenece este host}
        alias                           {alias-descriptivo}
        address                         {ip}
        parents                         {host del que depende}
        contact_groups                  {grupo de contacto};
        max_check_attempts              {maxima cantidad de chequeos}
        notification_interval           {intervalo de tiempo entre notificaciones}
        notification_period             {priodo de tiempo de notificaciones}
        notification_options            {cuando enviar notificaciones}
    }
```

use

Template de host a utilizar

host_name

Nombre del host

hostgroup_name

Grupos a los que pertenece este host

address

Direccion IP del host

parents

Host del que depende y que esta delante suyo, por ejemplo puede ser un router o un equipo que le brinde la conectividad etc, y en el mapa se dibujara como dependiente de ese nodo

contact_groups

Grupos de contacto a los que enviar las notificaciones

max_check_attempts

Maxima cantidad de chequeos a efectuar por Nagios, antes de enviar un OK como resultado

notification_interval

Esta directiva se utiliza para definir el número de las “unidades del tiempo” para esperar antes de re-notificar a un contacto que este servidor todavía está abajo o inalcanzable. A menos que se haya cambiado la directiva interval_length del valor prefijado de 60, este número significará minutos. Si este valor se establece a 0, Nagios re-no notificará contactos sobre los problemas para este host - solamente una notificación del problema será enviada.

notification_period

Periodo de tiempo en el cualse envia notificacion y notificacion

notification_options

Esta directiva indica a Nagios en que momentos debe enviar notificaciones de estado

- d = DOWN cuando el host esta caido
- u = UNREACHABLE cuando el host no es visible o es inalcanzable
- r = RECOVERY (OK) cuando el host se recupero
- f = FLAPPING cuando es de estado cambiante
- n = NONE no enviar notificaciones

Se deberá crear el un subdirectorio correspondiente al nombre del host y segun corresponda ubicarlo en el directorio servers/{linux-windows-lotus} o routers y deentro crear un archivo hosts.cfg con la configuración anteriormente explicada,

Informacion extendida de host

La informacion extendida de host se utiliza para el look and feel de los host dentro de los mapas de estado, ya sea tanto el 2D como el 3D

```
define hostextinfo{
      host_name       linuxoracle
      notes           Servidor Oracle de uniface
      icon_image      oracle.png
      icon_image_alt  Oracle
      vrml_image      oracle.png
      statusmap_image oracle.gd2
  }

define hostextinfo{
      host_name       {nombre de host}
      notes           {descripcion para el host}
      icon_image      {logo para ver en la interfaz web}
      icon_image_alt  {texto para el logo}
      vrml_image      {logo para ver en el mapa 3D}
      statusmap_image {logo para ver en el mapa 2D}
  }
```

host_name

Nombre del host al cual corresponde la información dada

notes

Pequeña nota descriptiva de la informacion del host a presentar en los mapas de estado

icon_image

Icono a visualizar en el entorno html de la consola Nagios

icon_image_alt

Texto alternativo para mostrar al logo

vrml_image

Icono a visualizar en el mapa 3D

statusmap_image

Icono a visualizar en el mapa 2D

Se deberá crear en el archivo hostextinfo.cfg dentro subdirectorio correspondiente

al host con las entradas de configuración anteriormente explicadas. Lo iconos se encuentran dentro del directorio {nagiosdir}share/images/logos/ tanto en su version png como gd

Para convertir una imagen png comun a un icono gd2 (necesario para la generacion del grafico de statusmap 2D) debemos ejecutar el siguiente comando:

```
pngtogd2 mi-icono.png mi-icono.gd2  1  1
```

El primer parametro es mi ya existente imagen png, el segundo parametro es el nombre de archivo de salida en formato gd2, el parametro 1 se refiere a que la cree en formato raw (crudo), y el segudo parametro es para que lo cree sin compresion, todo esto se realiza dentro del directorio logos anteriormente explciado.

Dependencia del estado de host por el estado de otro host

En casos donde hay host donde su estado depende del estado de otro host, es posible especificar dependencias. Aqui vemos una entrada tipica

```
define hostdependency{
        host_name                               linuxsap2
        dependent_host_name                     linuxoracle
        notification_failure_criteria           d
    }


define hostdependency{
        host_name                               {nombre del host a referise}
        dependent_host_name                     {nombre del host del cual depende}
        notification_failure_criteria           {opciones de notificacion}
    }
```

host_name

Nombre del host al cual corresponde la información dada

dependent_host_name

Nombre del host del cual depende

notification_failure_criteria

Opciones de notificacion en caso de cada estado

Se deberá crear en el archivo hostdependency.cfg dentro subdirectorio correspondiente al host con las entradas de configuración anteriormente explicadas.

Definiendo tiempos de chequeo

Dentro de Nagios se pueden establecer definiciones para controlar cuando diversos aspectos de la lógica de vigilancia y alerta sobre elementos monitoreador por ejemplo que tal servicio o host se chequee en ciertos intervalos de tiempo, por ejemplo si un host no debe ser chequeado los domingos o un servicio debe ser chequeado solo de lunes a viernes de 08:00 HS a 18:00 HS etc

A su ves las definiciones Timeperod pueden contener varios tipos de directivas, entre los días de semana, días del mes, y las fechas. El orden de precedencia de los distintos tipos de directivas (en orden descendente) es el siguiente:

- Fecha del calendario (2008-01-01)

- Mes específico fecha (1 de enero)
- Mes día genéricos (Día 15)
- Compensar los días de la semana de mes específico (2 º martes de diciembre)
- Compensar los días de la semana (3 º lunes)
- Normal del día de la semana (martes)

Ejemplos de las diferentes directivas timeperiod que podemos implementar :

Chequeos las 24 horas los 7 dias 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
        }
```

Chequeos en las horas laborales lunes a viernes de 9 a 17

```
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
        }
```

Chequeos sin tiempos, o sea no chequea en ningún momento

```
# 'none' timeperiod definition
define timeperiod{
        timeperiod_name none
        alias           No Time Is A Good Time
        }
```

En ciertos dias feriados excluir el chequeo servicios, ej Navidad, Año nuevo etc

```
define timeperiod{
        name                    us-holidays
        timeperiod_name         us-holidays
        alias                   U.S. Holidays
        january 1               00:00-00:00     ; New Years
        monday -1 may           00:00-00:00     ; Memorial Day (last Monday in May)
        july 4                  00:00-00:00     ; Independence Day
        monday 1 september      00:00-00:00     ; Labor Day (first Monday in
September)
        thursday -1 november    00:00-00:00     ; Thanksgiving (last Thursday in
November)
        december 25             00:00-00:00     ; Christmas
        }
```

Definimos un periodo de tiempo que chequee las 24 horas del dia los 7 dias de la semana, pero que incluya las excepciones anteriormente mostradas

```
define timeperiod{
        timeperiod_name 24x7_sans_holidays
        alias           24x7 Sans Holidays
        use             us-holidays             ; Agregar excepciones
        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
        }
```

Otra manera de evitar tiempos de chequeo es con la directiva exclude

Como ejemplo podemos definir 3 timeperiods

```
define timeperiod{
        name                    weekdays
        timeperiod_name         weekdays
        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
        }
define timeperiod{
        name                    weekends
        timeperiod_name         weekends
        saturday                00:00-24:00
        sunday                  00:00-24:00
      }
define timeperiod{
        name                    holidays
        timeperiod_name         holidays
        january 1               00:00-24:00    ; New Year's Day
        2008-03-23              00:00-24:00    ; Easter (2008)
        2009-04-12              00:00-24:00    ; Easter (2009)
        monday -1 may           00:00-24:00    ; Memorial Day (Last Monday in May)
        july 4                  00:00-24:00    ; Independence Day
        monday 1 september      00:00-24:00    ; Labor Day (1st Monday in September)
        thursday 4 november     00:00-24:00    ; Thanksgiving (4th Thursday in
November)
        december 25             00:00-24:00    ; Christmas
        december 31             17:00-24:00    ; New Year's Eve (5pm onwards)
      }
```

Ahora definimos un timeperiod llamadas por ejemplo, incluya los dias de la semana, pero excluya los dias festivos

```
define timeperiod{
      timeperiod_name         llamadas
      use               weekdays    ; Include weekdays
      exclude                 holidays    ; Exclude holiday dates/times defined
elsewhere
      }
```

Alternando dias, o sea desde el primero de agosto de 2007 cada dos dias notificar, si en vez de / 2 ponemos / 14 lo realizara cada 14 dias

```
define timeperiod{
      timeperiod_name         john-oncall
      2007-08-01 / 2          00:00-24:00 ; Every two days, starting August 1st, 2007
```

```
      }
```

En la entrada del contacto deberemos especificarle los timeperiods para hosts y servicios

```
define contact{
      contact_name                  john
      ...
      host_notification_period      john-oncall
      service_notification_period   john-oncall
      }
```

Manejadores de Eventos - Event Handler

Cuando hablamos de **event handler** o **manejador de eventos**, nos referimos a funciones que responden a eventos que se producen, como pueden ser un cambio de estado.

Programando plugins

Desarrollar plugins de chequeos para Nagios es extremadamente flexible, ya que no dependemos del lenguaje de programación debido a que Nagios toma la salida resultante de su ejecución.

Deberemos conocer bien lo que queremos chequear y conocer los indicadores que nos mostraran si deberemos expresarlos como un OK, un WARNING o un CRITICAL.

Luego deberemos reflejar esos estados en su código de retorno o Exit status, dependiendo del código del mismo Nagios entenderá que debe mostrar.

Exit status	Estado de Servicio	Estado de Host	Descripcion
0	OK	UP	El plugin es capaz de verificar el servicio y que parece estar funcionando correctamente
1	WARNING	UP/DOWN/UNREACHABLE	El plugin es capaz de verificar el servicio, pero que parecía estar por encima de un umbral de “advertencia” o parece no estar funcionando correctamente
2	CRITICAL	DOWN/UNREACHABLE	El plugin detecta que o bien el servicio no funciona o que está por encima de un umbral “crítico”
3	UNKNOWN	DOWN/UNREACHAB	Argumentos de línea

		LE	de comandos no válida o fallas internas del plugin (por ejemplo error en un socket o dns) que le impiden realizar las operaciones especificadas

Añadir nuestro plugin

Para añadir nuestro plugin a Nagios

- Deberemos copiar nuestro ejecutable (ya sea script o binario) al directorio */usr/local/nagios/libexec* con permisos de ejecución.
- Crearle una entrada dentro del archivo de configuración de Comandos.
- Crear un Servicio y asignarle ese comando para que se encargue de dicho chequeo.

Notas sobre el desarrollo de plugins

- No ejecutar comandos del sistema, sin especificar su ruta de acceso completa.
  - Utilice spopen () si deben ser ejecutados comando externos en su plugin en C.
  - Ver los principales ejemplos de plugin para ver cómo se hace esto.
- No haga archivos temporales a menos que sea absolutamente necesario.
  - Si se necesitan archivos temporales, asegúrese de que el plugin y elimine el archivo temporal, cuando finalice.
- No se deje engañar por enlaces simbólicos
  - Si el plug-in abre todos los archivos, tomar medidas para asegurarse de que no están siguiendo un enlace simbólico a otro lugar en el sistema.
- Validar todas las entradas
- Usar rutinas en utils.c utils.pm o escribir más según sea necesario

Referencias sobre el desarrollo de plugins

http://nagios.sourceforge.net/docs/3_0/pluginapi.html

http://nagiosplug.sourceforge.net/developer-guidelines.html

Ejemplo de desarrollo paso a paso

Plugin consultando SNMP

Dado el caso que queramos obtener información interna de determinado Host y necesitemos que este disponible para consultarla por SNMP, podemos incluir dicha informacion. Para ellos deberemos incluir diferentes directivas de configuracion en el archivo *snmpd.conf*, y haremos uso de la tabla externa del objeto UCD. La *Tabla externa UCD* es una tabla extensible de comandos de la cual obtendremos su código de resultado de ejecución y su salida.

/etc/snmp/snmpd.conf

```
syslocation Unknown (configure /etc/snmp/snmpd.local.conf)
syscontact Root <root@localhost> (configure /etc/snmp/snmpd.local.conf)
```

```
exec comando1 /bin/comando1
exec comando2 /bin/comando2
exec comando3 /bin/comando3
exec comando3 /bin/comando4

rouser consultorsnmp auth
#createUser consultorsnmp MD5 consultorsnmp123
```

Luego podemos ver los resultados obtenidos realizando una consulta SNMP a *UCD-SNMP-MIB::extTable* o *.1.3.6.1.4.1.2021.8*, obteniendo resultados similares a :

snmpwalk -v 3 -l authNoPriv -a MD5 -u consultorsnmp -A consultorsnmp123 127.0.0.1 UCD-SNMP-MIB::extTable*

```
UCD-SNMP-MIB::extIndex.1 = INTEGER: 1
UCD-SNMP-MIB::extIndex.2 = INTEGER: 2
UCD-SNMP-MIB::extIndex.3 = INTEGER: 3
UCD-SNMP-MIB::extIndex.4 = INTEGER: 4
UCD-SNMP-MIB::extNames.1 = STRING: comando1
UCD-SNMP-MIB::extNames.2 = STRING: comando2
UCD-SNMP-MIB::extNames.3 = STRING: comando3
UCD-SNMP-MIB::extNames.4 = STRING: comando4
UCD-SNMP-MIB::extCommand.1 = STRING: /bin/comando1
UCD-SNMP-MIB::extCommand.2 = STRING: /bin/comando2
UCD-SNMP-MIB::extCommand.3 = STRING: /bin/comando3
UCD-SNMP-MIB::extCommand.4 = STRING: /bin/comando4
UCD-SNMP-MIB::extResult.1 = INTEGER: 0
UCD-SNMP-MIB::extResult.2 = INTEGER: 0
UCD-SNMP-MIB::extResult.3 = INTEGER: 0
UCD-SNMP-MIB::extResult.4 = INTEGER: 0
UCD-SNMP-MIB::extOutput.1 = STRING: salida-comando
UCD-SNMP-MIB::extOutput.2 = STRING: salida-comando
UCD-SNMP-MIB::extOutput.3 = STRING: salida-comando
UCD-SNMP-MIB::extOutput.4 = STRING: salida-comando
UCD-SNMP-MIB::extErrFix.1 = INTEGER: 0
UCD-SNMP-MIB::extErrFix.2 = INTEGER: 0
UCD-SNMP-MIB::extErrFix.3 = INTEGER: 0
UCD-SNMP-MIB::extErrFix.4 = INTEGER: 0
UCD-SNMP-MIB::extErrFixCmd.1 = STRING:
UCD-SNMP-MIB::extErrFixCmd.2 = STRING:
UCD-SNMP-MIB::extErrFixCmd.3 = STRING:
UCD-SNMP-MIB::extErrFixCmd.4 = STRING:
```

Ramas SNMP :

- UCD-SNMP-MIB::extNames
  - En esta rama obtendremos el nombre que le hemos asignado a dicho comando
- UCD-SNMP-MIB::extCommand
  - Esta rama nos devolvera la ruta completa al ejecutable del comando
- UCD-SNMP-MIB::extResult
  - Nos devolvera el resultado de la ejecucion del comando fue exitosa o no, devolviendonos su *exit status*
- UCD-SNMP-MIB::extOutput
  - Aqui obtendremos la salida del comando con el string o expresion que necesitemos conocer

Referencias sobre consultas SNMP a la tabla UCD

http://net-snmp.sourceforge.net/docs/mibs/ucdavis.html

Ejemplo de consulta SNMP con Perl por medio de Net::SNMP

Ejemplo simple de como consultar una variable SNMPv1.

```
#!/usr/local/bin/perl
use strict;
use warnings;
use Net::SNMP;

my $OID_sysUpTime = '1.3.6.1.2.1.1.3.0';

my ($session, $error) = Net::SNMP->session(
    -hostname  => shift || '127.0.0.1',
    -community => shift || 'public',
);

if (!defined $session) {
    printf "ERROR: %s.\n", $error;
    exit 1;
}

my $result = $session->get_request(-varbindlist => [ $OID_sysUpTime ],);

if (!defined $result) {
    printf "ERROR: %s.\n", $session->error();
    $session->close();
    exit 1;
}

printf "The sysUpTime for host '%s' is %s.\n",
$session->hostname(), $result->{$OID_sysUpTime};

$session->close();

exit 0;
```

Ejemplo simple de como consultar una variable SNMPv3, igualmente se pueden consultar mas de una variable por sesion. En este caso estamos consultado la variable extOutput.1 para ver la salida del primer comando.

```
#!/usr/bin/perl
use Net::SNMP;
$oid = ".1.3.6.1.4.1.2021.8.1.101.1";

$snmpv3_username = "consultorsnmp";     # SNMPv3 username
$snmpv3_password = "consultorsnmp123"; # SNMPv3 password
$snmpv3_authprotocol = "md5";          # SNMPv3 hash algorithm (md5 / sha)
$snmpv3_privpassword = "";             # SNMPv3 hash algorithm (md5 / sha)
$snmpv3_privprotocol = "des";          # SNMPv3 encryption protocol (des / aes /
aes128)
$version = "3";
$timeout = "2";
$hostname = "127.0.0.1";
# Crear la sesion SNMP
        ($s, $e) = Net::SNMP->session(
                -username       =>  $snmpv3_username,
                -authpassword   =>  $snmpv3_password,
                -authprotocol   =>  $snmpv3_authprotocol,
                -hostname       =>  $hostname,
                -version        =>  $version,
```

```
                -timeout        =>  $timeout,
        );
        if ($s){
        } else {
            print "CRITICAL - El agente no responde, SNMP v3 ($e)\n";
            exit(2);
        }
    $s->get_request($oid);
        foreach ($s->var_bind_names()) {
            $oid_consulta   = $s->var_bind_list()->{$_};
        }
$s->close();
print $oid_consulta;
```

Ejemplo de consulta de un puerto TCP con Perl por medio de IO::Socket::INET

check_port.pl $HOSTADDRESS$ $ARG1$

```
#!/usr/bin/perl

use IO::Socket::INET;
my $IP = $ARGV[0];
my $PORT = $ARGV[1];
my $STATUS = "0";
my $sock = new IO::Socket::INET->new(
          PeerAddr => $IP,
          PeerPort => $PORT,
          Proto => 'tcp',
          );
          $sock or $STATUS = "2";

if ( $STATUS eq 0 ) {
   print "El puerto $ARGV[1] esta OK";
   exit 0;
} else {
   print "No puedo abrir el puerto $ARGV[1] WARNING";
   exit 2;
   die;
}
```

Ejemplo de consultas MySQL

Podemos realizar plugins que dependan de la salida de otros plugins sin necesidad de consultar o ejecutar nuevamente estos, uilizando los datos contenidos en la base de datos MySQL

Por ejemplo si queremos consultar la lista de miembros de un grupo de hosts

```
    SELECT nagios_hosts.display_name
    FROM
    nagios_hostgroups,nagios_instances,
    nagios_hosts,nagios_hostgroup_members,
    nagios_objects
    WHERE
    nagios_hostgroups.hostgroup_id=nagios_hostgroup_members.hostgroup_id
    AND
    nagios_hostgroup_members.host_object_id=nagios_hosts.host_object_id
    AND
    nagios_hostgroups.hostgroup_object_id=nagios_objects.object_id
    AND
    nagios_objects.name1 = '{grupo_de_host_a_consultar}'
    ORDER BY nagios_hosts.display_name ASC
```

Para conocer el estado de un servicio de un determinado host

```
    SELECT
    nagios_servicestatus.perfdata,nagios_servicestatus.output
    FROM
    nagios_services,nagios_servicestatus,
    nagios_objects,nagios_instances
    WHERE
    nagios_services.instance_id=nagios_instances.instance_id
    AND
    nagios_servicestatus.service_object_id=nagios_services.service_object_id
    AND
    nagios_servicestatus.service_object_id=nagios_objects.object_id
    AND
    nagios_objects.name2 = '{servicio_a_consultar}'
    AND
    nagios_services.config_type='1'
    AND
    nagios_objects.name1 = '{host_que_lo_contiene}'
```

Para conocer el estado de un host

```
    SELECT
    nagios_hosts.display_name,
    nagios_hosts.alias,
    nagios_hoststatus.current_state,
    nagios_hoststatus.output
    FROM
    nagios_hoststatus,
    nagios_instances,
    nagios_hosts,
    nagios_objects
    WHERE
    nagios_hoststatus.host_object_id=nagios_objects.object_id
    AND
    nagios_hoststatus.host_object_id=nagios_hosts.host_object_id
    AND
    nagios_hosts.instance_id=nagios_instances.instance_id
    AND
    nagios_hosts.config_type='1'
    AND
    nagios_hosts.display_name = '{host_a_consultar}'
```

Para consultar los servicios pertenecientes a un determinado grupo de servicios

```
    SELECT
    obj2.name1 AS host_name,
    obj2.name2 AS service_description
    FROM nagios_servicegroups
    INNER JOIN nagios_servicegroup_members ON
    nagios_servicegroups.servicegroup_id=nagios_servicegroup_members.servicegroup_id
    INNER JOIN nagios_services ON
    nagios_servicegroup_members.service_object_id=nagios_services.service_object_id
    INNER JOIN nagios_objects AS obj1 ON
    nagios_servicegroups.servicegroup_object_id=obj1.object_id
    INNER JOIN nagios_objects AS obj2 ON
    nagios_servicegroup_members.service_object_id=obj2.object_id
    INNER JOIN nagios_instances ON
    nagios_servicegroups.instance_id=nagios_instances.instance_id
    WHERE nagios_servicegroups.alias ='{alias_service_group}'
    AND nagios_services.display_name ='{alias_de_servicio}'
```

Para conocer los estados de los hosts

```
SELECT nagios_hosts.display_name,nagios_hoststatus.current_state
FROM   nagios_hoststatus,nagios_hosts
```

```
WHERE  nagios_hoststatus.host_object_id = nagios_hosts.host_object_id;
```

Resultando en una tabla como esta

display_name	current_state
firewall	0
apache	1
servidor-sql	0
sap-db	0

Para conocer el estado de un servicio, independientemente de los hosts en los que se encuentre disponible

En este caso el servicio **HTTP**

```
SELECT obj1.name1 AS host_name,
nagios_services.service_object_id,
obj1.name2 AS service_description,
nagios_servicestatus.current_state
FROM `nagios_servicestatus`
LEFT JOIN nagios_objects AS obj1 ON nagios_servicestatus.service_object_id =
obj1.object_id
LEFT JOIN nagios_services ON nagios_servicestatus.service_object_id =
nagios_services.service_object_id
LEFT JOIN nagios_instances ON nagios_services.instance_id =
nagios_instances.instance_id
WHERE nagios_services.config_type = '1'
AND obj1.name2 = 'HTTP'
```

Resultando en una tabla como esta

host_name	service_object_id	service_description	current_state
concentrador_mpls	18	HTTP	0
router-vpn	23	HTTP	1
central	28	HTTP	2
router-45	33	HTTP	0

Consulta simple para obtener un pantallazo general del estado de servicios de un hostgroup

```
SELECT
   CASE st.current_state
       WHEN 0 THEN 'OK'
       WHEN 1 THEN 'WARNING'
       WHEN 2 THEN 'CRITICAL'
       WHEN 3 THEN 'UNKNOWN'
    END AS states,
    COUNT(st.current_state) AS NUMBER
FROM nagios_hostgroup_members m, nagios_services s,
    nagios_servicestatus st
    WHERE m.host_object_id = s.host_object_id
    AND s.service_object_id = st.service_object_id
    AND hostgroup_id IN
        (
            SELECT hostgroup_id
            FROM nagios_hostgroups hg, nagios_objects o
            WHERE hg.hostgroup_object_id = o.object_id
                  AND o.name1 = '{nombre_hostgroup}'
        )
GROUP BY st.current_state;
```

Resultando en una tabla como esta

states	number
OK	1754
WARNING	13
CRITICAL	84
UNKNOWN	20

Consulta simple para obtener un conteo general de los estados generales ordenados por meses de un año determinado.

```
SELECT
      CONCAT('',DATE_FORMAT(nagios_statehistory.state_time,'%M')) AS TEMPS,
      ELT(nagios_statehistory.state+1,'OK','WARNING','CRITICAL','UNKNOWN') AS ETAT,
      CONCAT('',COUNT(nagios_statehistory.state)) AS QUANTITE
FROM
      `nagios_statehistory`
      LEFT JOIN nagios_objects AS obj1 ON
nagios_statehistory.object_id=obj1.object_id
      LEFT JOIN nagios_instances ON
nagios_statehistory.instance_id=nagios_instances.instance_id
WHERE
      obj1.objecttype_id='2'
      AND nagios_statehistory.state_type = 1
      AND DATE_FORMAT(nagios_statehistory.state_time,'%Y') = 2010
GROUP BY
      TEMPS, ETAT
ORDER BY
      DATE_FORMAT(nagios_statehistory.state_time,'%m'),nagios_statehistory.state
```

Resultando en una tabla como esta

TEMPS	ETAT	QUANTITE
November	OK	1754
November	WARNING	13
November	CRITICAL	84
November	UNKNOWN	20

Trigger para guardar el último estado de los objetos

Nagios no guarda en ninguna tabla de MySQL el último estado de los objetos, pero guarda los cambios de estado en la tabla nagios_statehistory. Para mantener el último estado se puede usar un trigger:

```
CREATE TABLE last_state (object_id INT(11), state INT(6));

CREATE TRIGGER last_state after INSERT ON nagios_statehistory
FOR each ROW
BEGIN
  IF NEW.state_change = 1 THEN
    UPDATE last_state SET state = NEW.state WHERE object_id NEW.object_id;
  END IF;
END;//
```

Ese trigger hace un update así que hay que cronear esto para mantener al día con los objetos la tabla.

```
INSERT INTO last_state
        SELECT obj.object_id, 0
        FROM nagios.nagios_objects AS obj
        WHERE obj.object_id NOT IN
                (SELECT object_id FROM last_state);
```

Luego estas consultas podemos integrarlas en PHP o Perl, y obtener resultados que necesitemos para reflejarlos en un visor web, o en los resultados de un plugin.

