UNIVERSIDAD NACIONAL AUTÓNOMA DE NICARAGUA.
UNAN-LEÓN.

FACULTAD DE CIENCIAS.

Departamento de Computación.

Trabajo Monográfico para optar al título de:

Ingeniero en Sistemas de Información.

TEMA:

Instalación y Configuración de Herramientas GNU para Gestión de Redes y Estudio del Protocolo SNMP .

ELABORADO POR:

- **Br. Nerlyng Roxana Corrales Quiroz.**
- **Br. David Antonio Maradiaga Gutiérrez.**
- **Br. Elkis Aismara Ortega Carrasco.**

TUTOR: Msc. ALDO RENÈ MARTINEZ.

León Junio del 2007

Dedicatoria.

A Dios, por darnos la fortaleza de concluir con nuestros estudios universitarios.

A nuestros padres por su apoyo y esfuerzo necesario para la conclusión de nuestros estudios.

A nuestras familias por su cariño y apoyo incondicional.

Agradecimiento.

Especiales los agradecimientos a nuestros padres, por su apoyo y cariño brindado.

A nuestro tutor Msc. Aldo René Martínez por su tiempo dedicado, lecciones y enseñanza brindada.

Al Lic. Derman Zepeda, Administrador de la Red de la UNAN-Managua, por su ayuda y apoyo incondicional.

A todos los profesores del Depto. de Computación por esos cuatro años de dedicación y enseñanza.

A nuestros compañeros de clases y amigos por los momentos compartidos.

Índice de Contenido

Índice de Figura

I. INTRODUCCIÓN.

En la actualidad varias redes se conectan entre sí con el uso de routers y un protocolo de interconexión de redes, de modo que los routers usan el protocolo para encubrir las características de las redes y proporcionar un servicio uniforme entre ellas, es decir, aunque cada red use una tecnología distinta y unas reglas específicas de transmisión, los hosts de cada red ven a la red de igual manera. Este es el poder de la abstracción de la interconexión entre redes.

La principal tecnología de interconexión de redes es el conjunto de protocolos *de Internet* llamados TCP/IP (Protocolo de Control de Transmisión/Protocolo de Internet), que se crearon en la Agencia de Proyectos de Investigación Avanzada de Defensa (DARPA) y que son los que se usan en las redes grandes (Internet), pero también en las interconexión de redes menores (redes locales).

El protocolo SNMP (Simple Network Management Protocol; Protocolo Simple de Gestión de Red) a utilizar en nuestro estudio- se sitúa en la capa de transporte de la pila OSI, o en la capa de aplicación de la pila de protocolos TCP/IP. Gráficamente se podría ver así:

Aplicación **SNMP**
Transporte (TCP, **UDP**,...)
Interred(**IP**)
Interfaz de red
Física

El término Simple Network Management Protocol (SNMP) se usa para referirse a un conjunto de especificaciones para la gestión de redes que incluyen el protocolo en sí mismo, la definición de estructuras de datos, y otros aspectos asociados. A continuación se mencionan los conceptos principales de SNMP.

Sistema de Gestión de Red:

Comprende un conjunto de herramientas (Hardware y Software) para monitorizar y controlar la red, es decir, los diferentes elementos funcionales de la red.

Se pueden mencionar los siguientes Elementos de un Sistema de Gestión:

- Estación de gestión (Gestor).
- Agentes.
- Información de gestión
- Protocolo de Gestión de Red.

Herramientas a utilizar en el desarrollo de nuestro trabajo investigativo:
Lo primero que vamos a utilizar:

- Red Ethernet 10/100 Mb previamente diseñada en laboratorio Alcalá.
- Que las computadoras a utilizar tengan el sistema operativo Linux, preferiblemente la distribución OPEN-SUSE10.2.
- Software a evaluar: **MRTG, CACTI, IPAUDIT, NTOP, BIGSISTER, WIRESHARP**.

II. Antecedentes.

A pesar de que la gestión de redes es un tema de gran importancia en el campo de la tecnología, ya que permite que equipos conectados a una red tengan un mejor funcionamiento; en el Departamento de Computación de la UNAN-León se han realizado pocos trabajos relacionados al tema de redes, los cuales se tratan de Seguridad en Redes, interconexión de redes, etc. Pero hasta la fecha no se ha elaborado ningún trabajo sobre Gestión de Redes.

III. Justificación.

Nuestro trabajo investigativo lo realizamos con la intención de proporcionar un documento y una ayuda a los estudiantes de Ingeniería Telemática del Departamento de Computación de la UNAN – León, en el curso de Gestión de Redes, ya que actualmente en el departamento no hay ningún documento sobre este tema.

En nuestro país muy pocas empresas que hacen uso de la tecnología en redes con software libre cuentan con mecanismos de Gestión de Redes, por lo que pretendemos configurar algunas herramientas y proporcionarles información de las mismas.