NOTA: Importante el modulo mysql de PHP no soporta la consulta a los PROCEDURE de MySQL, para ello deberemos utilizar el modulo mysqli de PHP, otra cosa tambien importante es que mysqli soporta conecciones persistentes, asi que para evitarse tiempo de implementacion a veces es necesario o implementar un script en Perl, o hacer que el resultado del PROCEDURE se almacene en una tabla temporal y realizar una consulta simple desde ahi.

Correlación de eventos

Si bien Nagios provee de monitoreo en vivo para saber el estado de nuestros servicios, a veces necesitamos un proceso para analizar los datos de eventos e identificar patrones, que nos ayudaran a entender causas comunes y causas iniciales, por medios de reglas para estados predefinidos.

Para poder determinar estos casos deberemos implementar una Bitácora centralizada de eventos del sistema (SYSLOG).

Sistema de loggueo centralizado

Se tiene la necesidad de un sistema que reciba y almacenes archivos de registros determinados de n* servidores. Que se disponga de una interfaz de acceso a esos datos, para la organizacion y procesamiento de los mismos, con lo cual se podran detectar eventos, resolver situaciones y emitir distintos tipos de alarmas.

Syslog es un estándar de facto para el envío de mensajes de registro en una red informática IP. Por syslog se conoce tanto al protocolo de red como a la aplicación o biblioteca que envía los mensajes de registro.

Un mensaje de registro suele tener información sobre la seguridad del sistema, aunque puede contener cualquier información. Junto con cada mensaje se incluye la fecha y hora del envío.

Es útil registrar, por ejemplo:

- Un intento de acceso con contraseña equivocada
- Un acceso correcto al sistema
- Anomalías: variaciones en el funcionamiento normal del sistema
- Alertas cuando ocurre alguna condición especial
- Información sobre las actividades del sistema operativo
- Errores del hardware o el software

También es posible registrar el funcionamiento normal de los programas; por ejemplo, guardar cada acceso que se hace a un servidor web, aunque esto suele estar separado del resto de alertas.

El protocolo syslog es muy sencillo: existe un ordenador servidor ejecutando el servidor de syslog, conocido como syslogd (demonio de syslog). El cliente envía un pequeño mensaje de texto (de menos de 1024 bytes).

Los mensajes de syslog se suelen enviar vía UDP, por el puerto 514, en formato de texto plano. Algunas implementaciones del servidor, como syslog-ng, permiten usar TCP en vez de UDP, y también ofrecen Stunnel para que los datos viajen cifrados mediante SSL/TLS.

Generalidades de un sistema de log.

Log es el registro de acciones y eventos que tienen lugar en un sistema.

Los logs son el primer registro de la actividad en los sistemas y redes.

Los logs de un sistema son una parte primaria de la seguridad y pueden ser usados en la detección de ataques e intrusiones, así como en el análisis de fallas de hardware y software.

El programa syslog, es una interface que provee un framework standard para que tanto programas como el mismo sistema operativo puedan generar mensajes que peuden ser almacenados localmente o ser enviados a un host remoto. Originalmente escrito para Unix, se convirtió en un standard que se usa en muchos sistemas operativos y dispositivos de red.

¿ Cual es la utilidad de un sistema de syslog centralizado ?

En un sistema de syslog centralizado, un server común recibe todos los mensajes de syslog de todos los sistemas de la red. Esto incluye logs de los servers Unix/linux/Windows etc, firewalls, y dispositivos de red (routers, switches, etc)

Hay varias ventajas de un sistema de syslog centralizado

- El syslog puede ser conectado en un segmento de red diferente protegido por un firewall

para mantener más segura la información

- Teniendo los mensajes de todos los equipos, puede hacerse una correlación de ataques o

fallas en distintos puntos de una manera mucho más sencilla. Por ejemplo si en el syslog aparece un mensaje de desconexión de la interface de red de varios servers en el mismo momento, es lógico suponer una falla en el switch donde estos servers estan conectados.

- Un usuario no deseado que haya ingresado en un server, no podrá alterar los mensajes que

se hayan almacenado en el server central.

- Se pueden generar alertas usando sistemas de monitoreo de logs.

Sistema de monitoreo de logs

El análisis de logs es una herramienta muy importante para mantener el control de servers y dispositivos de red.

Sin embargo esta es una de las tareas que más tiempo consume y por consiguiente que menos se hace.

Con la cantidad de mensajes informativos que se generan en un sistema de log, detectar en forma manual los mensajes de problemas es muy dificultoso y con mucha probabilidad de error. Esto se vé aumentado cuando se usa un sistema de syslog centralizado, donde la información proviene de varias fuentes distintas. Muchas soluciones de monitoreo se basan en sumarizar la información de archivos de log de dias previos.

Esto es muy útil para la generación de estadísticas y análisis posterior a una falla o intrusión, pero no tanto para la resolución de problemas.

Un administrador no puede actuar en forma proactiva, previamente a que el error ocurra.

Muchas fallas o accesos no autorizados se ven precedidos por mensajes que de haber sido detectados, podrían haber permitido tomar acciones preventivas.

Por ejemplo, un acceso no autorizado via ssh, puede haber estado precedido por una gran cantidad de intentos fallidos de acceso.

Disponer una solución online de monitoreo, permite disponer de herramientas que pueden ayudar a prevenir problemas graves antes que ocurran.

Detectar eventos en el momento que ocurren permite generar acciones en ese mismo instante y no luego de las consecuencias.

Siguiendo con el ejemplo del acceso ssh, podría bloquearse el acceso ssh desde determinada dirección IP despues de un número de intentos fallidos de acceso.

Un concepto que aparece aquí es el de correlación de eventos.

Un sistema automatizado de análisis de logs que pueda hacer una correlación de varios eventos simplifica y acelera el monitoreo de eventos consolidando alertas y mensajes de error en un mensaje más corto y fácil de entender.

Una serie de operaciones están relacionadas con la correlación de eventos.

Compresión toma varias apariciones del mismo evento y se examina la duplicación de información, se remueve las redundacias y se reporta como un único evento. De esta manera 1000 mensajes “route failed” se convierte en un único alerta que dice “route failed 1000 times” Recuento reporta un número específico de eventos similares como uno solo. Esto se diferencia de la compresión en que no solo cuenta en que sea el mismo evento, sino que se supere un determinado umbral para generar un reporte.

Supresión asocia prioridades con alarmas y permite que el sistema suprima un alarma de prioridad más baja si ha ocurrido un evento de prioridad mayor.

Generalización asocia alarmas con eventos de más alto nivel que son los que son reportados.

Esto puede ser útil para por ejemplo para correlacionar eventos de múltiples discos de un array.

No se necesita ver cada mensaje específico si se puede determinar que el array completo tiene problemas.

Correlación basada en tiempo puede ser útil estableciendo causalidad. A menudo una información puede ser obtenida relacionando eventos que tienen una relación temporal específica.

Ejemplos genéricos:

- El Evento A está seguido del Evento B.
- Este es el primer Evento A desde el Evento B reciente.
- El Evento A sigue al Evento B dentro de los dos minutos.
- El Evento A no fue observado dentro del Intervalo I.

Objetivos

Simplificar y optimizar la administración de diferentes servicios para conocer su estado minuto a minuto, y elaborar planes de acción. A su vez el sistema debe ser simple de utilizar por el administrador, y ser posible de ver via web los registros del sistema, realizar busquedas etc.

Es útil registrar, por ejemplo:

- Un intento de acceso con contraseña equivocada
- Un acceso correcto al sistema
- Anomalías: variaciones en el funcionamiento normal del sistema
- Alertas cuando ocurre alguna condición especial
- Información sobre las actividades del sistema operativo
- Errores del hardware o el software

También es posible registrar el funcionamiento normal de los programas; por ejemplo, guardar cada acceso que se hace a un servidor web, aunque esto suele estar separado del resto de alertas.

Puede logguear tanto por UDP como por TCP, teniendo compatibilidad con el viejo syslog, soportando a su vez muchas mas opciones y tareas que el syslog comun.

Sobre Syslog-NG

Syslog-NG es un sistema para el envío de mensajes de registro en una red.

Es útil registrar, por ejemplo:

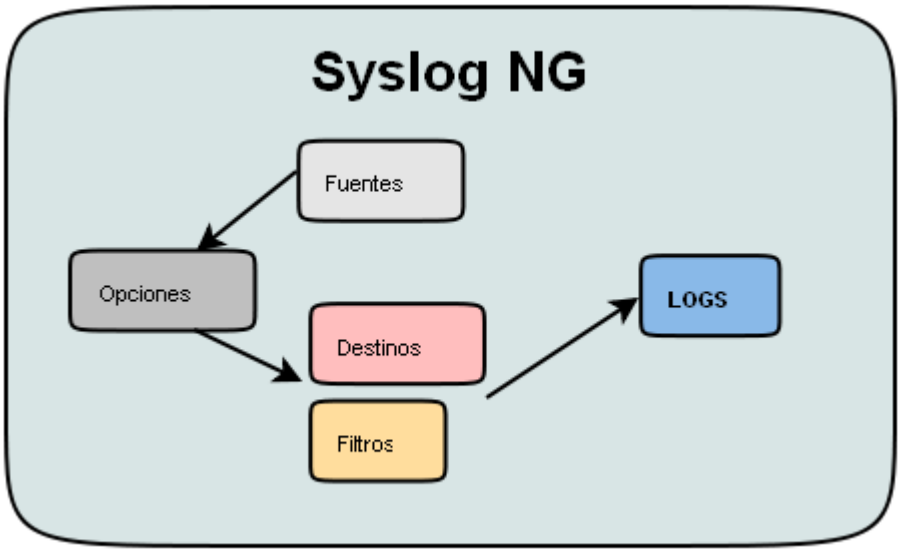
- Un intento de acceso con contraseña equivocada
- Un acceso correcto al sistema
- Anomalías: variaciones en el funcionamiento normal del sistema
- Alertas cuando ocurre alguna condición especial
- Información sobre las actividades del sistema operativo
- Errores del hardware o el software

También es posible registrar el funcionamiento normal de los programas; por ejemplo, guardar cada acceso que se hace a un servidor web, aunque esto suele estar separado del resto de alertas.

Puede logguear tanto por UDP como por TCP, teniendo compatibilidad con el viejo syslog, soportando a su vez muchas mas opciones y tareas que el syslog comun.

Explicacion tecnica simple

Syslog-NG se compone en capas de funcionamiento



Osea tengo en primer lugar opciopnes generales, luego fuentes de donde obtener datos de los registros, ya sean locales, externos por red (udp, tcp, archivos de texto), y luego tenemos destinos y filtros configurados, uniendolos forman los registros de sistema

Tareas

Se instalo el software syslog-ng dentro de un servidor Debian.

```
apt-get install syslog-ng
```

En la instalacion como resultante se tiene el siguiente archivo de configuracion principal

/etc/syslog-ng/syslog-ng.conf

Clientes

En los clientes se procedió a configurar el syslog convencional para que loggueara por UDP hacia el servidor syslog.

syslog-ng.conf

```
destination syslog_server {
    tcp( "10.1.1.2" port(514) );
};

filter      f_local6       { facility(local6); };

filter f_auth {
    facility(auth, authpriv);
};
log {
    source(src);
    filter(f_auth);
    destination(syslog_server);
};

log {
    source(src);
    filter(f_local6);
```

```
    destination(syslog_server);
};
```

syslog.conf

```
auth.*;authpriv.*;auth.notice;auth.error;auth.info;authpriv.none;     @10.1.1.2
local6.*                @10.1.1.2
```

/etc/profile.local

Esto es para logguear todos los comandos que ejecutan los usuarios, funciona aunque ejecuten un sudo o un su sin perder el rastro del usuario original, siempre y cuando no modifiquen la variable de entorno.

```
export PROMPT_COMMAND='RETRN_VAL=$?;logger -p local6.debug "$(whoami) [$$]: $
(history 1 | sed "s/^[ ]*[0-9]\+[ ]*//" ) [$RETRN_VAL]"'
```

Configuración del Servidor

Debian GNU/Linux Etch

Archivo /etc/syslog-ng/syslog-ng.conf

Opciones generales, fuentes de donde obtener información, filtros y destinos paracada filtro

```
options {
      long_hostnames(off);
      # doesn't actually help on Solaris, log(3) truncates at 1024 chars
      log_msg_size(8192);
      # buffer just a little for performance
      # sync(1); <- Deprecated - use flush_lines() instead
      flush_lines(1);
      # memory is cheap, buffer messages unable to write (like to loghost)
      log_fifo_size(16384);
      # Hosts we don't want syslog from
      #bad_hostname("^(ctld.|cmd|tmd|last)$");
      # The time to wait before a dead connection is reestablished (seconds)
      time_reopen(10);
      #Use DNS so that our good names are used, not hostnames
      use_dns(no);
      dns_cache(yes);
      #Use the whole DNS name
      use_fqdn(no);
      keep_hostname(no);
      chain_hostnames(no);
      #Read permission for everyone
      perm(0644);
      # The default action of syslog-ng 1.6.0 is to log a STATS line
      # to the file every 10 minutes.  That's pretty ugly after a while.
      # Change it to every 12 hours so you get a nice daily update of
      # # how many messages syslog-ng missed (0).
      stats(43200000);
};

# Log Interno
source interno {
        # message generated by Syslog-NG
        internal();
        # standard Linux log source (this is the default place for the syslog()
        # function to send logs to)
        unix-stream("/dev/log");
```

```
        # messages from the kernel
        file("/proc/kmsg" log_prefix("kernel: "));
};

# Create destination to LogZilla
destination d_logzilla {
   program("/usr/local/nagios/syslog/scripts/db_insert.pl"
   template("$HOST\t$FACILITY\t$PRIORITY\t$LEVEL\t$TAG\t$YEAR-$MONTH-$DAY\t$HOUR:
$MIN:$SEC\t$PROGRAM\t$MSG\n")
   );
};
source externo {
        # use the following line if you want to receive remote UDP logging messages
        # (this is equivalent to the "-r" syslogd flag)
        # Ampliamos el limite de conexiones para recibir gran cantidad de datos de
LOGS externos
        udp(ip(0.0.0.0) port(514));
        tcp(ip(0.0.0.0) port(514)  max-connections(150) );
};

# Tell syslog-ng to log to our new destination
log {
    source(externo);
    destination(d_logzilla);
};

# Alertas de Nagios Eventdb

destination d_eventdb {
    pipe(
        "/usr/local/nagios/var/rw/syslog-ng.pipe",
        template("$HOST\t$SOURCEIP\t$PRI\t$YEAR-$MONTH-$DAY\t$HOUR:$MIN:
$SEC\t$PROGRAM\t$MSG\n")
        template_escape(no)
    );
};

log {
    source(externo);
    destination(d_eventdb);
};
```

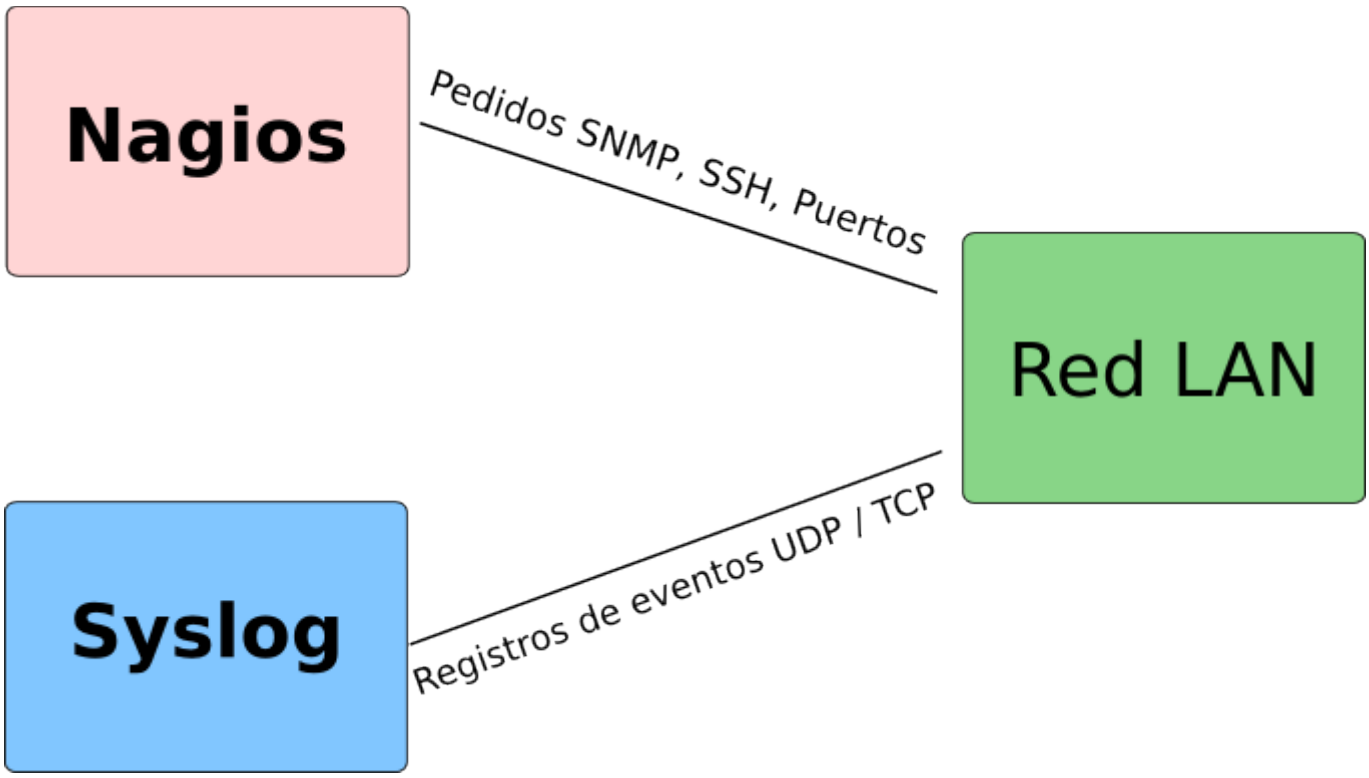
PHP-Syslog-ng

Para la visualizacion de los logs via web, busqueda y demas operaciones para su administracion se instalo, configuro y modifico a necesidad parte de codigo del sofware php-syslog-ng que provee de una interfaz web con soporte de busquedas y que servia para cubrir la necesidad planteada.

Una vez instalado se configuro el apache para su puesta en marcha de la siguiente manera (se omitieron las directivas de autenticacion).

```
Alias /syslog "/usr/local/php-syslog-ng/html"
<Directory "/usr/local/php-syslog-ng/html">
    Options All
    Order allow,deny
    Allow from all
    SSLRequireSSL
    AllowOverride None
</Directory>
```

Integración con Nagios



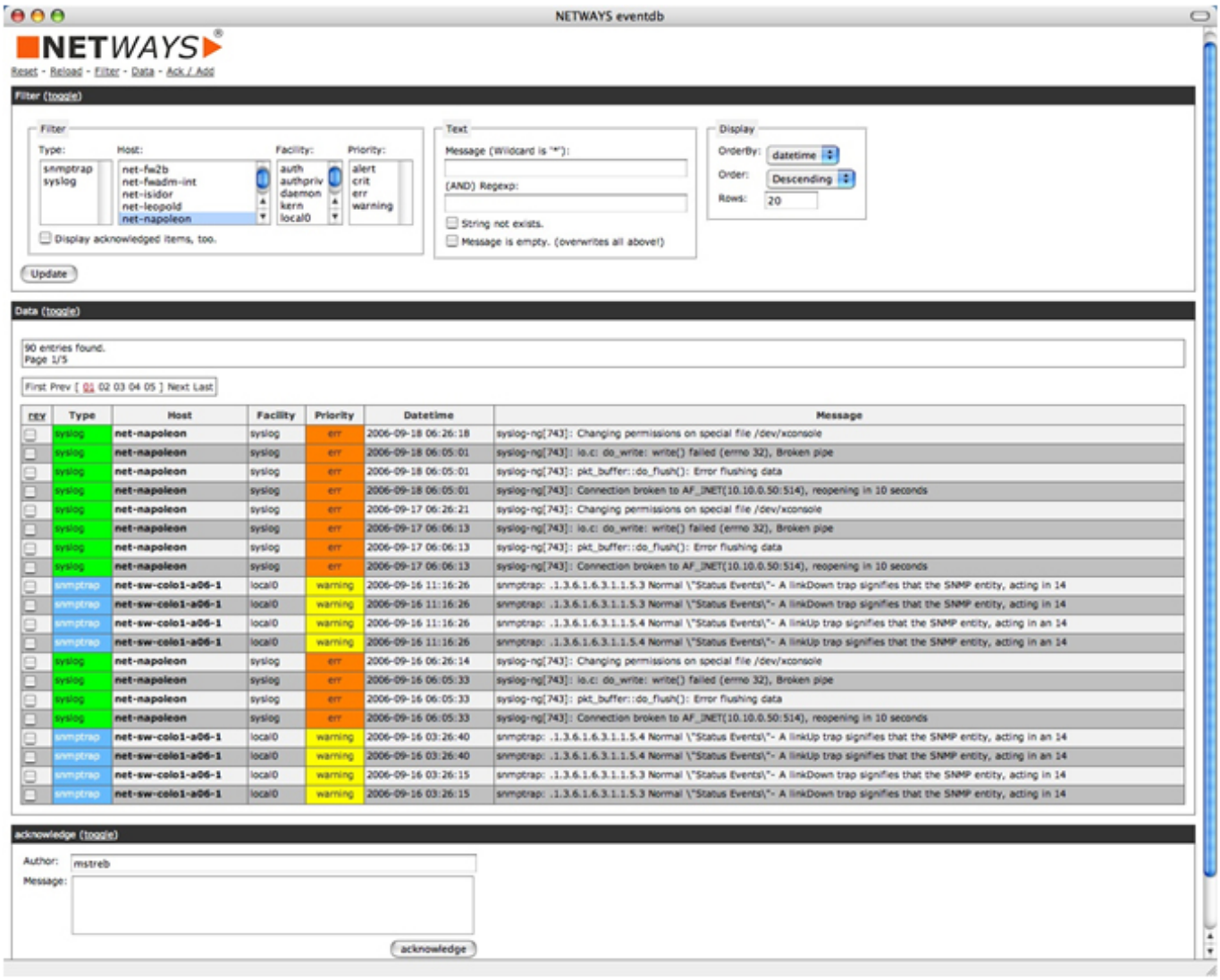
Eventdb

¿Que es Eventdb?

EventDB es una herramienta para facilitar el tratamiento de los datos basados en eventos de Syslog para herramientas de monitoreo como Nagios o Icinga y su integración con los mismos se logra a través de un plug-in de chequeo. Cuenta con varios plugins para diferentes fuentes de datos. La interfaz web permite a los usuarios buscar, filtrar y reconocer todos los eventos.

¿Cómo funciona?

El EventDB toma los eventos recibidos y los almacena en una base de datos MySQL. Se tomando los datos de syslog-ng desde un pequeño demonio perl (syslog-ng2mysql.pl). syslog-ng2mysql.pl abre una unix-pipe por un lado desde donde recibe los mensajes de syslog-ng, y utiliza DBI en el otro para escribir datos en MySQL.



Referencias

TIP´s varios

Sobre performance

- Compilar Nagios y NDO con optimizaciones para nuestro procesador

http://en.gentoo-wiki.com/wiki/Safe_Cflags

- Ver los tips de la documentacion oficial

http://nagios.sourceforge.net/docs/3_0/tuning.html

- Modificar opciones de buffer y cache de MySQL
- Establecer los valores que necesitemos de timeouts de Nagios
- Ubicar el archivo de estado de Nagios, sus archivos temporales en el sistema de archivos de memoria compartida
  - nagios.cfg : status_file=/dev/shm/status.dat
  - nagios.cfg : temp_file=/dev/shm/nagios.tmp
  - nagios.cfg : temp_path=/dev/shm

Nagios utiliza cada vez que necesita ejecutar un check dos forks en vez de uno,

esto le trae entre otros beneficios el aislamiento del proceso hijo. Sin embargo la utilización de CPU se incrementa necesariamente. Deshabilitando esta opción puede incrementarce la performance en instalaciones grandes.

Configuración por defecto:

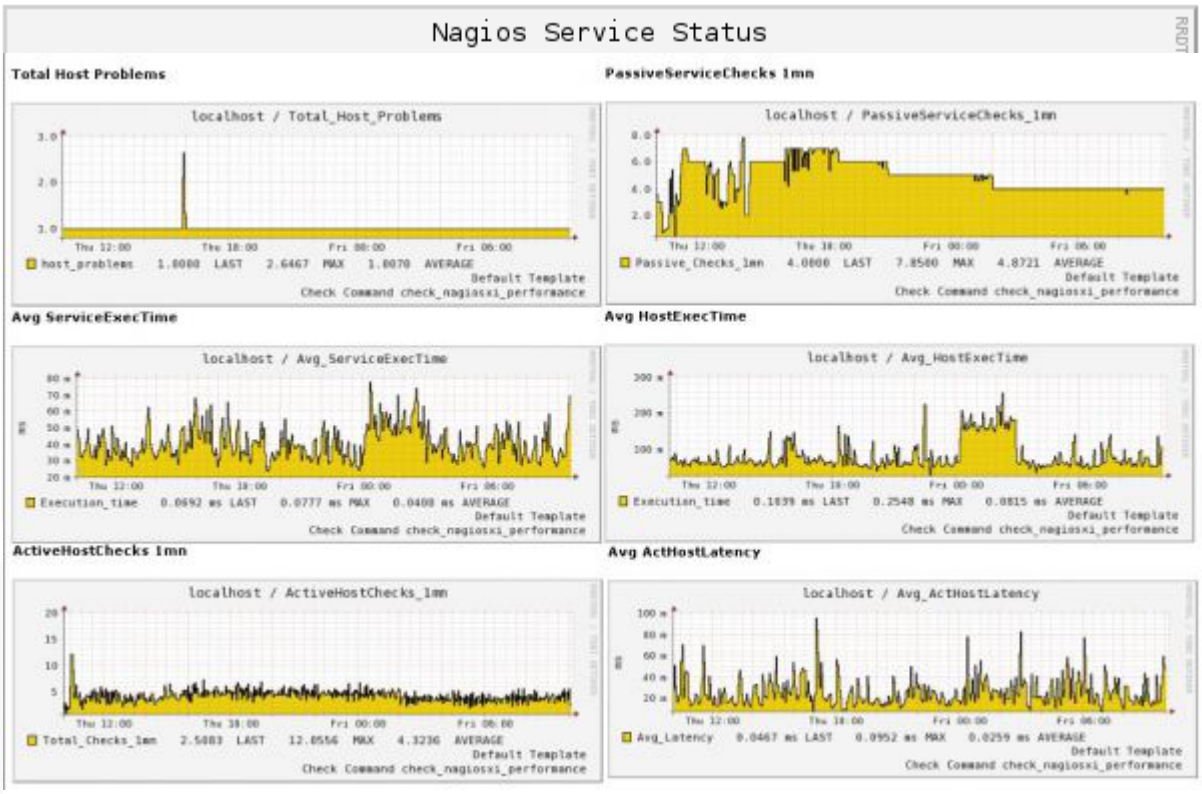
```
child_processes_fork_twice=1
```

Configuración para generar menos forks:

```
child_processes_fork_twice=0
```

Midiendo la performance de Nagios

La utilidad **nagiostats** permite obtener información variada sobre la ejecución de Nagios que puede ser muy útil para medir la performance de monitoreo. Puede obtener información, ya sea en formato legible en variables compatibles con RRD+PNP / MRTG. Esto nos sirve para informarnos efectivamente de las frecuencias de monitoreo y saber si estas son correctas y si la información obtenida es actual o tiene algún desfasaje.




```
Program Running Time: 0d 5h 20m 39s
Nagios PID: 10119

Used/High/Total Command Buffers:        0 / 0 / 64
Used/High/Total Check Result Buffers:   0 / 7 / 512

Total Services:                         95
Services Checked:                       94
Services Scheduled:                     91
Services Actively Checked:              94
Services Passively Checked:             1
Total Service State Change:             0.000 / 78.950 / 1.026 %
Active Service Latency:                 0.000 / 4.272 / 0.561 sec
Active Service Execution Time:          0.000 / 60.007 / 2.066 sec
Active Service State Change:            0.000 / 78.950 / 1.037 %
Active Services Last 1/5/15/60 min:     4 / 68 / 91 / 91
Passive Service State Change:           0.000 / 0.000 / 0.000 %
Passive Services Last 1/5/15/60 min:    0 / 0 / 0 / 0
Services Ok/Warn/Unk/Crit:              58 / 16 / 0 / 21
Services Flapping:                      1
Services In Downtime:                   0
```

```
Total Hosts:                            24
Hosts Checked:                          24
Hosts Scheduled:                        24
Hosts Actively Checked:                 24
Host Passively Checked:                 0

Total Host State Change:                0.000 / 9.210 / 0.384 %
Active Host Latency:                    0.000 / 0.446 / 0.219 sec
Active Host Execution Time:             1.019 / 10.034 / 2.764 sec
Active Host State Change:               0.000 / 9.210 / 0.384 %
Active Hosts Last 1/5/15/60 min:        5 / 22 / 24 / 24
Passive Host State Change:              0.000 / 0.000 / 0.000 %
Passive Hosts Last 1/5/15/60 min:       0 / 0 / 0 / 0
Hosts Up/Down/Unreach:                  18 / 4 / 2
Hosts Flapping:                         0
Hosts In Downtime:                      0

Active Host Checks Last 1/5/15 min:     9 / 52 / 164
   Scheduled:                           4 / 23 / 75
   On-demand:                           3 / 23 / 69
   Cached:                              2 / 6 / 20
Passive Host Checks Last 1/5/15 min:    0 / 0 / 0
Active Service Checks Last 1/5/15 min:  9 / 80 / 244
   Scheduled:                           9 / 80 / 244
   On-demand:                           0 / 0 / 0
   Cached:                              0 / 0 / 0
Passive Service Checks Last 1/5/15 min: 0 / 0 / 0
External Commands Last 1/5/15 min:      0 / 0 / 0
```

Flap Detection

Flap detection es un mecanismo mediante el cual Nagios induce que un servicio esta cambiando de estado continuamente sin entrar en un régimen permanente. El procedimiento del que se vale flap detection para detectar que un servicio o host esta en ese estado (“flapping”), es tomar los últimos 21 resultados de los checks y verificar cuantos cambios de estado hay en los mismos. Analizando más en detalle, dentro de 21 resultados hay 20 posibles cambios de estado. Suponiendo que dentro de estos 20 cambios de estado posibles, hubo 10 cambios de estado reales, la relación daría como resultado 10/20=0,50, es decir, 50%. Mediante dos parámetros se configuran los umbrales para que el mecanismo considere que el servicio o host se encuentra en ese estado.

Es interesante entender que el concepto de “flapping” se puede asignar por servicio ya sea en el template utilizado o en la definición de un servicio en particular. Igualmente es conveniente dejar la configuración general en su estado original y solo alterar la configuración particular del servicio que presenta problemas de esta índole.

Otra consideración importante es que se pueden quitar estados a interpretarse como cambio de estado, por ejemplo el estado “UNKNOWN” podría considerarse como que no hubo cambio respecto del estado anterior. Para ello hay que utilizar la directiva flap_detection_options dentro de la configuración del servicio o host.

Sobre los flags de compilacion

Ejemplo para un Pentium Dual E2160 o un Intel Quad Core

```
CHOST="i686-pc-linux-gnu"
CFLAGS="-march=prescott -O2 -pipe -fomit-frame-pointer"
CXXFLAGS="${CFLAGS}"
```

Los incluimos de la siguiente manera en el proceso de compilacion de Nagios

```
CHOST="i686-pc-linux-gnu" CFLAGS="-march=prescott -O2 -pipe -fomit-frame-pointer"
CXXFLAGS="${CFLAGS}" ./configure
```

La misma configuracion es valida para compilar MySQL (ej 5.1.37)

```
CHOST="i686-pc-linux-gnu" CFLAGS="-march=prescott -O2 -pipe -fomit-frame-pointer"
CXXFLAGS="${CFLAGS}" ./configure --with-big-tables --with-
plugins=partition,myisam,innobase
```

NDO

Cuando la base de datos esta cargada con una gran cantidad de registros, el NDO comienza a tener un comportamiento erratico, y los datos reflejados no son los correctos, por lo tanto cada cierto tiempo hay que purgar algunas tablas, ese tiempo sera dependiendo de la cantidad de objetos a monitorear.

Podemos crear un script que realize dicha tarea

```
#!/bin/bash
echo "TRUNCATE TABLE nagios_servicechecks" | /usr/bin/mysql -u root --batch
--database=nagios
echo "TRUNCATE TABLE nagios_logentries" | /usr/bin/mysql -u root --batch
--database=nagios
echo "TRUNCATE TABLE nagios_service_contactgroups" | /usr/bin/mysql -u root --batch
--database=nagios
echo "TRUNCATE TABLE nagios_hostchecks" | /usr/bin/mysql -u root --batch
--database=nagios
```

Para luego incluirlo en el **crontab**, por ejemplo si tenemos alrededor de mas de 600 hosts y alrededor de 2000 servicios o mas, chequeando ambos en intervalos de ente 1 y 5 minutos, podemos establecer su ejecucion en 15 minutos.

```
0,15,30,45 * * * root   /usr/local/nagios/sbin/truncar_db.sh
```

Igualmente en las opciones de configuracion del daemon ndo2db tenemos opciones referidas a esto, aunque igualmente dependeremos del script a realizar ya que el NDO en si mismo puede fallar. La configuracion se refiere a valores en minutos.

```
# Keep timed events for 24 hours
#max_timedevents_age=1440
max_timedevents_age=120
# Keep system commands for 1 week
#max_systemcommands_age=10080
max_systemcommands_age=1
# Keep service checks for 1 week
#max_servicechecks_age=10080
max_servicechecks_age=240
# Keep host checks for 1 week
#max_hostchecks_age=10080
max_hostchecks_age=240
# Keep event handlers for 31 days
#max_eventhandlers_age=44640
max_eventhandlers_age=60
```

Reparar tabla de NDO : En caso de que una tabla se corrompa, deberemos bajar el servicio NDO para que nagios no siga escribiendo registros y ejecutar la siguiente orden desde el shell.

```
mysqlcheck —-auto-repair nagios nagios_servicechecks;
```

Retoques al codigo fuente

Para que cuando elegimos exportar los informes de disponibiliad en formato CSV, no nos aparesca en la ventana del navegador y en cambio nos abra un cuadro de dialogo para elegir guardar o abrir con un programa externo, deberemos modificar los encabezados HTTP que imprime Nagios en dicho momento.

cgi/avail.c

```
        if(output_format==HTML_OUTPUT)
                printf("Content-type: text/html\r\n\r\n");
        else{
                printf("Content-type: plain/text\r\n\r\n");
                return;
                }
```

Lo cambiaremos como :

```
        if(output_format==HTML_OUTPUT)
                printf("Content-type: text/html\r\n\r\n");
        else{
                printf("Content-Disposition: attachment;filename=informe_mensual-
nagios.csv
                        Content-type:application/csv\r\n\r\n");
                return;
                }
```

Scripts útiles

A continuacion se detallan scripts que pueden ser de utilidad para el dia a dia con Nagios

Volcado de estado actual : *Haciendo uso de la clase StatusLog disponible en CPAN podemo parsear el contenido del archivo status.dat de Nagios, para luego exportar su contenido*

```
#!/usr/bin/perl

use Nagios::StatusLog;

my $log = Nagios::StatusLog->new(
Filename => "/dev/shm/status.dat",
Version => 3.0
);
print("Content-type: text/xml\n\n");
print("<?xml version='1.0'?>");
print("<status>\n");
foreach my $host ($log->list_hosts()) {
        print("<host>\n<name>$host</name>\n<services>");
        foreach my $serv ($log->list_services_on_host($host)) {
                print ("<service>\n");
                print (ref $serv);
                my $st  = $log->service($host, $serv);
                foreach $tag ($st->list_tags()) {
                        print("<$tag>$$st{$tag}</$tag>\n");
                }
                print ("</service>\n");
        }
        print("</services>\n</host>\n")
}
print("</status>");
```

Recordemos que para instalar modulo de CPAN deberemos ejecutar

```
# perl -MCPAN -e shell
```

```
cpan> install MODULO::NOMBRE
Ejemplo para este caso
cpan> install Nagios::StatusLog
```

Ejemplo de uso de Nagios::Report

```
use Nagios::Report ;
my $x = Nagios::Report->new(q<local_cgi nagios_web_server nagios_user>, [ '24x7' ],
'thismonth')
    or die "Can't construct Nagios::Report object." ;
my @these_fields = qw(
        HOST_NAME
        PERCENT_TOTAL_TIME_UP
  ) ;
$x->mkreport(
\@these_fields,
                            # All records
                            #   sub { 1 },
                            # All records whose HOST_NAME starts with 'Alb'
                            #   sub { my %F = @_; my $h = $F{HOST_NAME}; $h =~
/^Alb/ },
                            # Regrettably, this is _NOT_ the same since
                            # @_ can't be used as a hash.
                            #   sub { $_{HOST_NAME} =~ /^Alb/ }
                            # All records with an up time percent < 98%
sub {
    my %F = @_ ;
    my $u = $F{PERCENT_TOTAL_TIME_UP};
    $u =~ s/%//;
    $u > 0
},
# Sort order
&comp( alpha => 0, ascend => 0, fields => [ qw(TOTAL_TIME_DOWN
TOTAL_TIME_UNREACHABLE) ]),
# Sorts descending by max of TOTAL_TIME_DOWN and TOTAL_TIME_UNREACHABLE
                            # DIY sorters remember that $a and $b _must_ be in
Nagios::Report package.
                            # eg by TOTAL_DOWN_TIME descending.
                            #   sub { my %f = @_ ;
                            #         package Nagios::Report;
                            #         $b->[$f{TOTAL_TIME_DOWN}] <=> $a-
>[$f{TOTAL_TIME_DOWN}]
                            #        },
                            # Same as
                            #  &comp(alpha => 0, ascend => 0, fields =>
['TOTAL_TIME_DOWN'])
                            # Same but harder,
                            #  sub { package Nagios::Report; $b->[16] <=> $a-
>[16] },
                            # Optional callback to add or mangle fields.
                            # Add 2 fields for downtime vals in hours minutes and
secs.
sub {
    $F = shift @_;
    $F->{PERCENT_TOTAL_TIME_UP} =~ s/%//;
    $F->{TIME_UP_HHMMSS}      =     t2hms( $F->{TOTAL_TIME_UP} ),
    $F->{TIME_DOWN_HHMMSS}          =     t2hms( $F->{TOTAL_TIME_DOWN} ),
    $F->{TIME_UNREACH_HHMMSS}=     t2hms( $F->{TOTAL_TIME_UNREACHABLE} ) ;
    qw(TIME_UP_HHMMSS TIME_DOWN_HHMMSS TIME_UNREACH_HHMMSS)
}
) ;
$x->csv_dump ;
```

Volcado de variables de entorno de Nagios (esto es útil cuando programamos

plugins que requiren variables de entorno y no sabemos como las tenemos que ver o capturar) :

```
#!/usr/bin/perl -w
my $o_hoststate        = $ENV{NAGIOS_HOSTSTATE};
# Nagios monitored host check output data
my $o_hostoutput       = $ENV{NAGIOS_HOSTOUTPUT};
# Nagios date when the event was recorded
my $o_datetime         = $ENV{NAGIOS_LONGDATETIME};
# The recipients defined in $CONTACTEMAIL$
my $o_to_recipients    = $ENV{NAGIOS_CONTACTEMAIL};
use Data::Dumper::Concise;
print "<pre>";
print Dumper(%ENV);
print "</pre>";
```

Ejemplo de mk_livestatus

```
#!/usr/bin/python
#
# Sample program for accessing the Livestatus Module
# from a python program
socket_path = "/var/lib/nagios/rw/live"

import socket
s = socket.socket(socket.AF_UNIX, socket.SOCK_STREAM)
s.connect(socket_path)

# Write command to socket
s.send("GET hosts\n")

# Important: Close sending direction. That way
# the other side knows, we are finished.
s.shutdown(socket.SHUT_WR)

# Now read the answer
answer = s.recv(100000000)

# Parse the answer into a table (a list of lists)
table = [ line.split(';') for line in answer.split('\n')[:-1] ]

print table
```

Llamando al socket directamente desde PHP.

```
$data_address="unix:///usr/local/nagios/var/rw/live";
$query="GET services\nFilter: host_groups >= ServiciosGenerales\nColumns: host_name
host_address display_name service_state plugin_output notes_url\nOutputFormat:
json\n\n";
$fp = fsockopen($data_address,0);
if (!$fp) {
    echo "$errstr ($errno)<br />\n";
} else {
    $returnval = "";
    fwrite($fp,$query);
    while (!feof($fp)) {
      $returnval = $returnval.fgets($fp, 128);
#     print $returnval;
    }
fclose($fp);
if ($returnval == "[] ")
    $returnval = "";
}
print_r(json_decode($returnval,true));
```

Llamando al socket indirectamente desde PHP utilizando *unixcat*.

Ejemplo de como obtener la lista de equipos y su estado desde php y exportarlo a formato CSV

```
$lista_servidores = shell_exec("/bin/echo -e \"GET hosts\nColumns: name address
alias state\n\" | /usr/local/bin/unixcat /usr/local/nagios/var/rw/live");
$lista_hosts = str_getcsv($lista_servidores, "\n");
```

Convertir el timestamp de nagios.log a human readable

```
tail -f nagios.log | perl -pe 's/(\d+)/localtime($1)/e'
```

LQL - The Livestatus Query Language

El lenguaje de consultas de Livestatus, basicamente consiste en consultas del tipo GET (case sensitive) con estos parámetros a los cuales consideraremos como equivalente a tablas SQL :

- hosts - your Nagios hosts
- services - your Nagios services, joined with all data from hosts
- hostgroups - you Nagios hostgroups
- servicegroups - you Nagios servicegroups
- contactgroups - you Nagios contact groups
- servicesbygroup - new in 1.1.3 - all services grouped by service groups
- servicesbyhostgroup - new in 1.1.3 - all services grouped by host groups
- hostsbygroup - new in 1.1.3 - all hosts group by host groups
- contacts - your Nagios contacts
- commands - your defined Nagios commands
- timeperiods - time period definitions (currently only name and alias)
- downtimes - all scheduled host and service downtimes, joined with data from hosts and services.
- comments - all host and service comments
- log - a transparent access to the nagios logfiles (include archived ones)ones
- status - general performance and status information. This table contains exactly one dataset.
- columns - a complete list of all tables and columns available via Livestatus, including descriptions!

Ver como Human Readable el timestamp del archivo nagios.log

```
# cat /usr/local/nagios/var/nagios.log | perl -pe 's/(\d+)/localtime($1)/e'
```

NetMySLA

netMySLA es una variedad de procedimientos almacenados de MySQL. Se creó un procedimiento para calcular un único valor de disponibilidad de servicio. El otro itera a través de todos los hosts y servicios para crear valores de disponibilidad de servicio y lo divide en plazos. Por lo tanto, usted puede calcular los valores una vez al día y consulta la tabla de resultados para obtener sus datos.

https://www.nagiosforge.org/gf/project/netmysla/

Ejemplo de consulta con netMySLA

Lista de argumentos

1. Hostname varchar (objects)
2. Servicename varchar (objects) can be null
3. Consider Downtime boolean
4. Consider Acknowledge boolean
5. Consider Unknown boolean
6. Consider Warning boolean
7. The initial state boolean (true = up, false = down)
8. Startdate datetime
9. Stopdate datetime
10. Debugflag boolean (Display some workflow output)
11. quiet (No resultset, only inserts the values into a table → np_sla)

```
CALL np_checkAvailability('srv-web1', 'HTTP', TRUE, TRUE, TRUE, TRUE, NULL, '2008-
01-11 00:00:00', '2009-07-01 00:00:00', FALSE, FALSE);
```

sla_id	host_object_id	host_name	service_object_id	service_name	outage_percent	availability_percent
1	122	srv-web1	1761	HTTP	0.0227476	99.9772

Ejemplo de como llamar al procedimiento desde PHP5, para por ejemplo conocer el SLA de **x** servicio en **n** cantidad de hosts.

```
$consulta ="SELECT nagios_hosts.display_name
    FROM
    nagios_hostgroups,nagios_instances,
    nagios_hosts,nagios_hostgroup_members,
    nagios_objects
    WHERE
    nagios_hostgroups.hostgroup_id=nagios_hostgroup_members.hostgroup_id
    AND
    nagios_hostgroup_members.host_object_id=nagios_hosts.host_object_id
    AND
    nagios_hostgroups.hostgroup_object_id=nagios_objects.object_id
    AND
    nagios_objects.name1 = '{hostgroup_a_consultar}'
    ORDER BY nagios_hosts.display_name ASC";



$pdo = new PDO("mysql:dbname=nagios;host=127.0.0.1", "nagios","nagios");
$lista_hosts = $pdo->prepare($consulta);
$lista_hosts->execute();
$hosts = $lista_hosts->fetchall();

$service = "Servicio a consultar en los hosts";

foreach($hosts as $host) {
      $query = $pdo->prepare("call np_checkAvailability('".$host['display_name']."',
'".$service."', false, false, false, false, true, '".$start_time."', '".
$end_time."', false, false)");
      $query->execute();
      print_r($query->fetch());
}
```

Devolviendonos algo como esto por cada hosts:

```
Array
(
    [sla_id] => 1
    [host_object_id] => 463
    [host_name] => 001
    [service_object_id] =>
    [service_name] => Servicio
    [outage_percent] => 0
    [availability_percent] => 100
)
```

Tambien podemos utilizar el script *np_aggregate.sh* para ejecutarlo por medio de cron y tener autogenerados los informes, por cada hosts y cada servicios, organizados *diaria, semanal, mensual y anualmente*.

Documentacion al vuelo

Para tener una documentacion actualizada de los equipos y sus servicios, lo ideal es implementar un wiki.

DokuWiki es un software para gestión de webs colaborativas de tipo wiki, escrito en lenguaje PHP y distribuido en códigon abierto bajo la licencia GPL.

Está enfocado para ser usado por grupos de desarrolladores, grupos de trabajo en general y pequeñas compañías.

Su sintaxis es similar a la de MediaWiki, aunque a diferencia de este software, la información se almacena en archivos de texto planos, por lo que no requiere el uso de una base de datos.

Características:

- Gestión de espacios de contenidos que permite un almacenamiento ordenado de los documentos.
- Soporte para imágenes y otros contenidos multimedia.
- Índices automatizados de contenidos.
- Control de versiones.
- Corrector ortográfico opcional.
- Interfaz traducido a múltiples idiomas, incluyendo el castellano.
- Posibilidad de utilizar plantillas de diseño.
- Disponibilidad de complementos (plugins) para extender la funcionalidad.
- Control de bloqueos para solucionar problemas de concurrencia.
- Gestión de usuarios.
- Búsqueda de texto completo.

Para integrarlo con Nagios podemos implementar un script en PHP como este :

```
<?php
/**
 * Forwarder to doku.php
 *
 * @license    GPL 2 (http://www.gnu.org/licenses/gpl.html)
 * @author     Joerg Linge <pitchfork@ederdrom.de>
 */
```

```
$host=str_replace(' ', '_',strtolower($_GET['host']));

$srv=str_replace(' ', '_',strtolower($_GET['srv']));


if ($host!="" && $srv!="") {

        header("Location: doku.php?id=nagios:".$host.":".$srv);

        exit;

} elseif($host!="") {

        header("Location: doku.php?id=nagios:".$host.":host");

        exit;

} else {

        header("Location: doku.php?id=nagios:index");

        exit;

}
?>
```

Con lo cual deberemos agregar algunas directivas a la configuración de Nagios.