La gestión de red juega un papel importante en el buen funcionamiento de las redes y se hace imprescindible su aplicación por las siguientes razones:

- Los sistemas de información son vitales y están soportados sobre redes.
- La información manejada tiende a ser cada día mayor y a estar más dispersa.
- Las nuevas tecnologías de red requieren de una gestión cada vez más especializada, que le permita el empleo eficiente de sus recursos de telecomunicaciones.
- El adecuado empleo de las tecnologías de gestión de red permite mejorar la eficiencia, disponibilidad y el rendimiento de las redes, aumentar la relación calidad/costo en el diseño de las redes, así como aumentar la satisfacción de los usuarios por el servicio de red proporcionado.

IV. Objetivos.

Objetivo general:

- Instalar y configurar las principales herramientas de gestión de redes GNU.
- Realizar un estudio del protocolo para gestión de redes SNMP.

Objetivos específicos:

- Instalar y configurar un agente SNMP, en los dispositivos a gestionar.
- Implementar una pequeña práctica para conocer y comprender el funcionamiento del protocolo SNMP y las reglas de codificación BER aplicadas a éste.
- Configurar las siguientes herramientas de gestión de red: MRTG, Cacti, Wireshark, IpAudit, BigSister, Ntop.
- Hacer una comparación de las Herramientas de Gestión de Red GNU, previamente instaladas y configuradas.
- Crear un documento de soporte para la clase de Gestión de Redes, la cual pertenece a la carrera de Ingeniería Telemática.

V. MARCO TEÓRICO.

1. GESTIÓN DE REDES.

1.1.Introducción a las redes.

¿Qué es una red? Una *red* es un sistema de transmisión de datos que permite el intercambio de información entre ordenadores. La información que pueden intercambiar los ordenadores de una red puede ser de lo más variada: correos electrónicos, vídeos, imágenes, música en formato MP3, registros de una base de datos, páginas Web, etc. La transmisión de estos datos se produce a través de un medio de transmisión o combinación de distintos medios: cables de fibra óptica, tecnología inalámbrica, enlaces vía satélite (el intercambio de información entre ordenadores mediante disquetes no se considera una red).

Clasificación según su tamaño: LAN, MAN y WAN

Las redes LAN (*Local Área Network*, Redes de *Á*rea Local) son las redes que todos conocemos, es decir, aquellas que se utilizan en nuestra empresa. Son redes pequeñas, entendiendo como pequeñas, las redes de una oficina, de un edificio. Debido a sus limitadas dimensiones, son redes muy rápidas en las cuales cada estación se puede comunicar con el resto.

Las redes WAN (*Wide Área Network*, Redes de *Á*rea *E*xtensa) son redes punto a punto que interconectan países y continentes. Por ejemplo, un cable submarino entre Europa y América, o bien una red troncal de fibra óptica para interconectar dos países. Al tener que recorrer una gran distancia sus velocidades son menores que en las LAN aunque son capaces de transportar una mayor cantidad de datos.

Como vemos, las redes LAN son pequeñas y las redes WAN, muy grandes: debe existir algún término para describir unas redes de tamaño intermedio. Esto es, las redes MAN (*Metropolitan Área Network*, Redes de Área Metropolitana). Son usadas para interconectar comercios, hogares y administraciones públicas.

Clasificación según su distribución lógica.

Todos los ordenadores tienen un lado cliente y otro servidor: una máquina puede ser servidora de un determinado servicio pero cliente de otro servicio.

- **Servidor**. Máquina que ofrece información o servicios al resto de los puestos de la red. La clase de información o servicios que ofrezca determina el tipo de servidor que es: servidor de impresión, de archivos, de páginas Web, de correo, de usuarios, de *IRC* (charlas en Internet), de base de datos...
- **Cliente**. Máquina que accede a la información de los servidores o utiliza sus servicios. Ejemplos: Cada vez que estamos viendo una página Web (almacenada en un servidor remoto) nos estamos comportando como clientes. También seremos clientes si utilizamos el servicio de impresión de un ordenador remoto en la red (el servidor que tiene la impresora conectada).

Dependiendo de si existe una función predominante o no para cada puesto de la red, las redes se clasifican en:

- Redes cliente/servidor. Los papeles de cada puesto están bien definidos: uno o más ordenadores actúan como servidores y el resto como clientes. Los servidores suelen coincidir con las máquinas más potentes de la red. No se utilizan como puestos de trabajo. En ocasiones, ni siquiera tienen monitor puesto que se administran de forma remota: toda su potencia está destinada a ofrecer algún servicio a los ordenadores de la red. Internet es una red basada en la arquitectura cliente/servidor.

- Redes entre iguales. No existe una jerarquía en la red: todos los ordenadores pueden actuar como clientes (accediendo a los recursos de otros puestos) o como servidores (ofreciendo recursos). Son las redes que utilizan las pequeñas oficinas, de no más de 10 ordenadores.

1.2.Introducción a la Gestión de Redes.

1.2.1 Conceptos.

Los marcos de Gestión de red estándar tradicionales (tales como SNMP (Simple Network Management Protocol)), aplicado a Internet o CMIP (Protocolo Común de Información de Gestión), aplicado a TMN (Telecomunication Management Network, Red de Gestión de Comunicación) fueron concebidos a principio de los años 90 para la Gestión de Redes de Comunicación. Fueron ellos quienes adoptaron el modelo gestor-agente, que gobierna las interacciones entre las distintas Aplicaciones de Gestión.