Ejemplo de un serviceextinfo, aunque tambien lo podemos aplicar dentro de la plantilla inicial del servicio.

```
define serviceextinfo {
  host_name router
  service_description ping
  notes_url /wiki/nagios.php?host=$HOSTNAME$&srv=$SERVICEDESC$
  icon_image help.png
}
```

Desde Dokuwiki haciendo uso de la extension PHP, podemos embeber los datos actuales de los objetos de Nagios, gracias a MK Live Status

```
<php>
$lista_servidores = shell_exec("/bin/echo -e \"GET hosts\nColumns: name address
alias state\n\" | /usr/local/bin/unixcat /usr/local/nagios/var/rw/live");
$lista_hosts = str_getcsv($lista_servidores, "\n");
echo "<div class='level2'>
<div class='table sectionedit3'>
<table class='inline'>
<th>Host</th><th>IP</th><th>Descripcion</th><th>Estado Actual</th><th>Metricas /
Informes</th>
";

foreach ($lista_hosts as $host) {
    $host = explode(";", $host);
    print "<tr><td>";
    print "<a href='doku.php?id=nagios:servidores:".$host['0']."'>".
$host['0']."</a>";
    print "</td><td>";
    print $host['1'];
    print "</td><td>";
    print $host['2'];
    print "</td><td>";
```

```
    if($host['3']==0) print "<a href='/nagios/cgi-bin/extinfo.cgi?type=1&host=".
$host['0']."'><font color='#008000'>ARRIBA</font></a>";
    if($host['3']==1) print "<a href='/nagios/cgi-bin/extinfo.cgi?type=1&host=".
$host['0']."'><font color='#FF0000'>CA&Iacute;DO</font><img
src='/nagios/wiki/lib/images/smileys/icon_exclaim.gif' class='middle' /></a>";
    if($host['3']==2) print "<a href='/nagios/cgi-bin/extinfo.cgi?type=1&host=".
$host['0']."'><font color='#FF0000'>UNREACHEABLE</font><img
src='/nagios/wiki/lib/images/smileys/icon_exclaim.gif' class='middle' /></a>";
    print "</td>";
    print "<td><a href='/pnp4nagios/index.php/graph?host=".
$host['0']."'>Performance</a> / <a href='/nagios/cgi-bin/avail.cgi?
get_date_parts=&host=".$host['0']."'>Disponibilidad</a></td></tr>";
}

echo "</table></div></div>";
</php>
```

Tambien desde dokuwiki por ejemplo si necesitamos podemos consultar comandos por SSH para mostrar su salida en la documentación de nuestro equipo, como un plus de un dato en tiempo real.

```
<php>
if (!function_exists("ssh2_connect")) die("function ssh2_connect doesn't exist");
if(!($con = ssh2_connect("oracle", 22))){
    echo "No puedo establecer una conexion SSH\n";
} else {
if (ssh2_auth_pubkey_file($con, 'root',
                          '/home/usuario/.ssh/id_rsa.pub',
                          '/home/usuario/.ssh/id_rsa', 'secret')) {
//  echo "Public Key Authentication Successful\n";
} else {
  die('Public Key Authentication Failed');
}
        if (!($stream = ssh2_exec($con, "/home/usuario/dokuwiki_archlogs.pl" ))) {
            echo "fail: unable to execute command\n";
        } else {
            // collect returning data from command
            stream_set_blocking($stream, true);
            $data = "";
            while ($buf = fread($stream,4096)) {
                $data .= $buf;
            }
            fclose($stream);
            print $data;
        }
    }
</php>
```

```
!/usr/bin/perl
use strict;
use warnings;

sub get_sorted_files {
   my $path = shift;
   opendir my($dir), $path or die "no puedo abrir $path: $!";
   my %hash = map {$_ => (stat($_))[9] || undef} # saltar listas vacias
    map  { "$path$_" }
    grep { m/.dbf/i }
    readdir $dir;
    closedir $dir;
    return %hash;
}

my %files = get_sorted_files("/oracle/arclog/DBID/");
print "
```

```
    <table class='inline'>
    <th>Archivo</th><th>Timestamp</th>";
foreach my $key (sort{$files{$a} <=> $files{$b}} keys %files) {
    my $filename = $key;
    $filename =~ s/\.\///g;
    $filename =~ s/\.\.//g;
    print "<tr><td>$filename</td><td>", scalar localtime($files{$key}),
"</td></tr>\n";
}
print "</table>";
```

si queremos mostrar los datos parseados de Dokuwiki como un simple HTML en algún sector fuera de la Wiki, podemos hacer uso de este script :

```
<?php //DokuWiki exporter - copylefted by Harvie.cz 2o1o (copy this file to dokuwiki
root)
header('Content-Type: text/html; charset=utf-8');
#if(!defined('DOKU_INC')) define('DOKU_INC',dirname(__FILE__).'/');
define('DOKU_INC','/usr/local/nagios/share/wiki/');
require_once(DOKU_INC.'inc/init.php');
require_once(DOKU_INC.'inc/common.php');
require_once(DOKU_INC.'inc/events.php');
require_once(DOKU_INC.'inc/parserutils.php');
require_once(DOKU_INC.'inc/auth.php');

function p_file_xhtml($id, $excuse=false){
    if(@file_exists($id)) return p_cached_output($id,'xhtml',$id);
    return p_wiki_xhtml($id, '', $excuse);
}

if($_GET['pagina']) {
  echo p_file_xhtml($_GET['pagina'], false);
}
?>
```

Plugins interesantes

check_multiaddr

A menudo sucede que tenemos hosts a monitorear con multiples ip disponibles, y dado el caso por ejemplo que necesitemos chequear por cualquiera de las dos IP para consultar disponibilidad, y que nuestro plugin intente por una o por otra con un timeout definido, para eso ya existe la solucion *check_multiaddr*, con la cual no necesitamos realizar ninguna modificacion a el código existente de nuestros plugins, por ejemplo en la direccion ip del host a monitorear puede ir *192.168.0.1,192.168.0.11,192.168.0.21*. Luego en la entrada de nuestro comando de chequeo realizamos modificaciones para que quede algo como esto :

```
define command{
        command_name check_multiple_dns
        command_line $USER1$/check_multiaddr.pl $USER1$/check_dns -H $ARG1$ -s
$HOSTADDRESS$
}
```

Con lo cual el utilitario *check_multiaddr* actuara simplemente de envoltorio de nuestros plugins, encargandose del timeout entre cada consulta a cada direccion y de devolver su salida con su exit status correspondiente.

Adjunto : check_multiaddr y una version alternativa basada en el mismo código

check_multiaddr.pl.txt.gz

check_all_ips.pl.txt.gz

apache

/etc/apache2/conf.d/security

Ciertos parametros son útiles para evitar cierta exposición de nuestro Apache.

```
ServerTokens Prod
ServerSignature Off
TraceEnable Off
```

check_sap

Para que funcionen correctamente algunas cosas de los plugins de Nagios CCMS, hay que realizar algunas minimas modificaciones por ejemplo en : En el plugin “Nagios SAP CCMS” hay que modificar algunas lineas de los archivos agnt_mon.h y sap_moni_ccm.h ya que en estos se establece el path de acceso a los archivos de configuración que por defecto los busca en /etc/sapmon, pero nuestro objetivo es que los busque en /usr/local/nagios/etc/sapmon, de una cierta manera quede mas centralizao u ordenado. Para mas información sobre SAP pueden ver el apartado aprendiendo SAP.

- **agnt_mon.h**

```
    #define AGENT_INI_FILE “/usr/local/nagios/etc/sapmon/agent.cfg”
    #define AGENT_LOGIN_FILE “/usr/local/nagios/etc/sapmon/login.cfg”
    #define AG_EZ_FILE “/usr/local/nagios/etc/sapmon/moni_tr.cfg”
```

- **sap_moni_ccm.h**

```
    #define AGENT_INI_FILE “/usr/local/nagios/etc/sapmon/agent.cfg”
    #define AGENT_LOGIN_FILE “/usr/local/nagios/etc/sapmon/login.cfg”
```

Archivo */usr/local/nagios/etc/sapmon/login.cfg* de ejemplo :

```
[LOGIN_PRD]
LOGIN=-d PRD -u nagios -p password -c 300 -h 10.1.1.90 -s 00
```

Archivo */usr/local/nagios/etc/sapmon/agent.cfg* de ejemplo :

DESCRIPTION = <Description of the Template> Arbitrary description field
MONI_SET_NAME = <Monitor collection> Monitor collection in RZ20
MONI_NAME = <Monitor name> Name of the specific monitor
MAX_TREE_DEPTH = <number> Monitor loading depth
PATTERN_0 = <SAPSID>\<Context>\<Monitor object>\<Monitor attribut>

```
[TEMPLATE_00]
DESCRIPTION="Load Average"
MONI_SET_NAME=SAP CCMS Admin Workplace
MONI_NAME="Operating System"
MAX_TREE_DEPTH=0
PATTERN_0="BCE\bcemain_BCE_26\CPU\5minLoadAverage"
PATTERN_0="*\*\CPU\5minLoadAverage"

[TEMPLATE_01]
MONI_SET_NAME=SAP CCMS Admin Workplace
MONI_NAME="Operating System"
MAX_TREE_DEPTH=0
PATTERN_2="BCE\bcemain_BCE_26\CPU\CPU_U*"

[TEMPLATE_02]
VALUE=DIALOG_RESPONSE_TIME

[TEMPLATE_03]
SYSTEM=BCE
```

```
APPL-SERVER=bcemain*
VALUE=DIALOG_RESPONSE_TIME

[TEMPLATE_04]
MONI_SET_NAME="SAP CCMS Monitor Templates"
MONI_NAME="Dialog Overview"
PATTERN_0="BCE\*\Dialog\FrontEndNetTime"


[TEMPLATE_05]
MONI_SET_NAME="SAP CCMS Monitor Templates"
MONI_NAME="Dialog Overview"
PATTERN_0="BCE\PWD*\Dialog\FrontEndNetTime"

[TEMPLATE_005]
MONI_SET_NAME="SAP CCMS Monitor Templates"
MONI_NAME="Dialog Overview"
PATTERN_0="P10\*\Dialog\FrontEndNetTime"

[TEMPLATE_06]
MONI_SET_NAME="SAP CCMS Monitor Templates"
MONI_NAME="Dialog Overview"
PATTERN_0="*"

[TEMPLATE_07]
MONI_SET_NAME="SAP CCMS Monitor Templates"
MONI_NAME="Availability and Performance Overview"
PATTERN_0="*\Availability\*_BCE*\"

# Standard SAP Templates
[TEMPLATE_99]
VALUE=CHECK_SAP_SYSTEMS

[TEMPLATE_105]
MONI_SET_NAME="SAP CCMS Monitor Templates"
MONI_NAME="Dialog Overview"
PATTERN_0="*UsersLoggedIn"

[TEMPLATE_110]
MONI_SET_NAME="SAP CCMS Monitor Templates"
MONI_NAME="Entire System"
PATTERN_0="*EsAct"

[TEMPLATE_200]
MONI_SET_NAME="SAP CCMS Monitor Templates"
MONI_NAME="Database"
PATTERN_0="*Fullest tablespace"

[TEMPLATE_210]
MONI_SET_NAME="SAP CCMS Monitor Templates"
MONI_NAME="Entire System"
PATTERN_0="*DBRequestTime"

[TEMPLATE_300]
MONI_SET_NAME="SAP CCMS Monitor Templates"
MONI_NAME="Operating System"
PATTERN_0="*5minLoadAverage"

[TEMPLATE_999]
MONI_SET_NAME="SAP CCMS Monitor Templates"
MONI_NAME="Entire System"
PATTERN_0="*"

[TEMPLATE_060]
```

```
#DESCRIPTION="Free Swap"
MONI_SET_NAME=SAP CCMS Admin Workplace
MONI_NAME="Operating System"
#MAX_TREE_DEPTH=0
PATTERN_0="*\*\Swap_Space\Freespace"

[TEMPLATE_070]
DESCRIPTION=Dialog Response Time
MONI_SET_NAME=SAP CCMS Monitor Templates
MONI_NAME=Dialog Overview
PATTERN_0=P10\*\Dialog\ResponseTime

[TEMPLATE_071]
DESCRIPTION=Dialog Response Time
MONI_SET_NAME=SAP CCMS Monitor Templates
MONI_NAME=Dialog Overview
PATTERN_0=*

[TEMPLATE_09]
DESCRIPTION="DialogResponseTime"
MONI_SET_NAME=SAP CCMS Monitors for Optional Components
MONI_NAME="Logon Load Balancing"
MAX_TREE_DEPTH=0
PATTERN_0="*\*\Dialog\ResponseTime"

[TEMPLATE_007]
DESCRIPTION="TEST"
MONI_SET_NAME=SAP CCMS Monitors for Optional Components
MONI_NAME="Logon Load Balancing"
MAX_TREE_DEPTH=0
PATTERN_0="SAP\Server3_SAP_00\R3Services\Dialog\ResponseTime"


[TEMPLATE_900]
DESCRIPTION=Java
MONI_SET_NAME=SAP J2EE Monitor Templates
MONI_NAME=Heartbeat
PATTERN_0=*

[TEMPLATE_901]
DESCRIPTION=Java
MONI_SET_NAME=SAP J2EE Monitor Templates
MONI_NAME=Applications
PATTERN_0=*

[TEMPLATE_06]
DESCRIPTION=Users-Logged-On
MONI_SET_NAME="SAP CCMS Monitor Templates"
MONI_NAME="Dialog Overview"
MAX_TREE_DEPTH=0
PATTERN_0=SID\hostname*\Di*\Us*


[TEMPLATE_870]
DESCRIPTION=Java
MONI_SET_NAME=Test J2EE Monitor Set
MONI_NAME=J2EE Engine Kernel
PATTERN_0="XQ1\XQ1 64 Serv 649148450 mchp7tpa\System Threads
Pool\MaximumThreadPoolSize"

[TEMPLATE_875]
DESCRIPTION=Java
MONI_SET_NAME=Test J2EE Monitor Set
MONI_NAME=J2EE Engine Kernel
```

```
PATTERN_0="XQ1\XQ1 64 Disp 649148400 mchp7tpa\General
(MessageContext)\AverageMSProcessTime"


[TEMPLATE_666]
DESCRIPTION=SAP Avg. DB Request Time dehq0srm
MONI_SET_NAME=CENTRAL MONITORING SYSTEM (SAP Basis Kerpen)
MONI_NAME=Test Systems SAP
PATTERN_0=SID\hostname\Dialog\DBRequestTime

[TEMPLATE_667]
MONI_SET_NAME = "SAP CCMS Technical Expert Monitors"
MONI_NAME     = "All Contexts on Local Application Server"
PATTERN_0     = "*"

[TEMPLATE_668]
MONI_SET_NAME = "SAP CCMS Technical Expert Monitors"
MONI_NAME     = "All Contexts on Local Application Server"
PATTERN_0     = "*\SYSTEM\Free space"


[TEMPLATE_471]
MONI_SET_NAME = "SAP CCMS Technical Expert Monitors"
MONI_NAME     = "All Contexts on Local Application Server"
PATTERN_0     = "*\PSAPSR3USR\Free space"

# Monitorear Extended memory en application Server
[TEMPLATE_784]
MONI_SET_NAME="SAP CCMS Monitor Templates"
MONI_NAME="Performance Overview"
PATTERN_0="SID\SID3_00\*EsAct"

[TEMPLATE_785]
MONI_SET_NAME="SAP CCMS Monitor Templates"
MONI_NAME="Performance Overview"
PATTERN_0="SID\SID2_00\*EsAct"

[TEMPLATE_786]
MONI_SET_NAME="SAP CCMS Monitor Templates"
MONI_NAME="Performance Overview"
PATTERN_0="SID\SID1_00\*EsAct"

# Monitorear Extended memory en Central Instance
[TEMPLATE_787]
MONI_SET_NAME="SAP CCMS Monitor Templates"
MONI_NAME="Performance Overview"
PATTERN_0="SID\SID_00\*EsAct"

[TEMPLATE_666]
MONI_SET_NAME = "SID - Monitor"
MONI_NAME     = "All Monitoring Contexts"
PATTERN_0     = "*"

[TEMPLATE_670]
MONI_SET_NAME="SID - Monitor"
MONI_NAME="All Monitoring Contexts"
PATTERN_0="SID\*\*Oracle messages*\ORA*"
```

Si queremos agregar en CCMS el monitoreo de X jobs debemos seguir esta guia http://help.sap.com/saphelp_nw04/helpdata/en/1c/48803d48de0610e10000000a114084/content.htm

```
# Monitoreo de Delay + Runtime de X job
[TEMPLATE_234]
```

```
MONI_SET_NAME="SID - Monitor"
MONI_NAME="All Monitoring Contexts"
PATTERN_0="SID\Background\E_001_RB_06\Delay + Runtime"
```

Ejemplo de como podemos definir el comando en Nagios :

```
define command {
        command_name                    check_sap
        command_line                    $USER1$/ccms/check_sap $ARG1$ $_HOSTSAPID$
        register                        1
}

define command {
        command_name                    check_sap_np
        command_line                    $USER1$/ccms/check_sap_np.sh $ARG1$
$_HOSTSAPID$
        register                        1
}
```

Por ejemplo si la salida del chequeo en SAP nos arroja *pipes* **|**, para que no tengamos problemas con un falso perfdata, podemos apelar a este script :

check_sap_np.sh

```
#!/bin/bash
CMD=`/usr/local/nagios/libexec/ccms/check_sap $1 $2`
EXIT=$?
echo $CMD | /bin/sed -e 's/|/-/g'
exit $EXIT
```

Para poder hacer correctamente el chequeo deberemos defini la variable **$_HOSTSAPID$** dentro de la configuración del host :

```
_SAPID                          PRD
```

Tivoli Storage Manager - TSM

IBM Tivoli Storage Manager es una plataforma de protección de datos y administración para copia de seguridad y restauración. Desde Nagios tenemos una gran variedad de scripts para verificar su correcto funcionamiento interno.

Deberemos tener configurado el cliente TSM dentro del mismo servidor donde tenemos instalado Nagios. Y dentro del archivo **dsm.sys** deberemos tener configurado los múltiples servidores para conectarnos. Es recomendable agregarle una variable dentro de Nagios a cada uno de los hosts definimos que funcione como servidor TSM.

```
SErvername  TSMPRINCIPAL1
  COMMmethod         TCPip
  TCPPort            1500
  TCPServeraddress   10.1.1.99


define host {
      host_name                         server_tsm
      alias                             TSM Server
      address                           10.1.1.99
      use                               plantilla_servidores
      _TSMSERVER                        TSMPRINCIPAL1
}
```

Definicion del commands.cfg para los comandos que utilizan dicha variable como argumento :

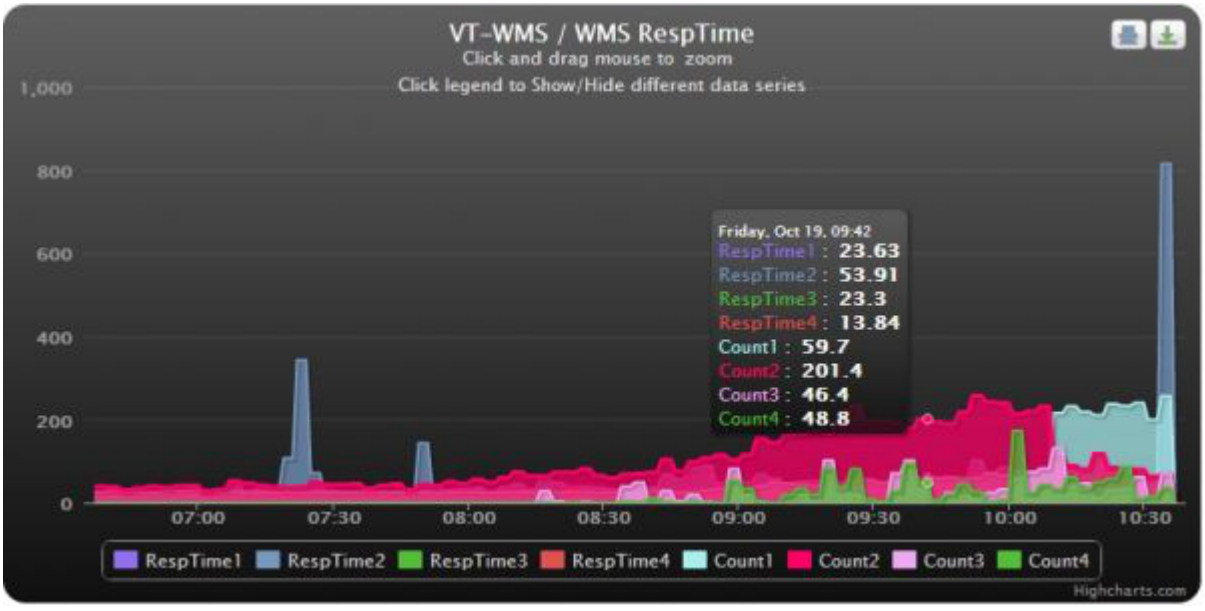
```
$USER1$/tsm/tsm_path.sh             $_HOSTTSMSERVER$
$USER1$/tsm/tsm_disk_pool.sh        $_HOSTTSMSERVER$
$USER1$/tsm/tsm_db_backuptime.sh    $_HOSTTSMSERVER$
$USER1$/tsm/tsm_fulldb_backup.sh    $_HOSTTSMSERVER$
$USER1$/tsm/tsm_schedule_errors.sh $_HOSTTSMSERVER$
$USER1$/tsm/tsm_occupancy.sh        $_HOSTTSMSERVER$
$USER1$/tsmmonitor.pl $ARG1$ $ARG2$      $_HOSTTSMSERVER$
$USER1$/tsm/check_tsm -H $HOSTADDRESS$ -U admin -P admin $ARG1$
```

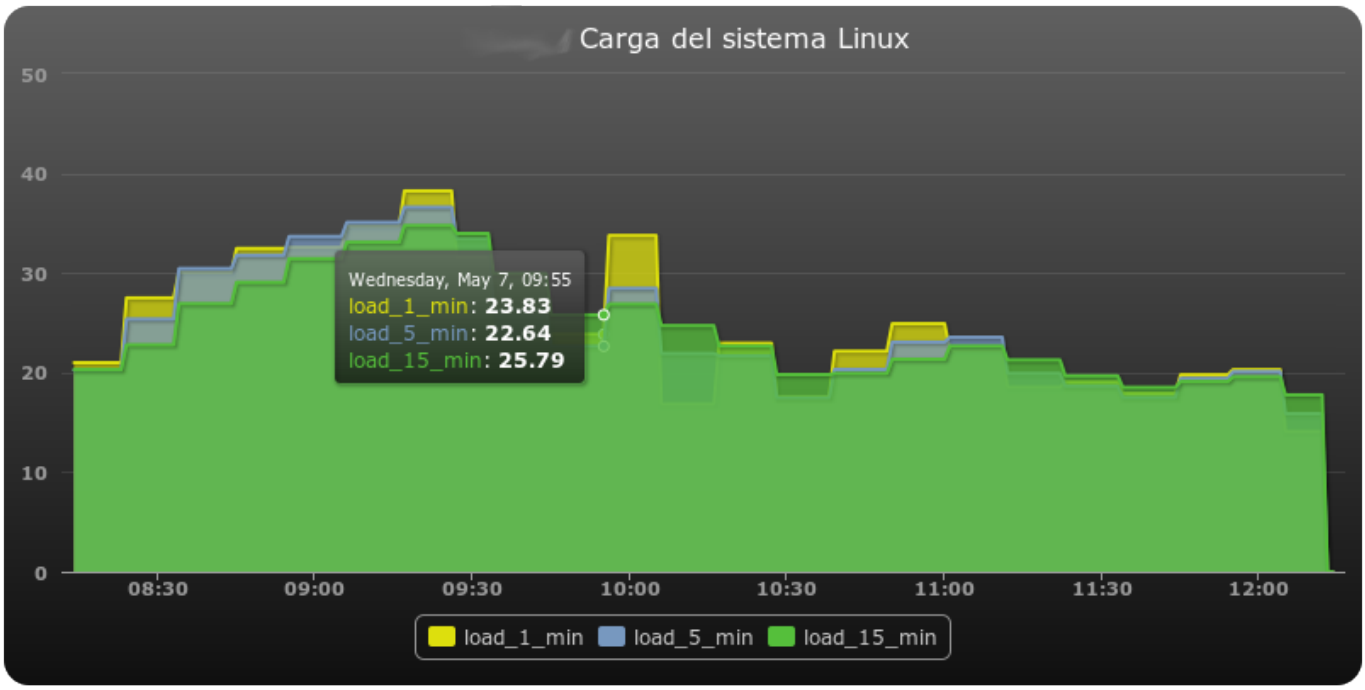
Highcharts for Nagios

Highcharts es una librería de JavaScript para generar gráficos interactivos de líneas, área, barras, columnas, torta y muchos más. Es totalmente compatible con la mayoría de los navegadores incluyendo el Safari para iPhone y el Internet Explorer desde su versión 6. Entre sus características se destacan:

- Gratis para uso No-Comercial
- Puro JavaScript
- Sintaxis de configuración sencilla
- Numerosos tipos de gráficos
- Etiquetas Tooltip
- Múltiples ejes
- Eventos
- Ejes de fecha y hora
- Zooming
- Carga externa de datos
- Gráfico invertido o eje invertido
- Rotación del texto en las etiquetas

En este caso vamos mencionar la implementación para tomar gráficos RRD de PNP4Nagios y transformarlos en en formato AJAX de Highcharts.





Además podemos capturar datos por medio de MK LiveStatus y dibujar gráficos con dichos datos, podemos por ejemplo consultar un array en JSON y escribir las variables, por ejemplo consultamos el numero de servicios en Nagios, luego de dichos servicios cuantos estan en status OK, WARNING y CRITICAL y devolvemos en porcentaje y obtenemos este simple código, que nos va a generar un gráfico que luego podremos exportar en PNG, SVG, JPG, PDF :

```
            data: [{
                    name: 'OK',
                    y: 96.6,
                color: colors[2],
                url: 'http://wiki.cayu.com.ar',
                },{
                    name: 'WARNING',
                    y: 2.0,
                color: colors[6],
                },{
                    name: 'CRITICAL',
                    y: 1.4,
                    sliced: true,
                    selected: true,
                color: colors[5],
                },
```

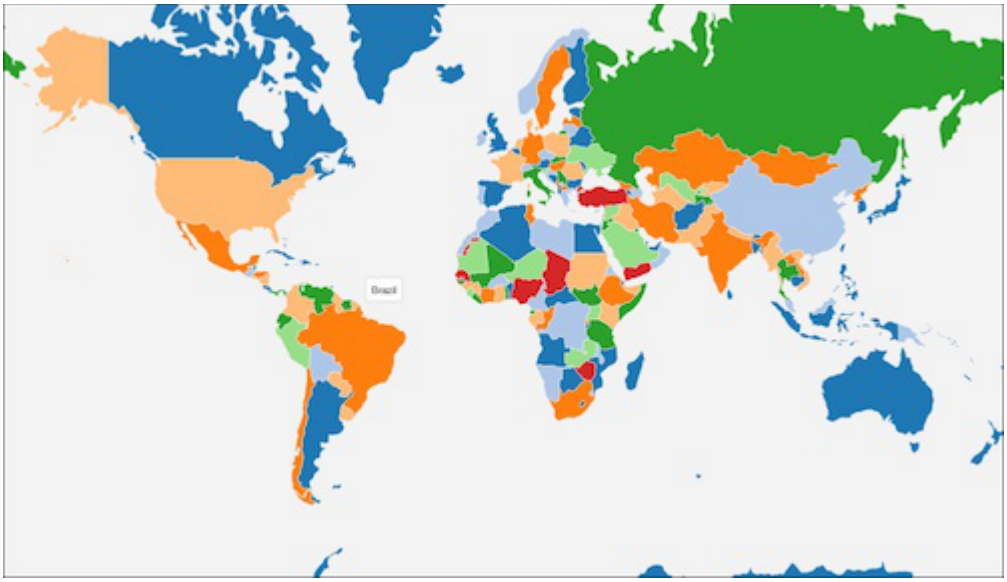
Para poder configurar el módulo de exportación de los gráficos a PNG, SVG, PDF.

Deberemos tener instalado en nuestro servidor el paquete http://xmlgraphics.apache.org/batik/ y luego indicar en el módulo de exportación lo siguiente :

```
define ('BATIK_PATH', 'batik-1.7/batik-rasterizer.jar');
```

jVectorMap

Con jVectorMap podemos dibujar mapas dinámicos vectoriales en AJAX, podemos hacer que por ejemplo al pasar el mouse por cada país por medio de datos obtenidos de MKLiveStatus muestre el estado de hosts y servicios de cada región.



Ejemplo muy simple :

```
    jQuery.noConflict();
    jQuery(function(){
      var $ = jQuery;
      $('#map').vectorMap({
        map: 'ar_mill_en',
      hoverOpacity: 0.7,
      hoverColor: false,
      backgroundColor: 'transparent',
      zoomButtons: false,
      zoomOnScroll: false,
        regionStyle: {
          initial: {
            fill: '#E1AF6C'
          }
      },
      onRegionClick: function(event, code){
          cargar_ajax('hostgroup_info.php?code='+code+'');
      }
       });
    })
```

Extrancción de datos de los RRD de PNP4Nagios

Podemos extraer los datos almacenados en los datos RRD, para luego procesarlos y mostralo en la herramienta de reportes que desarrollemos o por ejemplo, en ves de utilizar Highcharts podemos utilizar la libreria D3js o alguna otra.

https://servidor_nagios/pnp4nagios/xport/csv?host=servidor1&srv=Load%20Linux

timestamp;load_1_min_MIN;load_1_min_MAX;load_1_min_AVERAGE;load_5_min_MIN;load_5_min_MAX;load_5_min_AVERAGE;load_15_min_MIN;load_15_min_MAX;load_15_min_AVERAGE
1425027060;6.0332333333;6.0332333333;6.0332333333;5.899;5.899;5.899;5.7517666667;5.7517666667;5.7517666667
1425027120;6.0332333333;6.0332333333;6.0332333333;5.899;5.899;5.899;5.7517666667;5.7517666667;5.7517666667
1425027180;5.9871333333;5.9871333333;5.9871333333;6.2204;6.2204;6.2204;5.9390333333;5.9390333333;5.9390333333
1425027240;5.9871333333;5.9871333333;5.9871333333;6.2204;6.2204;6.2204;5.9390333333;5.9390333333;5.9390333333 .....

https://servidor_nagios/pnp4nagios/xport/xml?host=servidor2&srv=Load%20Linux

```
.....
      <entry>load_1_min_MIN</entry>
      <entry>load_1_min_MAX</entry>
      <entry>load_1_min_AVERAGE</entry>
      <entry>load_5_min_MIN</entry>
      <entry>load_5_min_MAX</entry>
      <entry>load_5_min_AVERAGE</entry>
      <entry>load_15_min_MIN</entry>
      <entry>load_15_min_MAX</entry>
      <entry>load_15_min_AVERAGE</entry>
    </legend>
  </meta>
  <data>

<row><t>1425027180</t><v>5.9871333333e+00</v><v>5.9871333333e+00</v><v>5.9871333333e
+00</v><v>6.2204000000e+00</v><v>6.2204000000e+00</v><v>6.2204000000e+00</v><v>5.939
0333333e+00</v><v>5.9390333333e+00</v><v>5.9390333333e+00</v></row>

<row><t>1425027240</t><v>5.9871333333e+00</v><v>5.9871333333e+00</v><v>5.9871333333e
+00</v><v>6.2204000000e+00</v><v>6.2204000000e+00</v><v>6.2204000000e+00</v><v>5.939
0333333e+00</v><v>5.9390333333e+00</v><v>5.9390333333e+00</v></row>

<row><t>1425027300</t><v>5.9871333333e+00</v><v>5.9871333333e+00</v><v>5.9871333333e
+00</v><v>6.2204000000e+00</v><v>6.2204000000e+00</v><v>6.2204000000e+00</v><v>5.939
0333333e+00</v><v>5.9390333333e+00</v><v>5.9390333333e+00</v></row>

<row><t>1425027360</t><v>5.9871333333e+00</v><v>5.9871333333e+00</v><v>5.9871333333e
+00</v><v>6.2204000000e+00</v><v>6.2204000000e+00</v><v>6.2204000000e+00</v><v>5.939
0333333e+00</v><v>5.9390333333e+00</v><v>5.9390333333e+00</v></row>

<row><t>1425027420</t><v>5.9871333333e+00</v><v>5.9871333333e+00</v><v>5.9871333333e
+00</v><v>6.2204000000e+00</v><v>6.2204000000e+00</v><v>6.2204000000e+00</v><v>5.939
0333333e+00</v><v>5.9390333333e+00</v><v>5.9390333333e+00</v></row>
....
```

https://servidor_nagios/pnp4nagios/xport/json?host=servidor3&srv=Load%20Linux

```
{"meta":
{"start":"1425027180","step":"60","end":"1425041580","rows":"241","columns":"9","leg
end":{"entry":
["load_1_min_MIN","load_1_min_MAX","load_1_min_AVERAGE","load_5_min_MIN","load_5_min
_MAX","load_5_min_AVERAGE","load_15_min_MIN","load_15_min_MAX","load_15_min_AVERAGE"
]}},"data":{"row":[{"t":"1425027180","v":
["5.9871333333e+00","5.9871333333e+00","5.9871333333e+00","6.2204000000e+00","6.2204
000000e+00","6.2204000000e+00","5.9390333333e+00","5.9390333333e+00","5.9390333333e+
00"]},{"t":"1425027240","v":
["5.9871333333e+00","5.9871333333e+00","5.9871333333e+00","6.2204000000e+00","6.2204
000000e+00","6.2204000000e+00","5.9390333333e+00","5.9390333333e+00","5.9390333333e+
00"]},{"t":"1425027300","v":
```

```
["5.9871333333e+00","5.9871333333e+00","5.9871333333e+00","6.2204000000e+00","6.2204
000000e+00","6.2204000000e+00","5.9390333333e+00","5.9390333333e+00","5.9390333333e+
00"]},{"t":"1425027360","v":
["5.9871333333e+00","5.9871333333e+00","5.9871333333e+00","6.2204000000e+00","6.2204
000000e+00","6.2204000000e+00","5.9390333333e+00","5.9390333333e+00","5.9390333333e+
00"]},{"t":"1425027420","v":
["5.9871333333e+00","5.9871333333e+00","5.9871333333e+00","6.2204000000e+00","6.2204
000000e+00","6.2204000000e+00","5.9390333333e+00","5.9390333333e+00","5.9390333333e+
00"]},{"t":"1425027480","v":
....
```

Extrancción de datos de las imágenes de los RRD de PNP4Nagios

Si por ejemplo necesitamos embeber las imágenes de pnp4nagios dentro de por ejemplo un reporte en PDF que realizamos con algun script en este caso en PHP, podemos ver el siguiente ejemplo :

```php
<?php
$HOST="srvlinux01";
$SERVICE="Linux Load Average";
$PNPURL="http://nagios.cayu.com.ar/pnp4nagios";
$REQUEST_URI="image?host=$HOST&srv=$SERVICE&view=1&source=0";
$GET="$PNPURL/".urlencode($REQUEST_URI);
print(file_get_contents($GET));
?>
```

Interfaz administrativa

Anteriormente se vio de manera detallada las directivas de configuracion necesarias para el funcionamiento de un servicio Nagios, igualmente esta configuracion se puede establecer por medio de una interfaz web, la misma es NagiosQL, es una aplicacion desarrollada en PHP.

Para funcionar dentros de nuestro actual servidor Nagios:

- PEAR Module: HTML_Template_IT 1.1 o superior
- PHP Extension: gettext
- PHP Extension: mysql
- PHP Extension: ftp
- Javascript en el navegador

Para instalar por ejemplo el soporte HTML_Template_IT, deberemos ejecutar:

```
pear install HTML_Template_IT
```

O si nos encontramos en una distribucion derivada de Debian

```
apt-get install php-html-template-it
```

Ademas si queremos soporte de idiomas deberemos asegurarmos que los locales de nuestro sistema tengan generado su enconding, ej. es_ES.utf8. Para ver los enconding disponibles deberemos ejecutar “locale -a”.

Si realizamos la instalacion en un hosts con HTTPS, deberemos modificar el campo *nagiosql.tbl_settings*, de *http* a *https*.

Deberemos instalar NagiosQL donde residen los archivos HTML de Nagios, usualmente en */usr/local/nagios/share*. Nuestra configuracion actual de Nagios

debera ser importada dentro de NagiosQL, para asi poder administrarla. Al realizar la importacion nos quedara una nomenclatura de directorios como esta

```
/etc/nagiosql/                -> Archivos de configuracion
      "        /hosts          -> Configuracion de Hosts
      "        /services     -> Configuracion de Servicios
      "        /backup/      -> Backups
      "      "  /hosts     -> Backups de Hosts
      "      "  /services  -> Backups de Servicios
```

Por cada cambio que se realiza NagiosSQL hace una copia de seguridad de dicha configuracion.

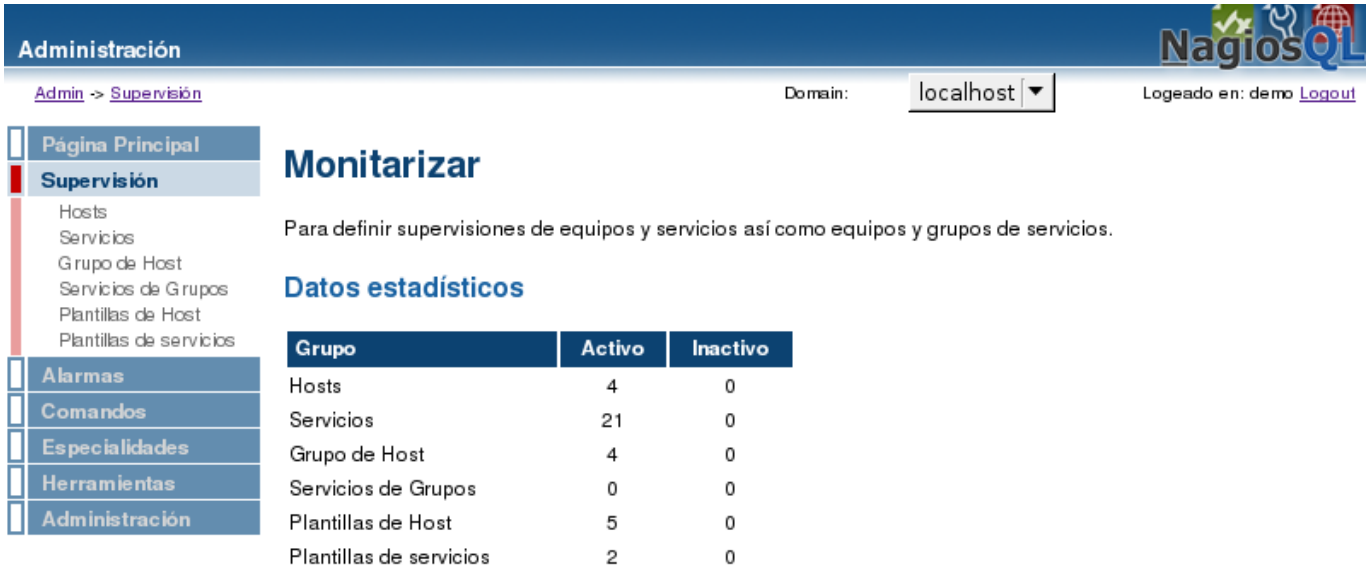
Para comenzar con la instalacion deberemos crear un archivo dentro del directorio de nagiosQL *touch nagiosql3/install/ENABLE_INSTALLER*. Ahora podremos comnezar la instalacion via web, una vez finalizada la instalacion deberemos eliminar el archivo creado *rm -f nagiosql3/install/ENABLE_INSTALLER*

Introduccion de uso de NagiosQL

Al ingresar a la ubicacion de NagiosQL nos aparecera una pantalla de login, en la cual deberemos indentificarnos con el usuario que creamos al momento de la instalacion.

Una vez ingresados al sistema, nos aparecera un menu con enlaces de uso a la izquierda, con diferentes secciones

Dentro de la seccion supervision tendremos un resumen con el conteo de elementos



SubSección Hosts: *Dentro de este apartado, tendremos un listado de los hosts presentes en la configuracion, indicando su estado si es activo o no, y a su vez dandonos la posibilidad de modificarlos y personalizar su modo de chequeo, alarmas etc.*

SubSección Servicios: *Dentro de este apartado, tendremos un listado de los servicios asignados a sus correspondientes hosts, indicando su estado si es activo o no, y a su vez dandonos la posibilidad de modificarlos.*

SubSección Grupos de Hosts: *Dentro de este apartado, tendremos un listado de los grupos de hosts, en donde queramos definir un conjunto de hosts que tengan que ver entre sí por algún servicio o alguna función que cumplan.*

SubSección Grupo de Servicios: *Dentro de este apartado, tendremos un listado de los grupos de servicios, en donde queramos definir un conjunto de servicios que tengan que ver entre sí por alguna dependencia o que cumplan funciones parecidas.*

SubSección Plantillas de Hosts: *Dentro de este apartado, tendremos las plantillas que utilizaremos para afectar a los hosts de manera común en lo que respecta a intervalos de alarmas/notificaciones, modo de chequeo, contactos a quienes enviar datos, agrupación, servicios asignados, auto definido como una forma centralizada de cambiar la configuración de determinado conjunto de hosts.*

SubSección Plantillas de Servicios: *Dentro de este apartado, similar al anterior tendremos las plantillas que utilizaremos para afectar a los servicios de manera común en lo que respecta a intervalos de alarmas/notificaciones, modo de chequeo, contactos a quienes enviar datos, agrupación, hosts en cuales ejecutarse, auto definido como una forma centralizada de cambiar la configuración de determinado conjunto de servicios.*

SubSección Hosts

Al ingresar a este apartado de administración nos encontraremos con una pantalla similar a esta con la lista de hosts presentes en el sistema de monitoreo, indicándonos cuales se encuentran activos y si su configuración se encuentra actualizada con respecto a la presente en el archivo de configuración.

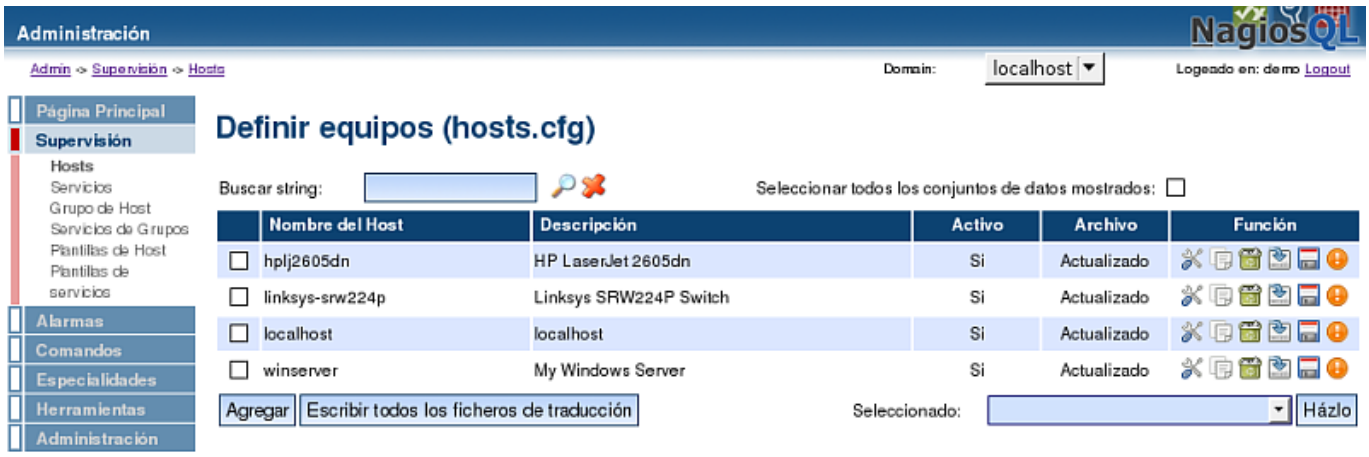
- Hacia su derecha podremos observar diferentes botones :

De izquierda a derecha

- El primer botón es para acceder a la información de configuración del host, con la correspondiente posibilidad de su posterior modificación.
- El segundo botón es para copiar y así duplicar la configuración de dicho host, en caso de que tengamos un equipo con similares características al realizar esto se nos hará mucho más fácil su implementación.
- El tercer botón es para generar la escritura en la configuración de Nagios para dicho host en caso de que hayamos cambiado algo.
- El cuarto botón es para descargados el archivo con la configuración individual de dicho equipo.
- El quinto y último botón es para mostrarnos información de las relaciones de configuración de dicho equipo, si es posible borralo y no afectar otras configuraciones mostrándonos un resumen como el siguiente:

Relación de información para **001** de la tabla**tbl_host:**
Relacionado con **tbl_contactgroup** entrada **admins** - borrar **posible**
Relacionado con **tbl_host** entrada **C001WS30** - borrar **posible**
Relacionado con **tbl_host** entrada **VDB_001** - borrar **posible**
Relacionado con **tbl_hostgroup** entrada **filiales** - borrar **posible**
Relacionado con **tbl_hostgroup** entrada **directo_red_interna** - borrar **posible**
Relacionado con **tbl_hostgroup** entrada **nocheck_estado** - borrar **posible**
Relacionado con **tbl_hosttemplate** entrada **filiales** - borrar **posible**
Relacionado con **tbl_service** entrada **ancho_ipsec0-Ancho de banda ipsec0** - borrar **posible**

Refiriéndonos al primer botón para proceder a editar un hosts existente o dar de alta uno nuevo, nos aparecera una pantalla como esta, donde podremos ir agregando los datos pertinentes del hosts en cuestion que necesitamos monitorear.



Al editar un host existente o dar de alta uno nuevo, nos aparecera una pantalla como esta, donde podremos ir agregando los datos pertinentes del host en cuestion que necesitamos monitorear.

A su vez deberemos utilizar una plantilla, en este caso la de generic-host o podemos utilizar otra con opciones y ajustes predefinidos para el tipo de monitoreo a utilizar para esta clase de host (intervalos de chequeo, notificaciones, alarmas,

dependencias, grupo de hosts).

- Como datos fundamentales tenemos:

Nombre del Host: *Aquí definiremos el nombre real o de configuración para nosotros, sin espacios, comas, guiones ú otros caracteres que no correspondan.*

Descripción: *Mínimo dato descriptivo de que función cumple el hosts, ejemplo 'srvmon0002 – Servidor de monitoreo'*

Dirección: *Dirección IP del equipo*

- Como datos adicionales tenemos:

Padres: *Podremos definir una lista de hosts de los que depende la ruta de llegada hacia ese determinado host dentro de la topología de red.*

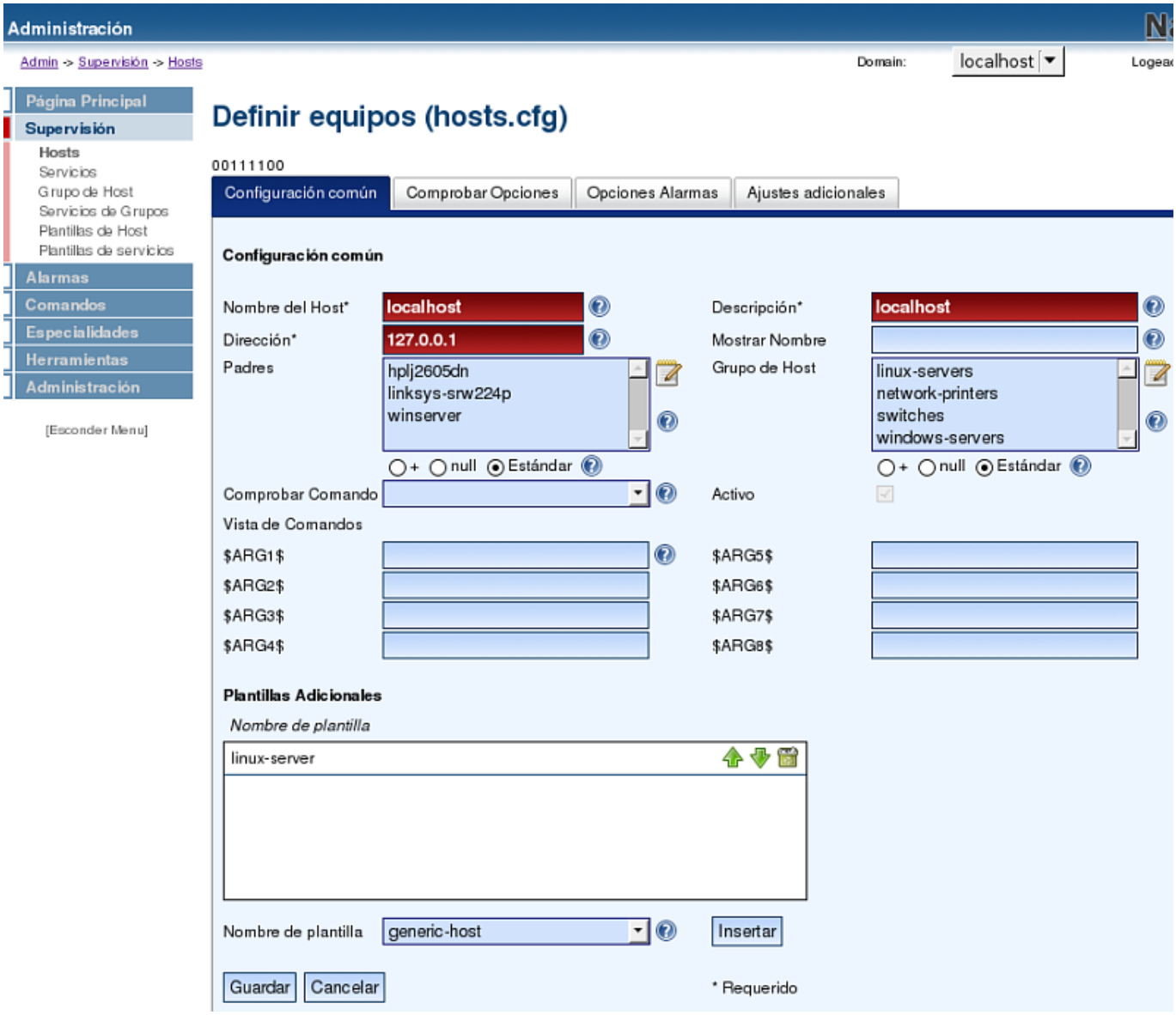
Grupo de Hosts: *Conjuntos a los cuales pertenece dicho hosts, igualmente si son muchos hosts de similares servicios que utilizan una misma plantilla, es mas practico definir este dato en ella.*

Comando de comprobación: *Comando para ejecutar el chequeo de comprobación del hosts, igualmente si son muchos hosts de similares servicios que utilizan una misma plantilla, es mas practico definir este dato en ella.*