Gestión de redes se refiere a la planificación, organización, supervisión y control de elementos de comunicaciones para garantizar un nivel de servicio, y de acuerdo a un coste.

Las tareas típicas englobadas en lo que se conoce como gestión de redes pueden ser el saber cual es el porcentaje de utilización de la conexión al proveedor de servicios de acceso a Internet, detectar que un encaminador o router no funciona como debiera, cuantos ordenadores hay conectados a una red local, detectar y registrar intentos de acceso no autorizados, entre otros.

La monitorización es la encargada de la gestión de redes que se ocupa de la observación y análisis del estado y del comportamiento de los recursos gestionados, abarcando cuatro fases:

- Definición de la información de gestión que se monitoriza.
- Acceso a la información de monitorización.
- Diseño de políticas de monitorización.
- Procesado de la información de monitorización.

Gráficamente se vería así:

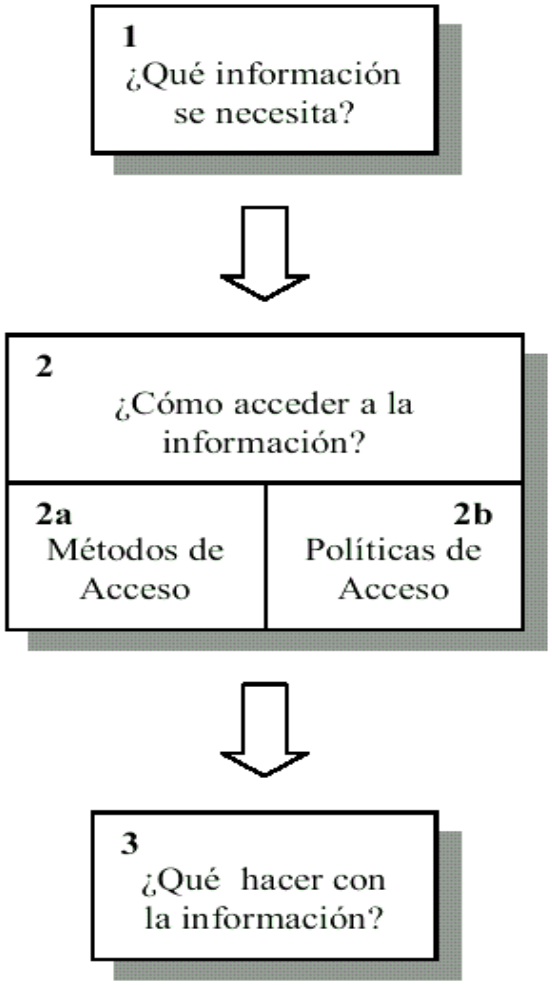


Fig. 1 Fases en la Monitorización de Red.

En la actualidad varias redes se conectan entre sí con el uso de routers y un protocolo de interconexión de redes, de modo que los routers usan el protocolo para encubrir las características de las redes y proporcionar un servicio uniforme entre ellas, es decir, aunque cada red use una tecnología distinta y unas reglas específicas de transmisión, los hosts de cada red ven a la red de igual manera; éste es el poder de la abstracción de la interconexión entre redes.

La principal tecnología de interconexión de redes es el *conjunto de protocolos de Internet* llamados TCP/IP (Protocolo de Control de Transmisión/Protocolo de Internet), que se crearon en la Agencia de Proyectos de Investigación Avanzada de Defensa (DARPA) y que son los que se usan en las redes grandes (Internet), pero también en las interconexión de redes menores (Redes Locales).

1.2.2 Sistema de Gestión de Red.

Esta compuesto de un conjunto de herramientas (Hardware y Software) para monitorizar, controlar y coordinar los elementos de la red.

Este sistema comprende:

- Supervisión.
- Comprobación.
- Sondeo.
- Configuración.
- Análisis.
- Evaluación y control de la red.

Se necesita de Hardware y Software adicional en los equipos de red para que:

- Desde el sistema de gestión se actúe sobre la red.
- Desde la red se notifique periódicamente al sistema de gestión el estado de la misma (desde los elementos activos).