Activo: *Podemos activar o desactivar de la configuración el equipo.*

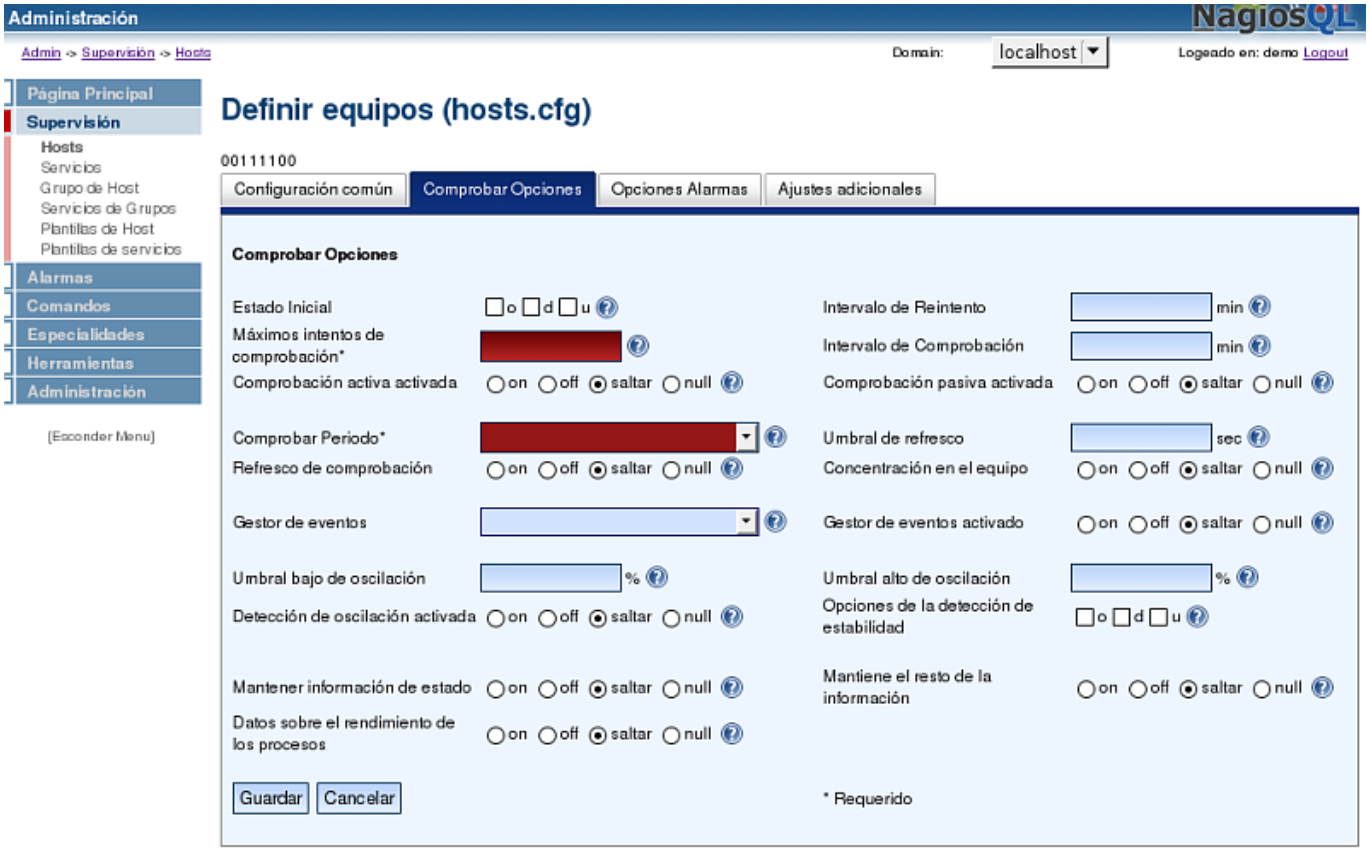
$ARG1$ - $ARG8$: *Lista de argumentos ordenados uno a uno para pasarle al comando de comprobación de dicho host.*

Plantillas: *Plantillas de las cuales depende el host para obtener determinados datos de configuración, dichos datos contenidos en estas se pueden omitir en la configuración del host ya que se desprenden de esta.*



Opciones de comprobación del Host

Aqui podemos ver las opciones generales del hosts, intervalos de chequeo, reintentos, periodo de chequeos etc. Donde dice ‘saltar’ es por que se deja esa opción a la determinada globalmente por la plantilla.



Estado inicial: *Por defecto Nagios al iniciar asume que el host esta activo, pero esto no necesariamente puede ser así, entonces podemos definirlo como:*

- O = UP
- D = DOWN
- U = UNREACHABLE

Máximos intentos de comprobación: *Máxima cantidad de intentos de comprobación antes de definir el estado final del host.*

Intervalo de reintento: *Intervalo de reintento entre cada intento de comprobación.*

Intervalo de comprobación: *Intervalo entre cada comprobación, incluidos sus reintentos.*

Comprobación Activa: *Es el tipo de chequeo por defecto y más usado por Nagios, ejecutado por sí mismo con un determinado margen de tiempo para su ejecución.*

Comprobación pasiva: *Es un chequeo ejecutado por aplicaciones externas y su resultado devuelto a Nagios para su procesamiento.*

Periodo de comprobaciones: *Son los lapsos de tiempo en los cuales se ejecutaran los chequeos de dicho host, por ejemplo las 24 horas del día los 7 días de la semana o solo horas laborales de Lunes a Viernes.*

Umbral de refresco: *Se utiliza en la implementación de chequeos pasivos y en entornos distribuidos para poner énfasis sobre el refresco de los datos de chequeo de un equipo.*

Refresco de comprobación: *Activar el refresco de chequeos.*

Concentración en el equipo: *Utilizar un comando luego de ejecutado el gestor de eventos.*

Gestor de eventos: *Comando a ejecutarse luego de obtenidos los resultados del chequeo del host.*

Gesto de eventos activado: *Aquí definiremos si utilizaremos o no un gestor de eventos luego de los chequeos.*

Umbral bajo de oscilación: *Aquí definimos el umbral mínimo para considerar una detección correcta del flapping de estado (cambios de estados muy repentinos)*

Umbral alto de oscilación: *Aquí definimos el umbral máximo para considerar una detección correcta del flapping de estado (cambios de estados muy repentinos)*

Detección de oscilación activada: *Activar o desactivar la detección de oscilación para el host.*

Opciones de detección de estabilidad: *Aquí definimos que estado consideramos para establecer el flapping de un host.*

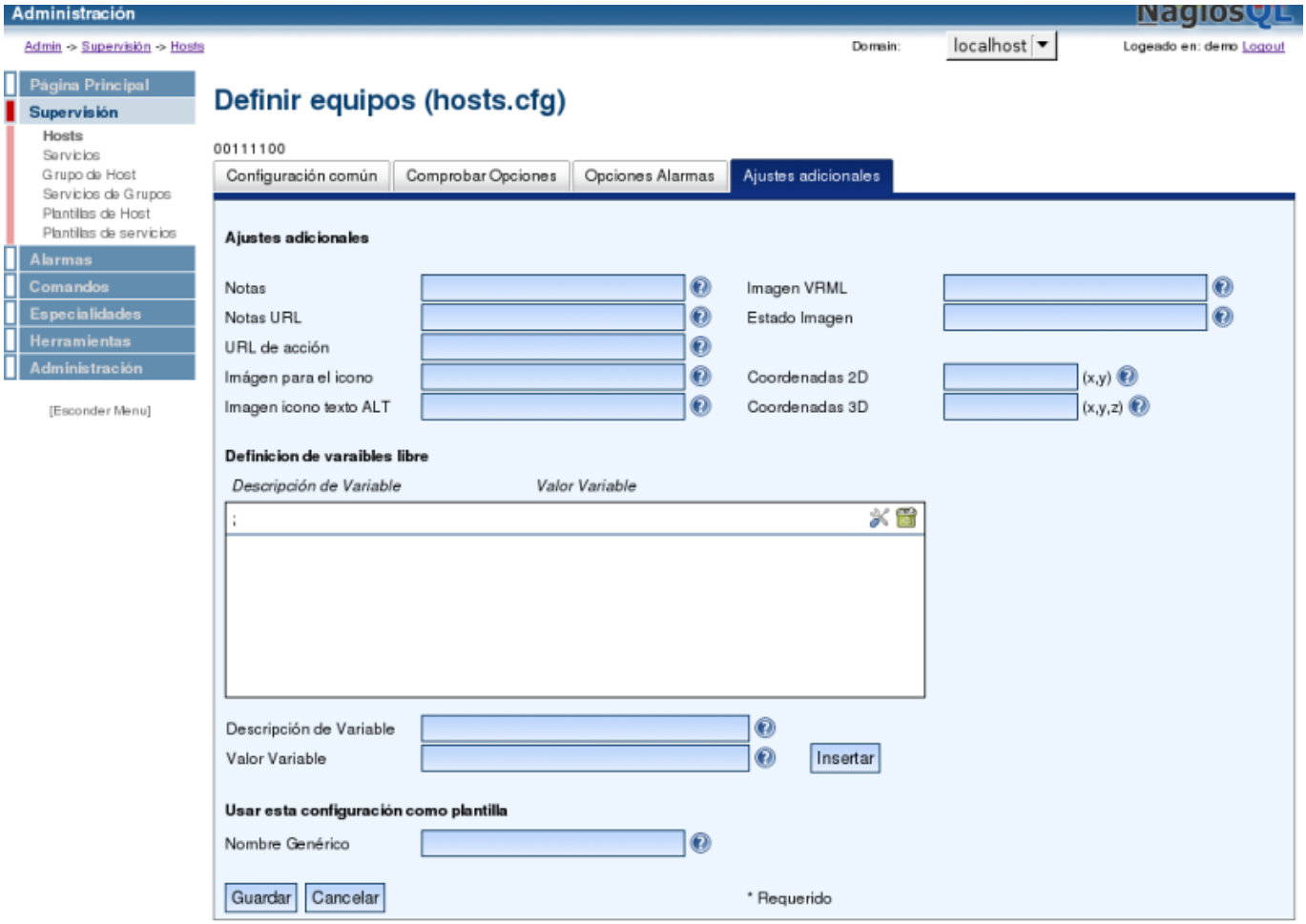
Mantener información de estado: *En caso de que reinicie Nagios, mantener la última información de estadística obtenida del chequeo del host*

Mantener el resto de la información: *Mantener otra información útil sobre el host entre los reinicios de Nagios.*

Datos sobre el rendimiento de los procesos: *Procesar la información de performance de los chequeos, esto es útil ya que varios plugins de chequeo y otras utilidades utilizan esta información para colección de datos.*

Opciones de alarmas del host

Desde este apartado podremos estableces las opciones de cómo llegaran las notificaciones del estado del host a personal encargado de recibirlas.



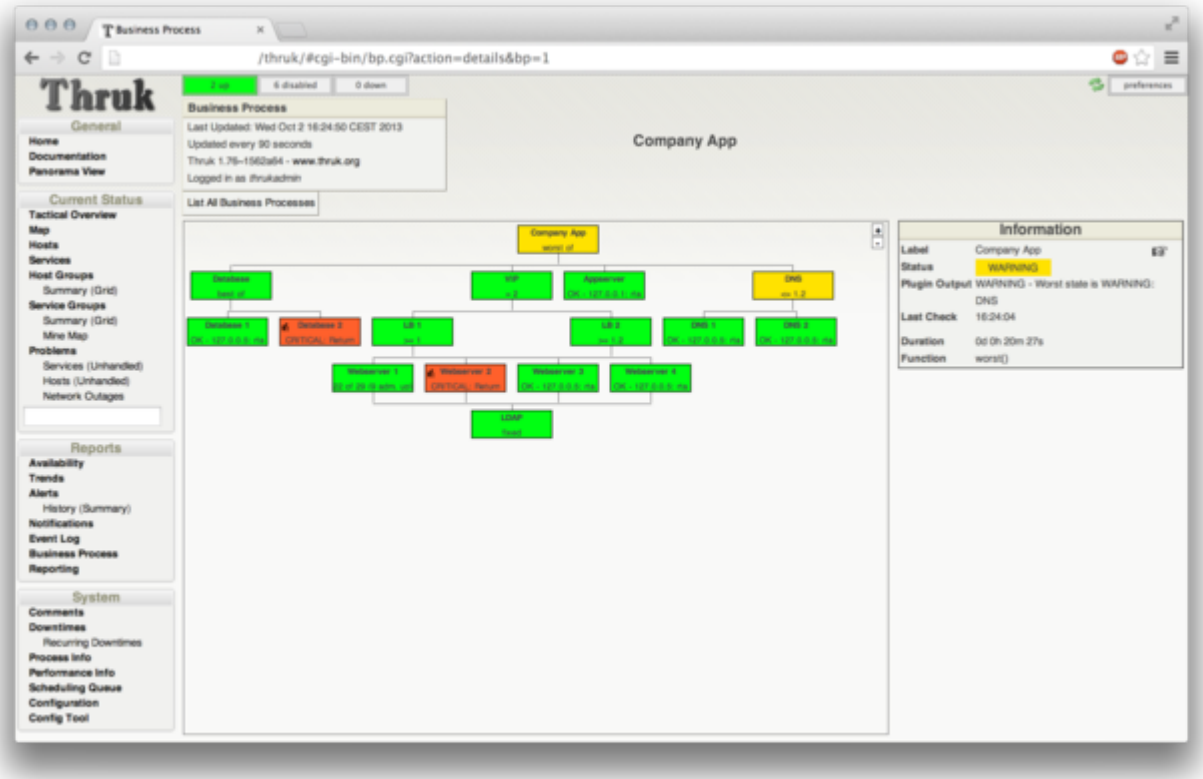
Interfaz administrativa

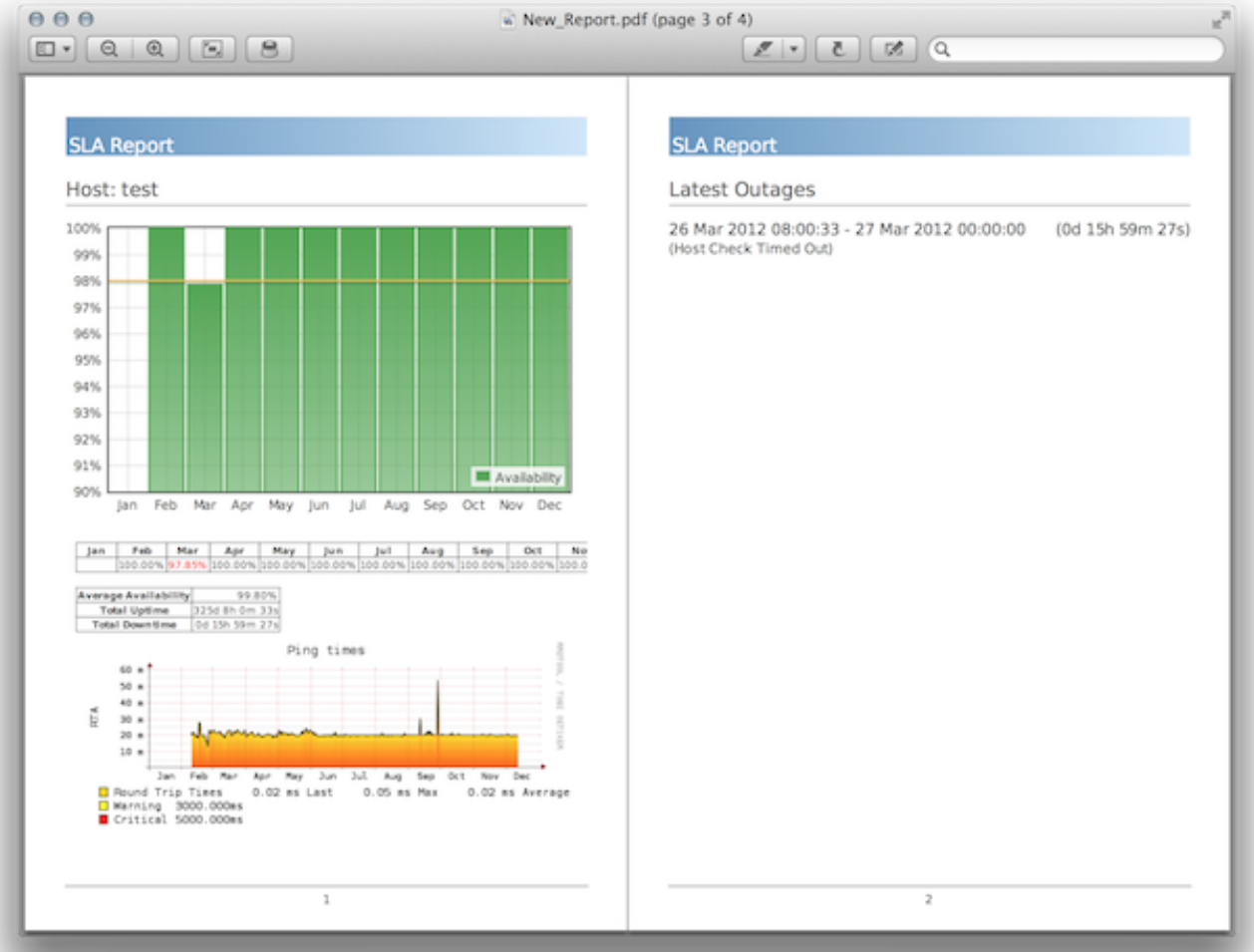
NOTA PERSONAL: Desde hace algunos años no recomiendo usar NagiosQL, en su lugar recomiendo tener los archivos cfg, con la menor cantidad de líneas posibles y siempre utilizar templates para todo.

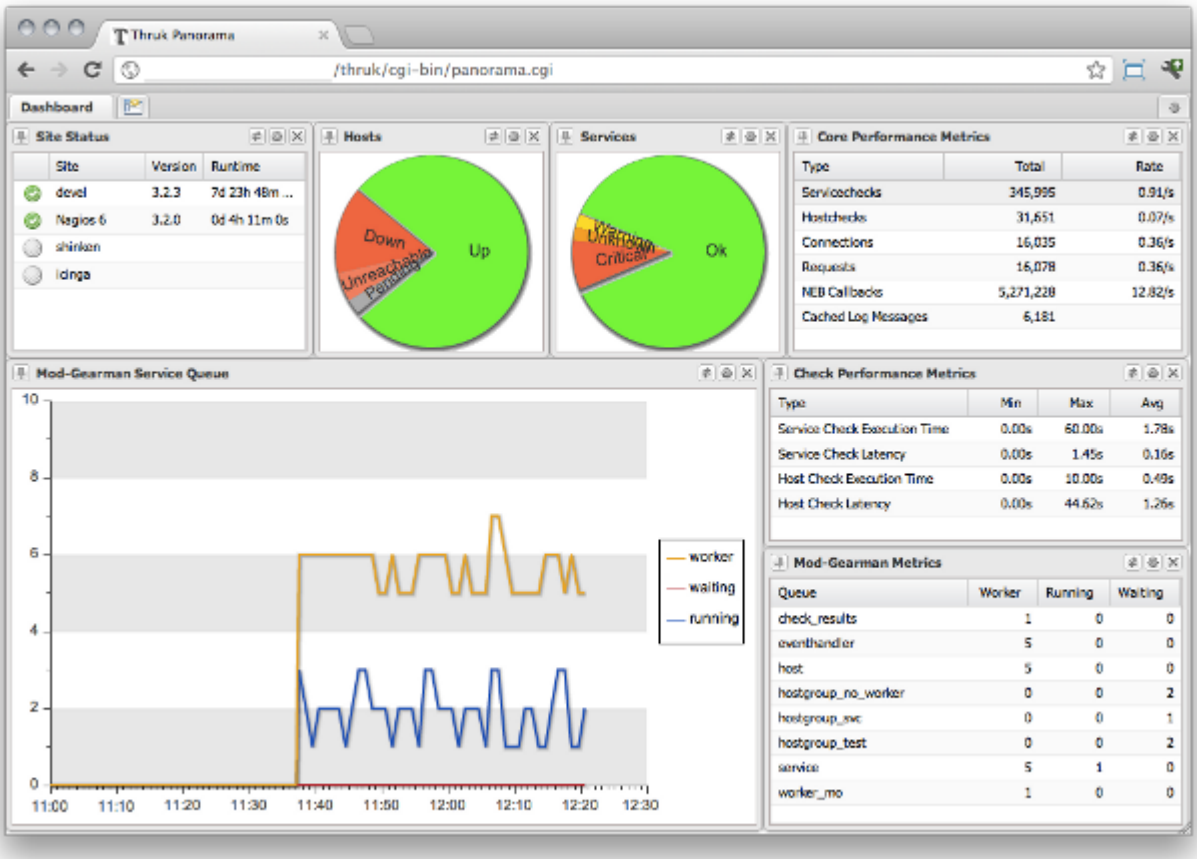
Interfaces alternativas

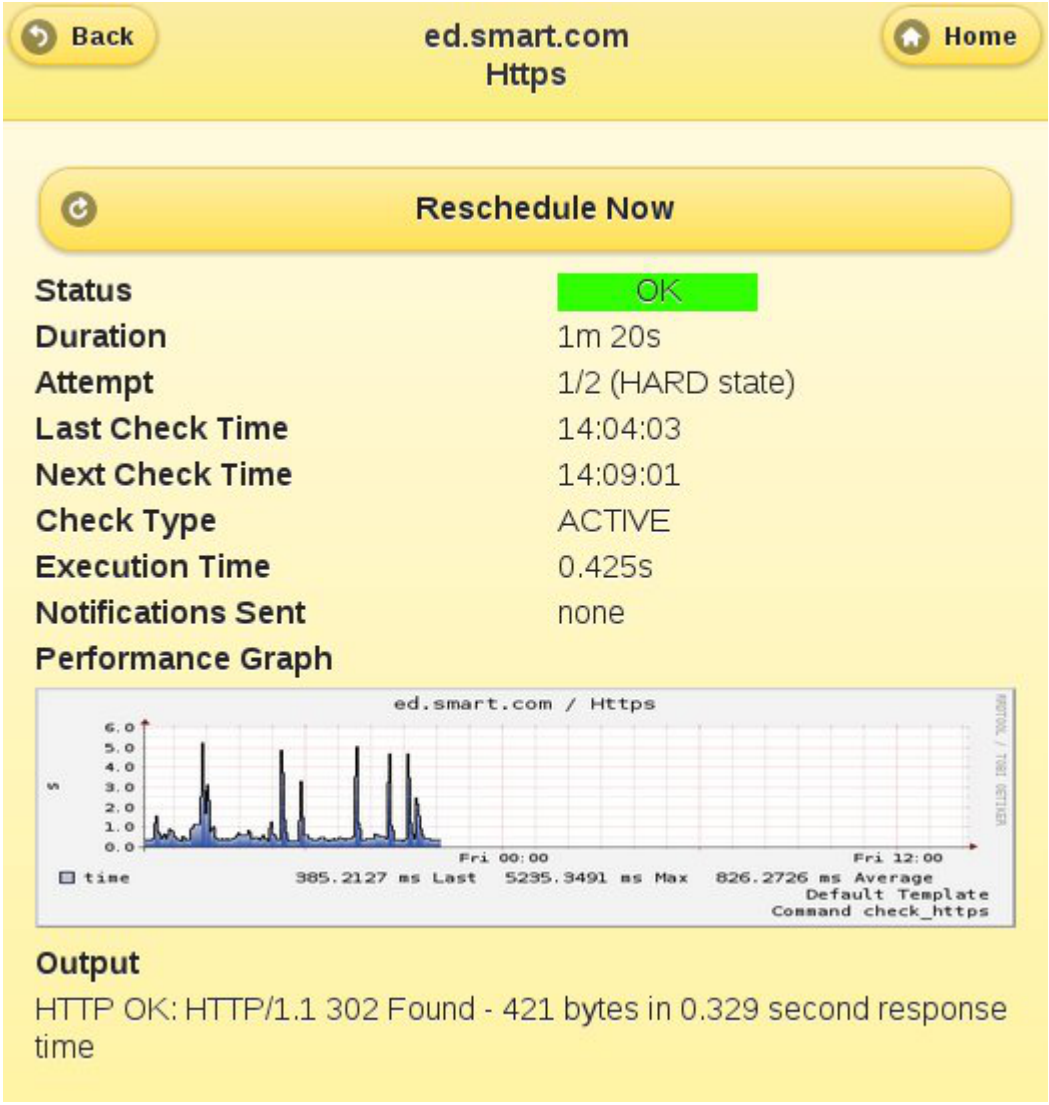
Thruk

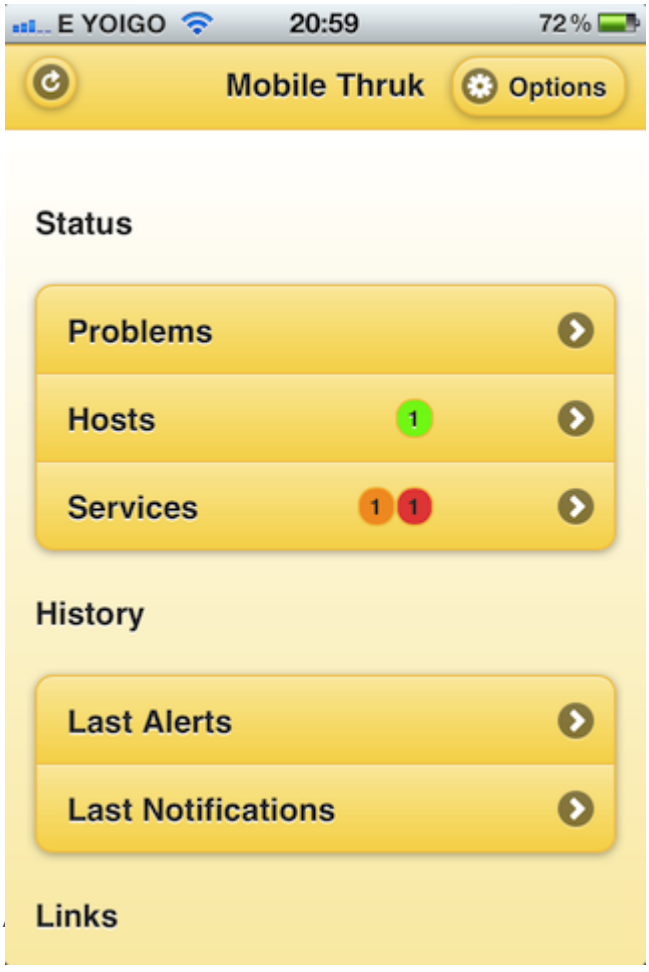
Thruk es una interfaz web para visualizar y explotar datos de Sistemas de Monitoreo, actualmente soporta Nagios, Icinga y Shinken. Añade configuracion via web, reportes SLA y una interfaz para dispositivos móviles.