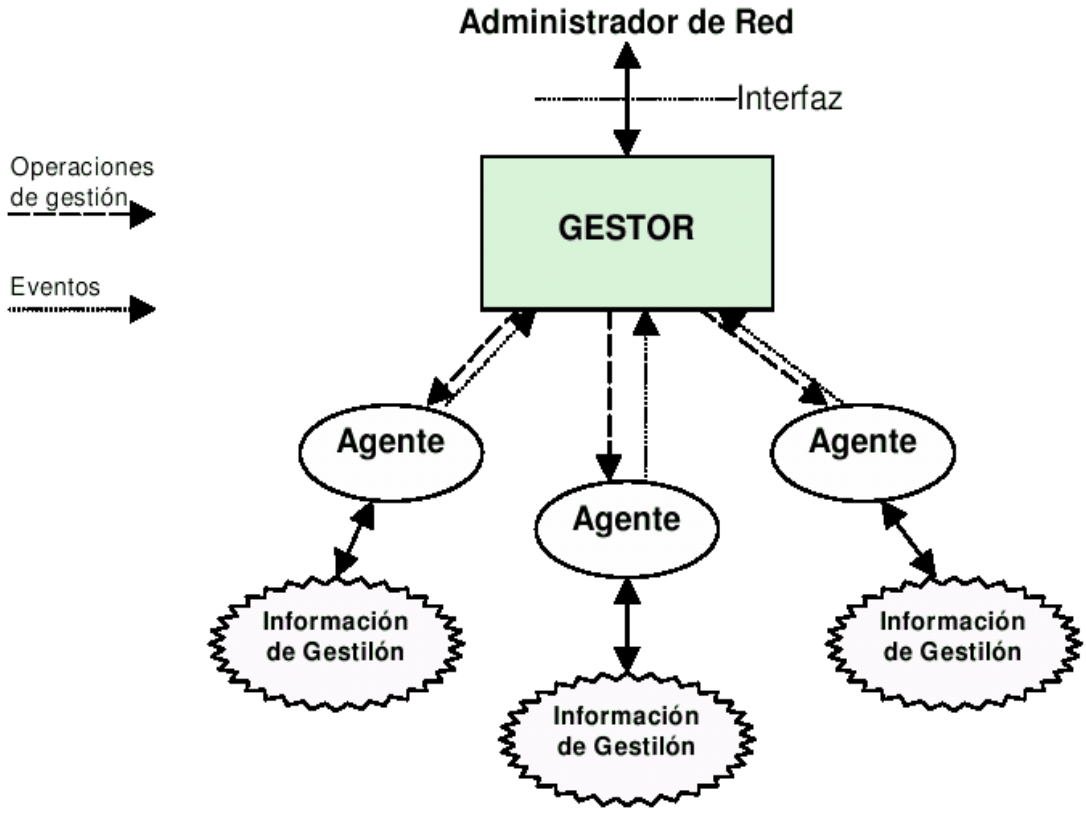


Fig. 2 Infraestructura de gestión.

1.2.3 Elementos del Sistema de Gestión:

- **Estación de gestión (Gestor).**

Son los elementos de un sistema de gestión que interaccionan con los operadores humanos y desencadenan las acciones pertinentes para llevar a cabo las operaciones por ellos invocadas. La estación de gestión sirve como interface entre el gestor de red (humano) y el sistema de gestión. La misma debe tener aplicaciones de gestión que sirvan para el análisis de datos, la recuperación de fallas, etc.; además, debe brindarle al gestor una interface que le permita monitorizar y controlar la red. Para ello debe ser capaz de trasladar los requerimientos del gestor dentro del sistema que gestiona los elementos remotos en la red. También debe poseer una base de información de gestión formada a partir de la recuperación de datos de los elementos gestionados.

Generalmente es un elemento autónomo, pero puede formar parte de un sistema compartido. Como mínimo comprende:

- Conjunto de aplicaciones de gestión, para análisis de datos de la red recuperación ante fallos, control de la red, etc.
- Interfaz de gestión, para monitorizar y controlar la red.
- Base de datos propia, contentiva de información de gestión obtenida de la base de datos residente en cada entidad de red gestionada.

- **Agentes.**

El otro elemento activo del sistema es el agente, un módulo de software que reside en los dispositivos a gestionar. El agente responde los pedidos de información y las acciones que provienen desde la estación de gestión, y puede enviarle a ésta información no solicitada, pero relevante. La estación de gestión y el agente se comunican utilizando el protocolo de gestión de red, en este caso SNMP.

Son los componentes de un sistema de gestión que llevan a cabo las operaciones de gestión invocadas por el gestor (o gestores) de la red.

En la mayoría de los sistemas GNU/Linux, se incluye un agente de SNMP que se trata de uno de los mas desarrollados en la actualidad. Se trata de la actualización de la librería SNMP de la Universidad de California en Davis (que a su vez se basa en la librería de la Universidad de Carnegie Mellon). La librería se llamaba, en versiones previas ucd-snmp, ahora se denomina net-snmp. La versión actual ha sido portada a GNU/Linux de la librería original por Juergen Schoenwaelder y Erik Schoenfelder, el desarrollador principal es Wes Hardaker.

Esta librería ha sido muy actualizada y desarrollada e incluye las herramientas de SNMP "tradicionales". Las últimas versiones parten de la base de código de la versión 2.1 y han sido tremendamente mejoradas.

La versión actual, la 4.1, incluye soporte para todas las versiones de SNMP (desde la uno, a la tres). Los agentes de SNMP que instala son perfectamente extensibles, tanto a través del propio código (con la API proporcionada) como a través de comandos definidos en la configuración.

Comprende:

- Hardware y Software propios de gestión en los dispositivos de red a gestionar, como son: hosts, puentes, conmutadores, concentradores, routers, módems, impresoras, etc.
- Aplicación que controla la recolección, procesamiento, análisis y visualización de la información de gestión.
- Se encarga de controlar la recolección, procesamiento, análisis y visualización de la información de gestión.

Podemos considerar que cada agente está formado por tres componentes:

- Funciones de usuario.
- Un protocolo de gestión, que permite monitorizar y controlar el agente.
- Instrucciones de gestión, que interactúan con la implementación del agente para permitir la monitorización y el control.

Se ha dicho que las estaciones de gestión sólo interactúan con los nodos. ¿Qué pasaría si el mismo nodo también fuera una estación de gestión? Es necesario apreciar que el modelo agente-gestor puede soportar directamente esto si consideramos que el software de cada agente del gestor puede realizar tanto la función de gestor como la de agente, es decir, que el modelo agente-gestor es también un modelo peer-to-peer.

- **Protocolo de gestión(SNMP – Simple Network Management Protocol).**

La estación de gestión y el agente se comunican utilizando el protocolo de gestión de red, en este caso SNMP, Protocolo a través del cual dialogan el Gestor y los Agentes.

Los agentes responden a solicitudes de información desde el Gestor y a órdenes o comandos desde el Gestor. Este protocolo permite al Gestor comunicarse con el agente para conocer el estado de los dispositivos (consultar la MIB – Management Infromation Base).

El protocolo no controla por sí mismo, sino que proporciona una herramienta con la que el administrador de red puede gestionar la red (supervisar, comprobar, sondear, configurar, analizar, evaluar, controlar, etc.).

También notifican al Gestor, de manera asíncrona, información relevante no solicitada por éste, y sí importante para la gestión.

Existen varios protocolos de gestión (CMIT/SMOSE, SNMP, CMOT, etc.), de ellos se destaca el protocolo SNMP, por varios motivos: es de fácil implantar, no requiere muchos recursos y es fácilmente extensible.

- **Información de gestión.**

La base de información de administración, denominada **MIB (Management Information Base, Base de Información de Gestión)**, constituye la descripción lógica de todos los datos de administración de la red. La MIB contiene información de estado y del sistema, estadísticas de rendimiento y parámetros de configuración. Desde el punto de vista de la gestión, los recursos de los dispositivos se representan como objetos.

Cada objeto es una variable que representa algún aspecto del elemento gestionado. Los objetos se organizan en la MIB. Esta funciona como un conjunto de puntos de acceso para la estación de gestión, quién lleva a cabo el monitoreo y control de la red recuperando o alterando los valores de los objetos.

La MIB trabaja con objetos, estos son definidos como variables de datos que representan algún aspecto de gestión del Agente. Representan los recursos a gestionar en un dispositivo de red. Sus valores reflejan el estado de los recursos de un dispositivo de la red. Los objetos están normalizados según la clase de dispositivo de red (router, puente, etc.).
A través del **MIB** se tiene acceso a la información para la gestión, contenida en la memoria interna del dispositivo en cuestión. MIB es una base de datos completa y bien definida, con una estructura en árbol, adecuada para manejar diversos grupos de objetos (información sobre variables/valores que se pueden adoptar), con identificadores exclusivos para cada objeto. Muchos de estos objetos son útiles para la gestión de fallos y de la configuración.

Las MIB's son accedidas usando un protocolo de gestión de red, en nuestro caso SNMP. Por intermedio del protocolo de gestión el Gestor puede:

- Cambiar la configuración de un Agente, modificando los valores de determinados objetos en la MIB.

- Monitorizar los valores de los objetos de la MIB.

El MIB define los objetos que pueden ser gestionados para cada capa en el protocolo TCP/IP. Hay dos versiones, MIB-I y MIB-II. MIB-I (que definen un total de 14 objetos y recientemente, con la introducción de MIB-II se definen hasta un total de 185 objetos) fue definida en el RFC 1213.

Un agente puede implementar varias MIBs concretas, pero todos implementan una general (MIB-II).Desde el punto de vista de la gestión, los recursos de los dispositivos se representan como objetos. La MIB funciona como un conjunto de puntos de acceso para la estación de gestión, quién lleva a cabo el monitoreo y control de la red recuperando o alterando los valores de los objetos.

1.3.Configuraciones de Gestión.

Se pueden construir relaciones jerárquicas entre las estaciones de gestión. Por ejemplo, se puede construir un sistema de gestión donde cada segmento de una LAN tiene una aplicación de gestión que controla el estado de los dispositivos de ese segmento; estas aplicaciones deberían informar a aplicaciones de estaciones de gestión regionales, las cuales deberían informar a estaciones de gestión entre empresas. En este ejemplo, el software de cada estación realiza un papel de gestor al monitorizar y controlar dispositivos que dependen de él jerárquicamente, y un papel de agente al informar y actuar según los comandos proporcionados por un superior jerárquico.

Hay que hacer significar que el concepto clave con las entidades de doble función es que la relación jerárquica depende de la configuración, mientras que la relación peer-to-peer depende de la arquitectura.

En principio se pueden plantear dos configuraciones de gestión:

- Centralizada (configuración tradicional)
- Distribuida

1.3.1 Configuración de Gestión Centralizada.

Adecuada para redes pequeñas en dimensiones, equipamiento y en volumen de tráfico de gestión.