Business Process
/thruk/#cgi-bin/bp.cgi?action=details&bp=1
Thruk
Business Process
Company App
General
Home
Documentation
Panorama View
Current Status
Tactical Overview
Map
Hosts
Services
Host Groups
Summary (Grid)
Service Groups
Summary (Grid)
Mine Map
Problems
Services (Unhandled)
Hosts (Unhandled)
Network Outages
Reports
Availability
Trends
Alerts
History (Summary)
Notifications
Event Log
Business Process
Reporting
System
Comments
Downtimes
Recurring Downtimes
Process Info
Performance Info
Scheduling Queue
Configuration
Config Tool
Information
Label
Company App
Status
WARNING
Plugin Output
WARNING - Worst state is WARNING:
DNS
Last Check
16:24:04
Duration
0d 0h 20m 27s
Function
worst()

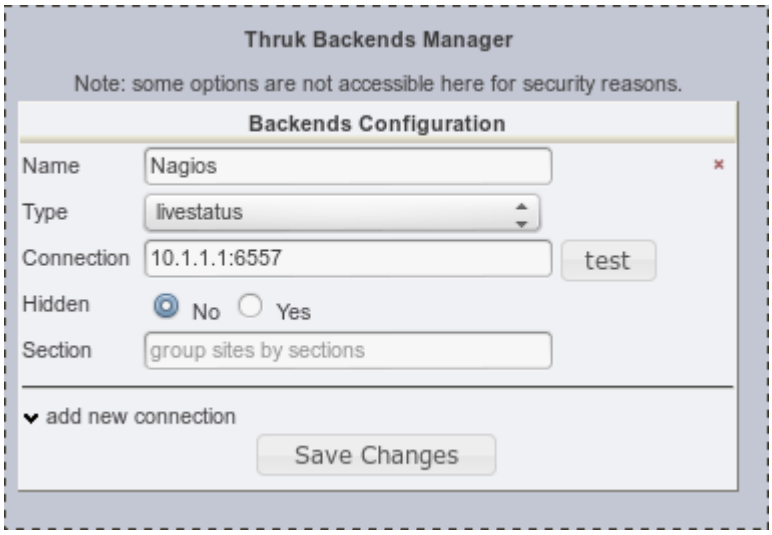

New_Report.pdf (page 3 of 4)
SLA Report
Host: test
100%
99%
98%
97%
96%
95%
94%
93%
92%
91%
90%
Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
Availability
Average Availability 99.80%
Total Uptime 325d 8h 0m 33s
Total Downtime 0d 15h 59m 27s
Ping times
RTA
Round Trip Times 0.02 ms Last 0.05 ms Max 0.02 ms Average
Warning 3000.000ms
Critical 5000.000ms
1
SLA Report
Latest Outages
26 Mar 2012 08:00:33 - 27 Mar 2012 00:00:00 (0d 15h 59m 27s)
(Host Check Timed Out)
2


Thruk Panorama
/thruk/cgi-bin/panorama.cgi
Dashboard
Site Status
Site
Version
Runtime
devel
3.2.3
7d 23h 48m ...
Nagios 6
3.2.0
0d 4h 11m 0s
shinken
icinga
Hosts
Down
Unreachable
Pending
Up
Services
Warning
Unknown
Critical
Ok
Core Performance Metrics
Type
Total
Rate
Servicechecks
345,995
0.91/s
Hostchecks
31,651
0.07/s
Connections
16,035
0.36/s
Requests
16,078
0.36/s
NEB Callbacks
5,271,228
12.82/s
Cached Log Messages
6,181
Mod-Gearman Service Queue
worker
waiting
running
Check Performance Metrics
Type
Min
Max
Avg
Service Check Execution Time
0.00s
60.00s
1.78s
Service Check Latency
0.00s
1.45s
0.16s
Host Check Execution Time
0.00s
10.00s
0.49s
Host Check Latency
0.00s
44.62s
1.26s
Mod-Gearman Metrics
Queue
Worker
Running
Waiting
check_results
1
0
0
eventhandler
5
0
0
host
5
0
0
hostgroup_no_worker
0
0
2
hostgroup_sw
0
0
1
hostgroup_test
0
0
2
service
5
1
0
worker_mo
1
0
0


Back
ed.smart.com
Https
Home
Reschedule Now
Status
OK
Duration
1m 20s
Attempt
1/2 (HARD state)
Last Check Time
14:04:03
Next Check Time
14:09:01
Check Type
ACTIVE
Execution Time
0.425s
Notifications Sent
none
Performance Graph
ed.smart.com / Https
6.0
5.0
4.0
3.0
2.0
1.0
0.0
Fri 00:00
Fri 12:00
time
385.2127 ms Last
5235.3491 ms Max
826.2726 ms Average
Default Template
Command check_https
Output
HTTP OK: HTTP/1.1 302 Found - 421 bytes in 0.329 second response time


E YOIGO
20:59
72%
Mobile Thruk
Options
Status
Problems
Hosts
1
Services
1
1
History
Last Alerts
Last Notifications
Links

Podemos tener múltiples instancias de Nagios remotas y visualizarlas todas en una única instancia de Thruk



Plugins

Plugins SNMP, caracteristicas generales

Script detail page	Description	performance	Supported platforms
check_snmp_storage	checks storages (disks,swap, memory, etc...)	Yes	All MIB2 compliant
check_snmp_int	checks interface states, usage on hosts, switch, routers, etc	Yes	All MIB2 compliant
check_snmp_process	checks if process are running, the number that are running, memory and cpu used.	No	All MIB2 compliant
check_snmp_load	checks the load or the cpu of a machine	Yes	Linux, Windows, Cisco, AS400, HP Procurve, LinkProof, Blucoat, Nokia,Fortinet, Netscreen,HPUX.
check_snmp_vrrp	checks the interface state of vrrp cluster	No	Nokia IP (VRRP & Clustering), LinkProof, Alteon
check_snmp_cpfw	checks Checkpoint Firewall1 status	Yes	Checkpoint Firewall
check_snmp_mem	Checks memory and swap usage	Yes	Linux/Netsnmp, Cisco, HP Switch
check_snmp_win	Checks windows services	No	Windows
check_snmp_css	Checks css services	No	CSS

	state		
check_snmp_env	Checks environemental status (fan, temp, power supply).	No	Cisco, Nokia, Bluecoat, IronPort, Foundry
check_snmp_nsbox	Checks nsbox vhost & diode status.	No	NetSecureOne Netbox
check_snmp_boostedge	Checks Boostedge services	No	Boostedge
check_snmp_linkproof_nhr	Checks linkproof NHR	No	Radware Linkproof

Comandos default en Nagios

El sistema Nagios incorpora un gran número de comandos por default que se pueden utilizar. Algunos de ellos hacen uso de librerías y paquetes que deben estar instalados en el sistema. En las siguientes líneas se comentan cuales son los comandos que vienen por default y que hacen.

check_by_ssh

Este comando es muy interesante interesante. Permite ejecutar comandos en ordenadores remotos vía SSH (de forma segura, por tanto). El resultado de ese comando será tomado por Nagios.

Uso : check_by_ssh -H <host> -C <command> [-fqv] [-1|-2] [-4|-6]

```
    [-S [lines]] [-E [lines]] [-t timeout] [-i identity]
    [-l user] [-n name] [-s servicelist] [-O outputfile]
    [-p port] [-o ssh-option]
```

Opciones: -h, –help

```
  Imprimir ayuda detallada
```

-V, –version

```
  Imprimir informacion de version
```

-H, –hostname=ADDRESS

```
  Hostname, direccion IP, or socket unix (path completo en este caso)
```

-p, –port=INTEGER

```
  Puerto a chequear
```

-4, –use-ipv4

```
  Usar IPv4
```

-6, –use-ipv6

```
  Usar IPv6
```

-1, –proto1

```
  Usar protocolo 1 de SSH [optional]
```

-2, –proto2

```
  Usar protocolo 2 de SSH [optional]
```

-S, –skip-stdout[=n]

```
  Ignora todas o las primeras n lineas de STDOUT [optional]
```

-E, –skip-stderr[=n]

Ignora todas o las primeras n lineas de STDERR [optional]

-f

Decirle al SSH que realize un fork en vez de una tty [optional]. Siempre devolver OK si ssh es ejecutado

-C, –command='COMMAND STRING'

Comando a ejecutar en la maquina remota

-l, –logname=USERNAME

Nombre de usuario SSH [optional]

-i, –identity=KEYFILE

Llave SSH autorizada [optional]

-O, –output=FILE

archivo de comando externo de Nagios [optional]

-s, –services=LIST

lista de nombres de servicios Nagios separados por ':' [optional]

-n, –name=NAME

nombre corto del host en la configuracion de Nagios [optional]

-o, –ssh-option=OPTION

llamar ssh con la opcion '-o' (puede ser usada multiples veces) [optional]

-q, –quiet

Tell ssh to suppress warning and diagnostic messages [optional]

-w, –warning=DOUBLE

Response time to result in warning status (seconds)

-c, –critical=DOUBLE

Response time to result in critical status (seconds)

-t, –timeout=INTEGER

Seconds before connection times out (default: 10)

-v, –verbose

Mostrar detalles de la linea de comandos para debug (Nagios truncara la salida)

La forma mas comun de uso, es por medio de una llave ssh con el argumento '-i'. De este modo el par de llaves deben tener una contraseña nula y la llave publica debe estar listada en el archivo authorized_keys del host remoto. Usualmente la llave debe estar restringida para correr solo un comando en el servidor remoto. Si el script remoto permite la adicion de argumentos, puede actuar como una agente al estilo proxy para ejecutar otros comandos remotos.

Ejemplo de como configurarlo como **comando**

```
define command {
 command_name    check_ssh_load
 command_line    $USER1$/check_by_ssh -H $HOSTADDRESS$ -C "/user/bin/check_load -w
$ARG1$ -c $ARG2$"
}
```

Ejemplo de como configurarlo en un **servicio**

```
define service {
 use                 local-service
```

```
 hostgroup_name      ssh-nagios-services
 service_description Current Load
 check_command       check_ssh_load!5.0,4.0,3.0!10.0,6.0,4.0
}
```

Ejemplo de como dar de alta el **hostgroup** de ejecucion

```
define hostgroup {
 hostgroup_name  ssh-nagios-services
 alias           Nagios over SSH
 members         remote1,remote2
}
```

check_dig

Este comando sirve para comprobar el funcionamiento del servicio de DNS en un equipo remoto. Utiliza dig para esto.

Usage:check_dig -l <query_address> [-H <host>] [-p <server port>] [-T <query type>] [-w <warning interval>] [-c <critical interval>] [-t <timeout>] [-a <expected answer address>] [-v]

Ejemplos: Esto enviara una consulta TCP al servidor DNS consultando por el nombre www.example.com

```
check_dig -H DNSSERVER -l www.example.com -A "+tcp"
```

Definir como comando:

```
define command{
    command_name    check_dig
    command_line    $USER1$/check_dig -H '$HOSTADDRESS$' -l '$ARG1$'
}
```

Definir como servicio:

```
define service{
    use                     generic-service
    host_name               nombre_host
    service_description     dig
    check_command           check_dig!www.google.com
}
```

check_disk

Este comando sirve para comprobar el espacio libre de un volumen montado en el sistema de ficheros donde se esté ejecutando Nagios. Permite especificar dos umbrales y generar disparadores advertencias cuando se supera el menor, y errores críticos cuando se supera el segundo.

Usage: check_disk -w limit -c limit [-W limit] [-K limit] {-p path | -x device} [-C] [-E] [-e] [-g group] [-k] [-l] [-M] [-m] [-R path] [-r path] [-t timeout] [-u unit] [-v] [-X type]

Ejemplos: check_disk -w 10% -c 5% -p /tmp -p /var -C -w 100000 -c 50000 -p /

```
  Checks /tmp and /var at 10% and 5%, and / at 100MB and 50MB
```

check_disk -w 100M -c 50M -C -w 1000M -c 500M -g sidDATA -r '^/oracle/SID/data.*$'

Checks all filesystems not matching -r at 100M and 50M. The fs matching the -r regex
are grouped which means the freespace thresholds are applied to all disks together

check_disk -w 100M -c 50M -C -w 1000M -c 500M -p /foo -C -w 5% -c 3% -p /bar

Checks /foo for 1000M/500M and /bar for 5/3%. All remaining volumes use 100M/50M

Definir como comando:

```
define command{
    command_name    check_all_disks
    command_line    $USER1$/check_disk -w '$ARG1$' -c '$ARG2$'
}
```

Definir como servicio

```
define service{
    use                     generic-service
    host_name               nombre_host
    service_description     Disk Space
    check_command           check_all_disks!20%!10%
}
```

check_disk_smb

Planificación, especificación, diseño y evaluación de redes Este comando funciona exactamente igual que check_disk pero realiza la comprobación utilizando samba para realizar la comprobación de volúmenes compartidos en quipos remotos, en redes Windows.

Perl Check SMB Disk plugin for Nagios

Usage: check_disk_smb -H <host> -s <share> -u <user> -p <password>

1. w <warn> -c <crit> [-W <workgroup>] [-P <port>]

-H, –hostname=HOST

Nombre NetBIOS del servidor

-s, –share=STRING

Recurso compartido a testear

-W, –workgroup=STRING

Grupo de trabajo o dominio usado (Default "WORKGROUP")

-u, –user=STRING

Usuario a loguearse en el servidor. (Defaults "guest")

-p, –password=STRING

Contraseña para loguearse en el servidor. (Defaults NULL)

-w, –warning=INTEGER or INTEGER[kMG]

Percent of used space at which a warning will be generated (Default: 85%)

-c, –critical=INTEGER or INTEGER[kMG]

Percent of used space at which a critical will be generated (Defaults: 95%)

-P, –port=INTEGER

Port to be used to connect to. Some Windows boxes use 139, others 445 (Defaults to smbclient default)

If thresholds are followed by either a k, M, or G then check to see if that

```
 much disk space is available (kilobytes, Megabytes, Gigabytes)
Porcentaje WARNING debe ser menor al porcentaje CRITICAL
 Warning (remaining) disk space should be greater than critical.
```

check_dns

Este comando permite hacer una consulta DNS para averiguar la dirección IP de un equipo dado el nombre o viceversa. Utiliza nslookup para ello; permite especificar el servidor DNS a usar o si no usa el o los especificados en /etc/resolv.conf.

Usage:check_dns -H host [-s server] [-a expected-address] [-A] [-t timeout] [-w warn] [-c crit]

check_dummy

Este comando permite realizar una consulta a un dispositivo ficticio (devuelve el mismo parámetro que se le pasa). Puede ser utilizado para comprobaciones y depuraciones.

Usage: check_dummy <estado numerico> [optional text]

check_flexlm

Este comando comprueba el funcionamiento de un sistema FlexLM. Este sistema es un servidor de licencias en red usado para obtener permisos de uso de software en red. Devuelve distintos errores dependiendo del estado de estos servidores de licencias.

check_ftp

Este comando realiza comprobaciones de conexión a un servidor FTP remoto. Permite conocer el estado de este servicio.

This plugin tests FTP connections with the specified host (or unix socket).

Usage:check_ftp -H host -p port [-w <warning time>] [-c <critical time>] [-s <send string>] [-e <expect string>] [-q <quit string>][-m <maximum bytes>] [-d <delay>] [-t <timeout seconds>] [-r <refuse state>] [-M <mismatch state>] [-v] [-4|-6] [-j] [-D <days to cert expiry>] [-S <use SSL>] [-E]

check_http

Este comando comprueba servicios HTTP y HTTPS en equipos remotos. Permite además realizar el seguimiento de redirecciones, tiempos de conexión, la expiración de los certificados para SSL, etcétera. Es especialmente útil para servidores web que sirvan de base para aplicaciones de comercio electrónico.

Usage: check_http -H <vhost> | -I <IP-address> [-u <uri>] [-p <port>]

```
    [-w <warn time>] [-c <critical time>] [-t <timeout>] [-L]
    [-a auth] [-f <ok | warn | critcal | follow>] [-e <expect>]
    [-s string] [-l] [-r <regex> | -R <case-insensitive regex>] [-P string]
    [-m <min_pg_size>:<max_pg_size>] [-4|-6] [-N] [-M <age>] [-A string]
    [-k string] [-S] [-C <age>] [-T <content-type>]
```

Ejemplos: CHECK CONTENT: check_http -w 5 -c 10 –ssl -H www.verisign.com

When the 'www.verisign.com' server returns its content within 5 seconds, a STATE_OK will be returned. When the server returns its content but exceeds the 5-second threshold, a STATE_WARNING will be returned. When an error occurs, a STATE_CRITICAL will be returned.

CHECK CERTIFICATE: check_http -H www.verisign.com -C 14

When the certificate of 'www.verisign.com' is valid for more than 14 days, a STATE_OK is returned. When the certificate is still valid, but for less than 14 days, a STATE_WARNING is returned. A STATE_CRITICAL will be returned when the certificate is expired.

Ejemplo desde un script

```
#!/bin/sh
/usr/local/nagios/libexec/check_http -H $1 -p 50000 -u /irj/portal  -A "Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0)" -k "Accept-Language: es-ES" -s "Bienvenido"
```

check_ifoperstatus

Este comando comprueba el estado de operación de interfaces de red remotas por medio de SNMP v1 o SNMP v3.

check_ifstatus

Este comando comprueba el estado general de interfaces de red remotas por medio de SNMP v1 o SNMP v3.

check_imap

Este comando realiza conexiones contra un servidor IMAP para comprobar su estado de funcionamiento. Permite generar advertencias y errores críticos.

Definir como comando :

```
define command{
    command_name    check_imap
    command_line    $USER1$/check_imap -H '$HOSTADDRESS$'
}
```

Chequear IMAP con SSL

```
define command {
    command_name    check_simap
    command_line    $USER!$/plugins/check_imap -p 993 -H '$HOSTADDRESS$' -S
}
```

Definir como servicio:

```
define service{
    use                     generic-service
    host_name               nombre_host
    service_description     imaps
    check_command           check_simap
}
```

check_ircd

Este comando comprueba el funcionamiento de un servidor de IRC remoto. Realiza conexiones para ello, esta escrito en Perl.

check_ldap

Este comando realiza conexiones y búsquedas LDAP contra un servidor remoto y comprueba así su estado de funcionamiento y si responde dentro del tiempo

esperado o no.

Usage: check_ldap -H <host> -b <base_dn> [-p <port>] [-a <attr>] [-D <binddn>]

```
     [-P <password>] [-w <warn_time>] [-c <crit_time>] [-t timeout]
     [-2|-3] [-4|-6]
```

Notes: If this plugin is called via 'check_ldaps', method 'STARTTLS' will be implied (using default port 389) unless –port=636 is specified. In that case 'SSL on connect' will be used no matter how the plugin was called. This detection is deprecated, please use 'check_ldap' with the '–starttls' or '–ssl' flags to define the behaviour explicitly instead.

check_load

Este comando trabaja en local en la máquina que está ejecutando el sistema Nagios. Comprueba la carga del sistema en función de unos umbrales que tiene preestablecidos y permite generar advertencias o errores severos según sea esta carga.

Uso:check_load [-r] -w WLOAD1,WLOAD5,WLOAD15 -c CLOAD1,CLOAD5,CLOAD15

-r, –percpu

```
  Divide la carga por el numero de CPU's siempre que fuera posible
   Cambia a WARNING si el promedio de carga excede “WLOADn”
   Cambia a CRITICAL si el promedio de carga excede “CLOADn”
```

Definir como comando :

```
define command{
    command_name    check_load
    command_line    $USER1$/check_load --warning=$ARG1$,$ARG2$,$ARG3$
--critical=$ARG4$,$ARG5$,$ARG6$
}


define service{
    use                         generic-service
    host_name                   nombre_host
    service_description         Current Load
    check_command               check_load!5.0!4.0!3.0!10.0!6.0!4.0
}
```

check_log

Este comando es muy interesante para administradores del sistema. Funciona en local y permite buscar coincidencia de patrones en ficheros de suceso. Cuando el patrón que se busca es encontrado, Nagios recoge esta incidencia.

Usage: check_log -F logfile -O oldlog -q query Usage: check_log –help Usage: check_log –version

check_mailq

Este comando funciona en local en la máquina que corre Nagios. Permite comprobar el número de mensajes que hay en espera en las colas de Sendmail. Se puede establecer un límite para que se genere una notificación en ese caso.

```
Usage: check_mailq -w <warn> -c <crit> [-W <warn>] [-C <crit>] [-M <MTA>] [-t
<timeout>] [-v verbose]

   Checks the number of messages in the mail queue (supports multiple sendmail
```

```
queues, qmail)
   Feedback/patches to support non-sendmail mailqueue welcome

-w (--warning)   = Min. number of messages in queue to generate warning
-c (--critical)  = Min. number of messages in queu to generate critical alert ( w <
c )
-W (--Warning)   = Min. number of messages for same domain in queue to generate
warning
-C (--Critical)  = Min. number of messages for same domain in queue to generate
critical alert ( W < C )
-t (--timeout)   = Plugin timeout in seconds (default = 15)
-M (--mailserver) = [ sendmail | qmail | postfix | exim ] (default = sendmail)
-h (--help)
-V (--version)
-v (--verbose)   = debugging output


Note: -w and -c are required arguments.  -W and -C are optional.
 -W and -C are applied to domains listed on the queues - both FROM and TO.
(sendmail)
 -W and -C are applied message not yet preproccessed. (qmail)
 This plugin uses the system mailq command (sendmail) or qmail-stat (qmail)
 to look at the queues. Mailq can usually only be accessed by root or
 a TrustedUser. You will have to set appropriate permissions for the plugin to work.
```

check_mrtg

Este comando también trabaja en local en la máquina que está ejecutando Nagios y permite monitorizar los ficheros de sucesos de MRTG; en concreto permite monitorizar cualquiera de los parámetros que se vuelcan sobre dichos ficheros como por ejemplo conexiones, carga del procesador, entrada, salida, etcétera. Permite establecer umbrales que si se superan generan notificaciones.

This plugin will check either the average or maximum value of one of the two variables recorded in an MRTG log file.

Usage:check_mrtg -F log_file -a <AVG | MAX> -v variable -w warning -c critical [-l label] [-u units] [-e expire_minutes] [-t timeout] [-v]

check_mrtgtraf

Este comando permite comprobar el servicio UPS en un equipo remoto y establecer umbrales para, según el valor devuelto, disparar una advertencia, un error severo o nada.

This plugin will check the incoming/outgoing transfer rates of a router, switch, etc recorded in an MRTG log. If the newest log entry is older than <expire_minutes>, a WARNING status is returned. If either the incoming or outgoing rates exceed the <icl> or <ocl> thresholds (in Bytes/sec), a CRITICAL status results. If either of the rates exceed the <iwl> or <owl> thresholds (in Bytes/sec), a WARNING status results.

Usage check_mrtgtraf -F <log_file> -a <AVG | MAX> -v <variable> -w <warning_pair>-c <critical_pair> [-e expire_minutes] [-t timeout] [-v]

Options: -h, –help

```
  Print detailed help screen
```

-V, –version

```
  Print version information
```

-F, –filename=STRING

File to read log from

-e, –expires=INTEGER

Minutes after which log expires

-a, –aggregation=(AVG|MAX)

Test average or maximum

-w, –warning

Warning threshold pair <incoming>,<outgoing>

-c, –critical

Critical threshold pair <incoming>,<outgoing>

Notes: - MRTG stands for Multi Router Traffic Grapher. It can be downloaded from

http://ee-staff.ethz.ch/~oetiker/webtools/mrtg/mrtg.html

- While MRTG can monitor things other than traffic rates, this

plugin probably won't work with much else without modification.