Comprende: Un Gestor, o quizás dos por razones de confiabilidad. Uno funcionando y el otro redundante y agentes con sus MIB´s asociadas, según número de dispositivos a gestionar.

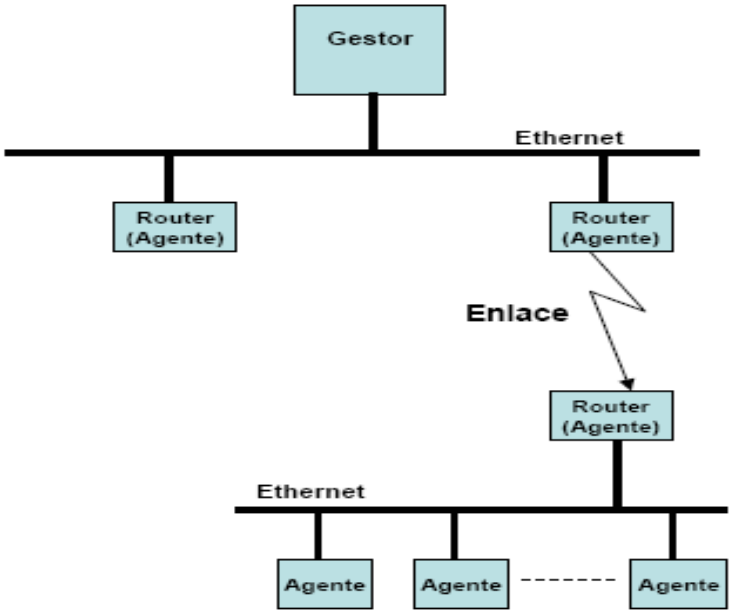


Fig. 3 Ejemplo de Gestión de Red Centralizada.

1.3.2 Configuración de Gestión Distribuida.

Es adecuada para redes grandes, donde un solo Gestor no resulta apropiado dado el volumen de carga de información de gestión que tendría que manejar desde los numerosos agentes como: un elevado tráfico de gestión.

En tal situación se opta por una gestión de red distribuida donde existen varios gestores a diferentes niveles, un gestor principal y varios gestores secundarios (intermedios), dicha gestión de red es jerarquizada.

Cada Gestor secundario se ocupa de la gestión de una parte de los Agentes. El Gestor principal, por intermedio de los gestores secundarios, se ocupa de la gestión de todos los Agentes de manera indirecta los Gestores secundarios desempeñan doble funcionalidad:

- Ante el Gestor principal actúan como Agentes.
- Antes los Agentes dependientes de ellos actúan como Gestores.

La gestión distribuida conlleva a dispersar la carga de procesamiento de gestión, y reducir la carga total de tráfico de gestión.

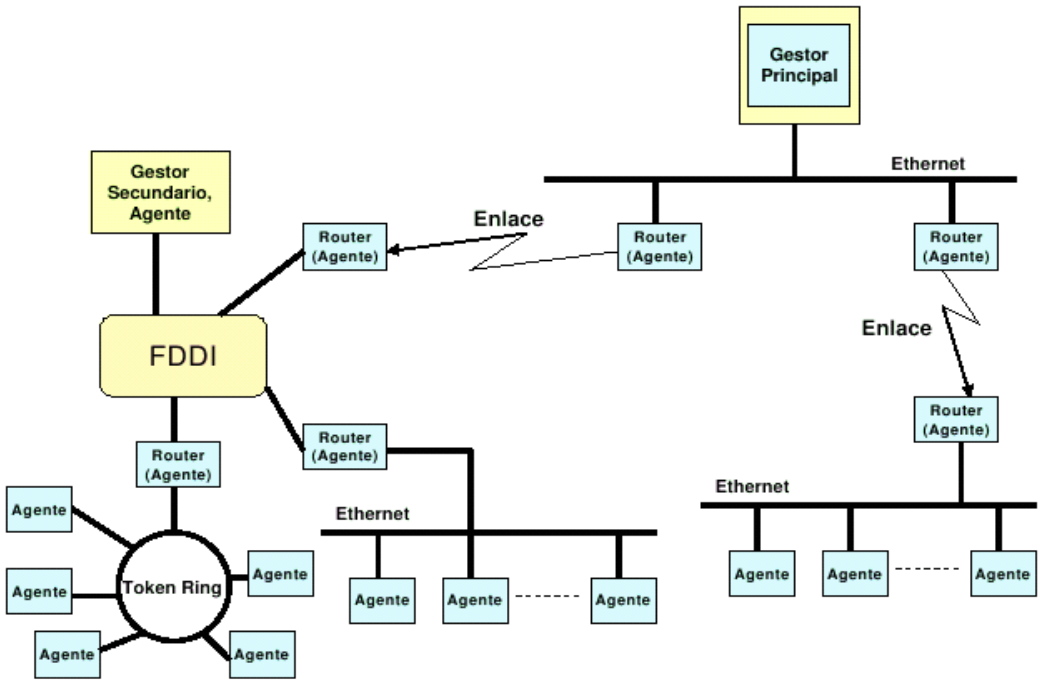


Fig. 4 Gestión de Red Distribuida

1.4.Protocolo Simple de Administración de Red (SNMP).

1.4.1 Conceptos.

El protocolo SNMP (*Simple Network Management Protocol*) utilizado para la gestión de redes basadas en TCP/IP, desde su anuncio como un estándar por medio de la CCITT (ahora UIT-T) en 1988, se convirtió en una de las herramientas más utilizadas gracias a su funcionalidad y forma simple de implementar. Esta funcionalidad ha sido mejorada en versiones siguientes como SNMPv2 y SNMPv2C. Sin embargo, hoy en día con la evolución del mundo de Internet éstas versiones dejan al descubierto grandes deficiencias en cuanto a seguridad. Por esta razón los grupos de expertos se han reunido para crear un nuevo conjunto de RFCs, conocidos como SNMPv3, orientados a corregir éstas deficiencias.

El funcionamiento de SNMP es sencillo, como dice el protocolo, aunque su implementación es tremendamente compleja. SNMP utiliza la capa de transporte de TCP/IP mediante el envío de datagramas UDP, sin embargo, el hecho de usar UDP hace que el protocolo no sea fiable (en UDP no se garantiza la recepción de los paquetes enviados, como en TCP).

El protocolo SNMP está cubierto por un gran número de RFCs (*Request For Comments*), entre ellos el RFC 1157, 1215 (versión 1), del 1441 al 1452 (versión 2), del 2271 al 2275 y del 2570 al 2575 (para SNMP v3).

SNMP se basa en un conglomerado de agentes. Cada agente es un elemento de la red que ofrece unas determinadas variables al exterior, para ser leídas o modificadas. Asimismo, un agente puede enviar "alertas" a otros agentes para avisar de eventos que tengan lugar. Generalmente se llama "gestor" al agente encargado de recibir estos eventos.

El protocolo SNMP es basado en tres conceptos:

- Managers ó Manejadores.
- Agents ó Agentes.
- MIB.

La idea básica de una Sistema Manejador de Red es la existencia de un Manejador y un Agente.

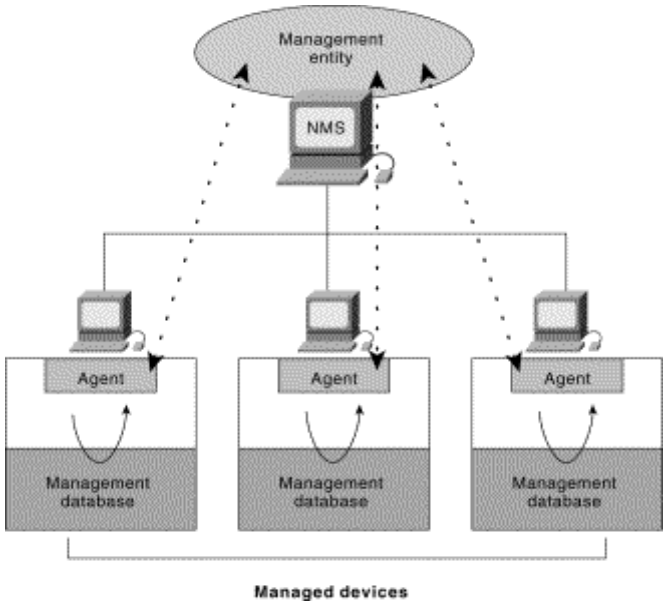


Fig. 5 Esquema de sistema manejador y agente.

En cualquier configuración, al menos un nodo manejador posee un software que soporta SNMP. La estación manejadora generalmente proporciona una interfaz al administrador de la red para controlar y observar los procesos de manejo de la misma. Esta interfaz permite al usuario realizar comandos (como por ejemplo desactivar un enlace, colectar estadísticas de un proceso determinado, etc.) y proporcionar información general del sistema. El punto principal de un Sistema Manejador de Red es un conjunto de

aplicaciones que reúnen las necesidades para ejercer las funciones. Como mínimo un sistema incluirá aplicaciones básicas para desarrollar las funciones de monitoreo, control de configuración y administración de las cuentas de los usuarios. Sistemas más sofisticados podrían incluir aplicaciones más elaboradas para estas categorías y con más posibilidades para la corrección de las fallas.

Por otro lado los dispositivos de red a ser manejados, incluyendo servidores, estaciones de trabajo, computadores personales, enrutadores, etc., son equipados con un módulo que incluye un software de Agente. El agente es responsable de:

- Colectar y mantener información sobre su ambiente local.
- Proporcionar información al Manejador de la red, ya sea en respuesta a un requerimiento o como un aviso de que algo anormal esta ocurriendo.
- Responder a los comandos ejecutados por el manejador para cambiar o alterar los parámetros de operación ó configuración local.

El esquema es sencillo, sin embargo su complejidad se incrementa a la hora de definir las variables (y su formato). Las variables ofrecidas para consulta por los agentes SNMP se definen a través de una MIB (*Management Information Base*, Base de Información de Gestión).