- The calculated i/o rates are a little off from what MRTG actually

reports. I'm not sure why this is right now, but will look into it
for future enhancements of this plugin.

check_nagios

Este comando se ejecuta en la máquina que está ejecutando Nagios y permite comprobar que el archivo de sucesos del sistema de monitorización no sea más antiguo de lo que se especifique.

Usage:check_nagios -F <status log file> -e <expire_minutes> -C <process_string>

Options: -h, –help

Print detailed help screen

-V, –version

Print version information

-F, –filename=FILE

Name of the log file to check

-e, –expires=INTEGER

Minutes aging after which logfile is considered stale

-C, –command=STRING

Substring to search for in process arguments

-v, –verbose

Show details for command-line debugging (Nagios may truncate output)

Examples: check_nagios -e 5 -F /usr/local/nagios/var/status.log -C /usr/local/nagios/bin/nagios

negate

Este comando sirve para, en combinación con cualquiera de los otros plugins, negar su valor. Por ejemplo, el uso normal del comando check_ftp es que devuelve OK cuando el servicio esté funcionando y CRITICAL cuando no. Con este comando se invierten los valores. Es útil para cuando se desea tener notificación explícita de

que algo está funcionando bien en lugar de cuando falla.

Usage:negate [-t timeout] [-owcu STATE] [-s] <definition of wrapped plugin>

check_nntp

Este comando establece conexiones NNTP contra un servidor remoto especificado para comprobar que el servicio de NEWS esté activo.

This plugin tests NNTP connections with the specified host (or unix socket).

Usage:check_nntp -H host -p port [-w <warning time>] [-c <critical time>] [-s <send string>] [-e <expect string>] [-q <quit string>][-m <maximum bytes>] [-d <delay>] [-t <timeout seconds>] [-r <refuse state>] [-M <mismatch state>] [-v] [-4|-6] [-j] [-D <days to cert expiry>] [-S <use SSL>] [-E]

check_nt

Este comando realiza peticiones a un equipo Windows NT/2000/XP remoto que esté ejecutando el servicio NSClient para comprobar parámetros locales a dicho equipo como por ejemplo uso de la CPU, de la memoria, del disco, etcétera.

check_ntp

Este comando ejecuta ntpdate para comprobar que el timestamp de la máquina local que ejecuta Nagios no difiere en más de lo especificado del timestamp de una máquina remota dado.

check_nwstat

Planificación, especificación, diseño y evaluación de redes Este comando realiza peticiones a un equipo Novell remoto que esté ejecutando el servicio MRTGEXT NLM para comprobar paráme tros locales a dicho equipo como por ejemplo uso de la CPU, de la memoria, del disco, etcétera.

check_oracle

Este comando permite comprobar el estado de un SGBD Oracle en un ordenador remoto así como el estado de los tablespaces, de bases de datos, de las caché, etcétera, de dicho servidor.

check_overcr

Este comando permite comprobar el estado de un servicio Over-CR ejecutándose en un sistema UNIX remoto. Realiza peticiones a este servicio para comprobar su estado.

check_pop

Este comando comprueba si el servicio POP de un equipo remoto está funcionando correctamente. Realiza peticiones para ello.

check_procs

Este comando funciona en la máquina donde se está ejecutando Nagios. Comprueba el número de procesos que se están ejecutando en la máquina y genera advertencias cuando este número sobrepasa el umbral especificado.

check_real

Este comando comprueba si el servicio REAL de un equipo remoto está funcionando correctamente. Realiza peticiones para ello.

check_rpc

Este comando comprueba si un servicio RPC remoto está registrado y funcionando correctamente. Utiliza para ello llamadas a rpcinfo.

check_sensors

Este comando funciona en la máquina local donde se ejecuta Nagios; necesita de paquetes adicionales instalados en el sistema de monitorización y su función es comprobar el estado del hardware de la máquina.

check_smtp

Este comando permite conocer el estado de un servicio SNMP de una máquina remota. Realiza conexiones a este servicio para averiguar la información necesaria.

check_snmp

Este comando permite conocer el estado de una máquina remota mediante la consulta a su agente SNMP. Utiliza para ello el protocolo SNMP en cualquiera de sus versiones 1, 2 ó 3.

check_ssh

Este comando permite controlar si el servicio SSH de una máquina remota está activo o no. Realiza peticiones a este servicio para obtener la información necesaria.

check_swap

Este comando funciona en local, en la máquina donde está instalado Nagios. Permite monitorizar el tamaño de la memoria de intercambio utilizada y generar advertencias o errores cuando este valor sobrepaso los umbrales establecidos.

check_tcp

Este comando permite realizar peticiones arbitrarias a conexiones (sockets) TCP contra sistemas remotos. Por tanto permite monitorizar cualquier servicio que utilice sockets TCP para recibir peticiones.

Definir como comando:

```
define command{
    command_name    check_tcp
    command_line    $USER1$/check_tcp -H '$HOSTADDRESS$' -p $ARG1$
}


define service{
    use                     generic-service
    host_name               nombre_host
    service_description     telnet
    check_command           check_tcp!23
}
```

check_time

Este comando permite comprobar si el servicio de hora (TIME) está funcionando en una máquina remota. Realiza conexiones a este servicio para obtener la información.

check_udp

Este comando permite realizar peticiones arbitrarias a conexiones (sockets) UDP contra sistemas remotos. Por tanto permite monitorizar cualquier servicio que utilice sockets UDP para recibir peticiones.

check_ups

Este comando permite monitorizar el estado del servicio UPS en máquinas remotas; para ello hace peticiones a este servicio. Necesita paquetes adicionales instalados en el sistema de monitorización.

check_users

Este comando permite conocer el número de usuarios conectados actualmente en el sistema local, en el que se está ejecutando Nagios. Genera advertencias y errores cuando el número supera el umbral fijado.

check_vsz

Este comando permite comprobar que el tamaño en memoria de un programa determinado no sea mayor de un límite fijado. Cuando se produzca el caso contrario se generarán advertencias y/o errores.

urlize

Este comando permite, usando con otro comando, que la salida de este último se pueda mostrar en la pantalla de un navegador en formato HTML como un enlace hipertexto navegable.

Usage:urlize <url> <plugin> <arg1> ... <argN>

Options: -h, –help

```
Print detailed help screen
```

-V, –version

```
Print version information
```

Examples: Pay close attention to quoting to ensure that the shell passes the expected data to the plugin. For example, in:

urlize http://example.com/ check_http -H example.com -r 'two words'

```
the shell will remove the single quotes and urlize will see:
```

urlize http://example.com/ check_http -H example.com -r two words

```
You probably want:
```

urlize http://example.com/ “check_http -H example.com -r 'two words'”

Importante Para que este comando funcione bien, deberemos tener configurado en el archivo *cgi.cfg* la siguiente directiva :

```
escape_html_tags=0
```

check_xml_url.sh

Esto lo necesite cuando tuve que XML de diferentes Webservices tengan la sintaxis correcta, para asi saber si habia una falla en los mismos.

```
#!/bin/bash
wget -q -O - --user-agent="Mozilla/5.0 (Macintosh; Intel Mac OS X 10.8; rv:21.0)
Gecko/20100101 Firefox/21.0" $1 | xmlstarlet val - 1>/dev/null 2>/dev/null
EXIT_STATUS=$?

if [[ $EXIT_STATUS -eq "0" ]]; then
    echo "OK - La sintaxtis es correcta : $EXIT_STATUS"
fi

if [[ $EXIT_STATUS -ne "0" ]]; then
    echo "CRITICAL - La sintaxtis no es correcta : $EXIT_STATUS"
fi
```

Ejemplo de configuración :

```
     check_command                       check_xml_url!
https://wsaa.afip.gov.ar/ws/services/LoginCms?wsdl
     check_command                       check_xml_url!
https://serviciosjava.afip.gob.ar/wsmtxca/services/MTXCAService?wsdl
```

check_xml_afip.php

En Argentina tenemos un servicio de Factura Electronica, si queremos verificar su status, deberemos consumir este WebService https://serviciosjava.afip.gob.ar/wsmtxca/services/MTXCAService/dummy , del cual obtendremos esta respuesta que debemos parsear :

```
<?xml version="1.0" encoding="UTF-8"?>
<ns1:dummyResponse xmlns:ns1="http://impl.service.wsmtxca.afip.gov.ar/service/">
   <appserver>OK</appserver>
   <authserver>OK</authserver>
   <dbserver>OK</dbserver>
</ns1:dummyResponse>
```

```
#!/usr/bin/php
<?php
/*
SimpleXMLElement Object
(
    [appserver] => OK
    [authserver] => OK
    [dbserver] => OK
)
*/
$xml_cae_dummy =
simplexml_load_file('https://serviciosjava.afip.gob.ar/wsmtxca/services/MTXCAService
/dummy');

$appserver_status = $xml_cae_dummy->appserver;
$authserver_status      = $xml_cae_dummy->authserver;
$dbserver_status  = $xml_cae_dummy->dbserver;

if (($appserver_status == 'OK') && ($authserver_status == 'OK') && ($dbserver_status
== 'OK')) {
    print("OK - [appserver] ".$appserver_status." [authserver] ".
$authserver_status." [dbserver] ".$dbserver_status."|rc=0\n");
    exit(0);
} else {
    print("CRITICAL - [appserver] ".$appserver_status." [authserver] ".
```

```
$authserver_status." [dbserver] ".$dbserver_status."|rc=2\n");
    exit(2);
}
?>
```

check_nfe_status

En Brasil se utiliza la Nota Fiscal eletrônica, este plug esta en desarrollo. Por eso no lo publico, hay un desarrollo en java para realizar estos chequeo : http://www.vivaolinux.com.br/dica/Plugin-NFe-2.00-Nagios

check_heartbeat

Script simple para chequear el estado de HeartBeat y sus nodos, muy útil por cierto

```
#!/bin/bash
# Author: Emmanuel Bretelle
# Date: 12/03/2010
# Description: Retrieve Linux HA cluster status using cl_status
# Based on http://www.randombugs.com/linux/howto-monitor-linux-heartbeat-snmp.html
#
# Autor: Stanila Constantin Adrian
# Date: 20/03/2009
# Description: Check the number of active heartbeats
# http://www.randombugs.com

# Get program path
REVISION=1.3
PROGNAME=`/bin/basename $0`
PROGPATH=`echo $0 | /bin/sed -e 's,[\\/][^\\/][^\\/]*$,,'`

NODE_NAME=`uname -n`
CL_ST='/usr/bin/cl_status'

#nagios error codes
#. $PROGPATH/utils.sh
OK=0
WARNING=1
CRITICAL=2
UNKNOWN=3

usage () {
    echo "\
Nagios plugin to heartbeat.

Usage:
  $PROGNAME
  $PROGNAME [--help | -h]
  $PROGNAME [--version | -v]

Options:
  --help -l Print this help information
  --version -v  Print version of plugin
"
}

help () {
    print_revision $PROGNAME $REVISION
    echo; usage; echo; support
}


while test -n "$1"
do
```

```
  case "$1" in
    --help | -h)
      help
      exit $STATE_OK;;
    --version | -v)
      print_revision $PROGNAME $REVISION
      exit $STATE_OK;;
#    -H)
#      shift
#      HOST=$1;;
#    -C)
#      shift
#      COMMUNITY=$1;;
    *)
      echo "Heartbeat UNKNOWN: Wrong command usage"; exit $UNKNOWN;;
  esac
  shift
done

$CL_ST hbstatus > /dev/null
res=$?
if [ $res -ne 0 ]
then
  echo "Heartbeat CRITICAL: Heartbeat is not running on this node"
  exit $CRITICAL
fi

declare -i I=0
declare -i A=0
NODES=`$CL_ST listnodes`

for node in $NODES
do
  status=`$CL_ST nodestatus $node`
  let I=$I+1
  if [ $status == "active" ]
  then
    let A=$A+1
  fi
done

if [ $A -eq 0 ]
then
  echo "Heartbeat CRITICAL: $A/$I"
  exit $CRITICAL
elif [ $A -ne $I ]
then
  echo "Heartbeat WARNING: $A/$I"
  exit $WARNING
else
  echo "Heartbeat OK: $A/$I"
  exit $OK
fi
</code>

<code>
define command {
 command_name    check_ha_by_ssh
 command_line    $USER1$/check_by_ssh -l root -H $HOSTADDRESS$ -C
"/usr/local/sbin/check_heartbeat.sh"
}
```

check_systemimager

Script para chequear que nuestras imágenes estén actualizadas a la fecha

```
#!/usr/bin/php -q
# Sergio Cayuqueo <cayu@cayu.com.ar>
# http://cayu.com.ar
<?php
$lista_imagenes = shell_exec("si_lsimage --verbose|grep Image");
$lista_imagenes = preg_split("/[\n]+/",$lista_imagenes);
$fecha_actual = date('Y.m.d');
foreach($lista_imagenes as $imagen) {
    if(strlen($imagen)>0) {
      if(@!$i) {
          $i=1;
      }
      $imagen = preg_split("/[\s]+/",$imagen);
      $imagenes[$i]['nombre'] = $imagen[2];
      $imagenes[$i]['actualizada'] = $imagen[4];
      $imagenes[$i]['ip'] = $imagen[8];
      if($imagen[4] == $fecha_actual) {
          $imagenes[$i]['estado'] = "ok" ;
      } else {
          $imagenes[$i]['estado'] = "critical" ;
            $critical=1;
      }
      $i++;
    }
}

if(@$critical) {
    $head = "CRITICAL - Hubo un desfasaje en una o mas imagenes\n";
    $exit = 2;
} else {
    $head = "OK - Todas las imagenes actualizadas a la fecha\n";
    $exit = 0;
}
print $head;
foreach($imagenes as $imagen) {
    if(strlen($imagen['nombre'])<9) {
      $tab = "\t\t";
    } else {
      $tab = "\t";
    }
    if($imagen['estado'] == "ok") {
      print "OK - ".$imagen['nombre']." ".$tab.$imagen['ip']."\n";
    } else {
        print "CRITICAL - ".$imagen['nombre']."   ".$tab.$imagen['ip']." \t
actualizado a : ".$imagen['actualizada']."\n";
    }
}
exit($exit);
?>
```

```
define command {
 command_name    check_si_by_ssh
 command_line    $USER1$/check_by_ssh -l root -H $HOSTADDRESS$ -C
"/usr/local/sbin/check_systemimager.php"
}
```

check_dba_2pc_pending

Chequeo de *Distributed transactions* de Oracle, por medio de SSH.

```
#!/usr/bin/perl
```

```
open(SQLPLUS_SELECT, "sqlplus -S \"/ as sysdba\"
@/usr/local/bin/dba_2pc_pending_select.sql |") or die "No puedo ejecutar: $!";
while (<SQLPLUS_SELECT>) {
    chomp($_);
    if(length($_)>1) {
        $select = $_;
    }
}
close SQLPLUS_SELECT;

if($select == "no rows selected") {
    print "OK - Resultado : ".$select." | pending=0;\n";
    exit 0;
} else {
open(SQLPLUS_DELETE, "echo \"execute sys.dbms_transaction.purge_lost_db_entry('".
$select."');\"|sqlplus -S \"/ as sysdba\" |") or die "No puedo ejecutar: $!";
    while (<SQLPLUS_DELETE>) {
      chomp($_);
      if(length($_)>1) {
          $delete = $_;
      }
    }
    close SQLPLUS_DELETE;
    print "WARNING - Resultado : ".$select." : ".$delete." | pending=1;\n";
    exit 1;
}
```

El script sql

```
SET pagesize 0
SET trimspool ON
SET headsep off
SELECT LOCAL_TRAN_ID FROM dba_2pc_pending;
exit;
```

El check_command en la configuracion del servicio

```
check_by_ssh!sudo su - oracle -c /usr/local/bin/dba_2pc_pending.pl
```

El sudoers del equipo a monitorear

```
monitoreo   ALL = NOPASSWD: /bin/su - oracle -c /usr/local/bin/dba_2pc_pending.pl
```

check_microstrategy

Chequear que el servicio de MicroStrategy este corriendo

```
define command {
        command_name                    check_microstrategy
        command_line                    $USER1$/check_microstrategy.sh $HOSTADDRESS$
        register                        1
}


define service {
        service_description             Servicio MicroStrategy
        use                             generic-service
        check_command                   check_microstrategy
        max_check_attempts              1
        check_interval                  1
        retry_interval                  1
        active_checks_enabled           1
        check_period                    24x7
        notification_period             24x7
```

```
        notification_options            r,c
        notifications_enabled           1
        register                        1
}
```

check_microstrategy.sh

```
#!/bin/sh
SALIDA_SSH=`ssh $1 -l monitoreo "sudo /msis/var/opt/MicroStrategy/bin/mstrctl -s
IntelligenceServer gs | grep state|  sed 's/<[^>]*[>]//g' | sed 's/\t//g' | sed
's/\n//g'"`
if [ $SALIDA_SSH="running" ]
then
    echo "OK - Proceso MicroStrategy corriendo"
    exit 0;
else
    echo "CRITICAL - Hay un problema con el proceso MicroStrategy"
fi
```

count_archlogs.pl

```
#!/usr/bin/perl
use strict;
use warnings;

my $ORACLE_SID=$ARGV[0];

sub get_sorted_files {
   my $path = shift;
   opendir my($dir), $path or die "no puedo abrir $path: $!";
   my %hash = map {$_ => (stat($_))[9] || undef} # saltar listas vacias
    map  { "$path$_" }
    grep { m/.dbf/i }
    readdir $dir;
    closedir $dir;
    return %hash;
}

my %files = get_sorted_files("/oracle/arclog/".$ORACLE_SID."/");
my $count = keys %files;

if($count < 60) {
    print "OK - Encontrados $count redo logs en /oracle/arclog/".
$ORACLE_SID."/|'redologs'=$count\n";
    exit 0;
} else {
    print "CRITICAL - Encontrados $count redo logs en /oracle/arclog/".
$ORACLE_SID."/|'redologs'=$count\n";
    exit 2;
}
```

select_count.sh

Si necesito alertar cuando crecen X registros en una tabla Oracle, en el dia de hoy se puede ejecutar esto :

```
echo -e "set head off\nset pagesize 0\nSELECT COUNT(DATA) FROM APPREG.DATA WHERE
DATA = TO_DATE(SYSDATE,'DD/MM/YY');" |  sqlplus -S "/ as sysdba" | awk '/^[ 0-
9\.\t ]+$/ {print int($1)}'
```

Dentro de un script quedaria asi :

```
#!/bin/bash
# Chequear registros en Tabla y alertar al llegar al limite
CONTEO=`echo -e "set head off\nset pagesize 0\nSELECT COUNT(DATA) FROM APPREG.DATA
WHERE DATA = TO_DATE(SYSDATE,'DD/MM/YY');" |  sqlplus -S "/ as sysdba" | awk '/^[ 0-
```

```
9\.\t ]+$/ {print int($1)}'`
LIMITE=5

if [[ "$CONTEO" -ge "$LIMITE" ]]
then
    echo "CRITICAL - Hay $CONTEO registros";
    exit 2;
else
    echo "OK - Hay $CONTEO registros";
    exit 0;
fi
```

O para conocer el estado de los tablespaces en Oracle

```
DATABASE_STATUS=`echo -e "set head off\nset pagesize 0\nSELECT status,
database_status FROM v\\$instance;" |  sqlplus -S "/ as sysdba"| cut -f1`

case "$DATABASE_STATUS" in
        MOUNTED)
            start
            echo "CRITICAL - Los tablespaces de la base de datos estan
$DATABASE_STATUS -" `date '+DATE: %m/%d/%y TIME:%H:%M:%S'`
            exit 2;
            ;;
        OPEN)
            echo "OK - Los tablespaces de la base de datos estan $DATABASE_STATUS -"
`date '+DATE: %m/%d/%y TIME:%H:%M:%S'`
            exit 1;
            ;;
        *)
            echo "CRITICAL - Hay algun error con la base de datos $DATABASE_STATUS
-" `date '+DATE: %m/%d/%y TIME:%H:%M:%S'`
            exit 2;
esac
```

check_snmp_ifstatus_v3.pl

Version modificada por mi del script check_snmp_ifstatus.pl para que pueda consultar por medio de snmp v3

check_snmp_ifstatus_v3.tgz

check_oracle_tablespace.sh

Version modificada por mi del script check_oracle_tablespace.sh para no tener que usar oratab y conectarse con el sqlplus sin necesidad de tnsnames.ora, ademas soporta perfdata para graficarla en pnp4nagios

Ej ./check_oracle_tablespace.sh -s SID -H 10.1.1.98 -P 1521 -w 91 -c 96

Si se quiere saltear un tablespace, ej el **UNDO** agregamos al final de la query :

```
AND fs.TABLESPACE_NAME NOT LIKE '%UNDO%'
```

Fix Me! Tengo que agregar una opción para agregarle a la linea de comandos que sea ignorar table space

check_oracle_tablespace.sh.gz

ssl-cert-check.sh

Script para chequear validez del certificados SSL

```
ssl-cert-check.sh -c Certificado.crt -n -x 60
```

ssl-cert-check.sh.gz

Notas importantes

Estos son los comandos que acompañan a Nagios y que deberan ser invocados cada uno con sus respectivos parámetros y su forma de ejecución. Para saber más acerca de estos datos, necesarios para el uso de los comandos, se pueden invocar en línea de comandos con el parámetro –h lo que mostrará en la pantalla una ventana de descripción del comando, los parámetros que usa y cómo se invoca. Recordar que los ejecutables de los comandos se encuentran dentro del directorio libexec de la instalación de Nagios.

Archivos y directorios a tener en cuenta al realizar un backup

Script de inicio de Nagios

/etc/init.d/nagios

Directorio de Nagios

/usr/local/nagios

Sitios de consultas

Sitios de donde descargar plugins y agregados para Nagios

- Paquete Standard de plugins

http://nagiosplug.sourceforge.net/

- Plugins con soporte SNMP

http://nagios.manubulon.com/

- Plugins SAP CCMS

http://sourceforge.net/projects/nagios-sap-ccms/

- Interfaz alternativa de Nagios para visualizar en BlackBerry

http://nagiosmobile.sourceforge.net/

- Plugins para chequeos en BlackBerry

http://www.1ight.fr/plugins/BlackBerry/

- Plugins de chequeo de Lotus Notes

http://barbich.net/

- Servidor de proyectos Nagios, plugins, templates etc en etapa de desarrollo

http://www.nagiosforge.org/

- Servidor de proyectos Nagios, plugins, templates etc

http://www.nagiosexchange.org

- Agregado de chequeo de eventos Nagios para firefox

https://addons.mozilla.org/en-US/firefox/addon/3607
http://code.google.com/p/nagioschecker/

- Visor de sucesos para el escritorio (Linux, Windows)

http://sourceforge.net/projects/nagstamon

Sitios generales sobre funcionamiento de redes y protocolos

- Wiki de Redes de Comunicaciones II - Universidad de Madrid

http://ariadna.ii.uam.es/wiki/wiki_rc2/doku.php

- Un ensayo con contenidos conceptuales

http://www.monografias.com/trabajos95/recursos-red-y-su-monitoreo/recursos-red-y-su-monitoreo.shtml

- Explicación teórica clara sobre como *Calcular el coste por caída de Servicio*

http://www.cantabriatic.com/calcular-el-coste-por-caida-de-servicio/

- Como elaborar um SLA (The Service Level Agreement) - Acordo de Nível de Serviço

http://www.tiespecialistas.com.br/2011/01/como-elaborar-um-sla-the-service-level-agreement/

Lugares donde fue citado como referencia el presente documento

TESIS *Sistema de monitoreo y control de redes inalámbricas para optimización del servicio de Internet en la empresa Intercompu Ailaca Ramírez, Carlos Vinicio*

http://repo.uta.edu.ec/handle/123456789/442
http://repo.uta.edu.ec/bitstream/handle/123456789/442/Tesis_t655ec.pdf?sequence=1

TESIS *Implementación de NOC para el monitoreo de Servicios e Infraestructura de Redes para el Banco de Loja, basado en Software Libre, Solís Álvarez , Camilo Javier*

http://dspace.utpl.edu.ec/bitstream/123456789/9187/1/SOLIS%20ALVAREZ%20CAMILO%20JAVIER%2028-03-2014.pdf

TESIS *GESTIÓN Y MONITOREO DE UN LABORATORIO CON HERRAMIENTAS OPEN SOURCE, Ramos Galicia, Juan Christian*

http://132.248.52.100:8080/xmlui/bitstream/handle/132.248.52.100/2815/Tesis.pdf?sequence=1
http://www.ptolomeo.unam.mx:8080/xmlui/handle/132.248.52.100/2815

TESIS *Implementación de un sistema de monitorización para la división de ingenierías civil y geomática utilizando la herramienta Nagios*

http://www.ptolomeo.unam.mx:8080/xmlui/bitstream/handle/132.248.52.100/4445/Tesis.pdf?sequence=1
http://www.ptolomeo.unam.mx:8080/xmlui/handle/132.248.52.100/4445

signa - Sistema Integral de Gestión y Notificación de Alarmas

http://signa.googlecode.com/svn/trunk/entrega2/anexos/Anexo%20F%20-%20Descripci%C3%B3n%20de%20los%20Sistemas%20Finalistas.doc
https://code.google.com/p/signa/