Cada agente SNMP ofrece información dentro de una MIB, tanto de la general (definida en los distintos RFCs) como de aquellas extensiones que desee proveer cada uno de los fabricantes. Así, los fabricantes de routers han extendido las MIBs estándar incluyendo información específica de sus equipos

Es importante resaltar que todas las aplicaciones de gestión de red generalmente comparten un protocolo común en toda la red. Este protocolo proporciona las funciones fundamentales para requerir información y ejecutar comandos hacia los agentes. Éste protocolo, es SNMP y hace uso de herramientas de comunicación como OSI ó TCP/IP.

SNMP (Protocolo Simple de Gestión de Red) permite a un servidor TCP/IP que ejecuta una aplicación SNMP, interrogar a otros nodos para estadísticas y condiciones de error de la red. Los otros servidores, que proporcionan agentes SNMP responden a estas preguntas y le permiten a un solo servidor recoger estadísticas de muchos nodos de la red.

El **Protocolo Simple de Administración de Red** o **SNMP** (Simple Network Management Protocol) es un protocolo de la Capa de Aplicación que facilita el intercambio de información de administración entre dispositivos de red. Él es parte de la suite de protocolos TCP/IP. SNMP permite a los administradores supervisar el desempeño de la red, buscar y resolver sus problemas, y planear su crecimiento.

SNMP no esta necesariamente limitado a las redes TCP/IP y ciertamente ayuda a que los datos esenciales en una red fluyan mejor.

1.4.2 Versiones.

El protocolo **Snmpv1** fue diseñado a mediados de los 80 por Case, McCloghrie, Rose, y Waldbusser, como una solución a los problemas de comunicación entre diferentes tipos de redes.

En un principio, su principal meta era el lograr una solución temporal hasta la llegada de protocolos de gestión de red con mejores diseños y mas completos. Pero esos administradores de red no llegaron y **SNMPv1** se convirtió en la única opción para la gestión de red.

El manejo de este protocolo era simple, se basaba en el intercambio de información de red a través de mensajes (**PDU's – Protocol Data Unit**). Además de ser un protocolo fácilmente extensible a toda la red, debido a ésto su uso se estandarizó entre usuarios y empresas que no querían demasiadas complicaciones en la gestión de sus sistemas informáticos dentro de una red.

No obstante este protocolo no era perfecto, además no estaba pensado para poder gestionar la inmensa cantidad de redes que cada día iban apareciendo. Para subsanar sus carencias surgió la versión 2 (SNMP v2). Las mayores innovaciones respecto a la primera versión son:

- Introducción de mecanismos de seguridad, totalmente ausentes en la versión 1. Estos mecanismos protegen la privacidad de los datos, confieren autentificación a los usuarios y controlan el acceso.
- Mayor detalle en la definición de las variables.
  Se añaden estructuras de la tabla de datos para facilitar el manejo de los datos. El hecho de poder usar tablas hace aumentar el número de objetos capaces de gestionar, con lo que el aumento de redes dejó de ser un problema.

Realmente esta versión 2 no supuso más que un parche, es más hubo innovaciones como los mecanismos de seguridad que se quedaron en pura teoría, no se llegaron a implementar. Por éstas razones se ha producido la estandarización de la versión 3. Con dos ventajas principales sobre sus predecesores:

- Añade algunas características de seguridad como privacidad, autentificación y autorización a la versión 2 del protocolo.
- Uso de Lenguajes Orientados a Objetos (Java, C++) para la construcción de los elementos propios del protocolo (objetos). Estas técnicas confieren consistencia y llevan implícita la seguridad, por lo que ayudan a los mecanismos de seguridad.

Específicamente las versiones SNMPv1 y SNMPv2 consisten de un conjunto de documentos que definen un protocolo de gestión de red, una estructura general para Manejadores de Información Básica (MIB) y un número especifico de datos estructurados de MIB para propósitos de manejo. En esencia el protocolo proporciona cuatro funciones:

- **Get:** Usado por un manejador para realizar algún requerimiento a un MIB de un agente.
- **Set:** Usado por un manejador para cambiar algún valor en un MIB de un agente.
- **Trap:** Usado por un agente para enviar un mensaje de alerta al manejador.
- **Inform:** Usado por el manejador para enviar un mensaje de alerta a otro manejador.

Es así de simple. Lo que le da el poder a SNMP son las extensivas estructuras de MIB estandarizados que hasta ahora han sido definidos ya que son ellos los que le permiten al agente colectar y almacenar la información que se necesita.

1.4.3 Arquitectura del Protocolo de Gestión de Redes.

SNMP fue diseñado como un protocolo de nivel de aplicación que forma parte de la suite de protocolos TCP/IP. Está propuesto para operar sobre el User Datagram Protocol (UDP).
La figura 6. muestra como se relaciona una aplicación de gestión SNMP que se encuentra en una estación de gestión, con un agente SNMP que está en el dispositivo gestionado. Desde la estación de gestión pueden enviarse tres tipos de mensajes SNMP en nombre de la aplicación de gestión: GetRequest, GetNextRequest, y SetRequest.

Debido a que SNMP utiliza UDP, protocolo no orientado a conexión, SNMP es en sí mismo no orientado a conexión. Por lo tanto, cada intercambio entre la estación de gestión y el agente es una transacción separada.

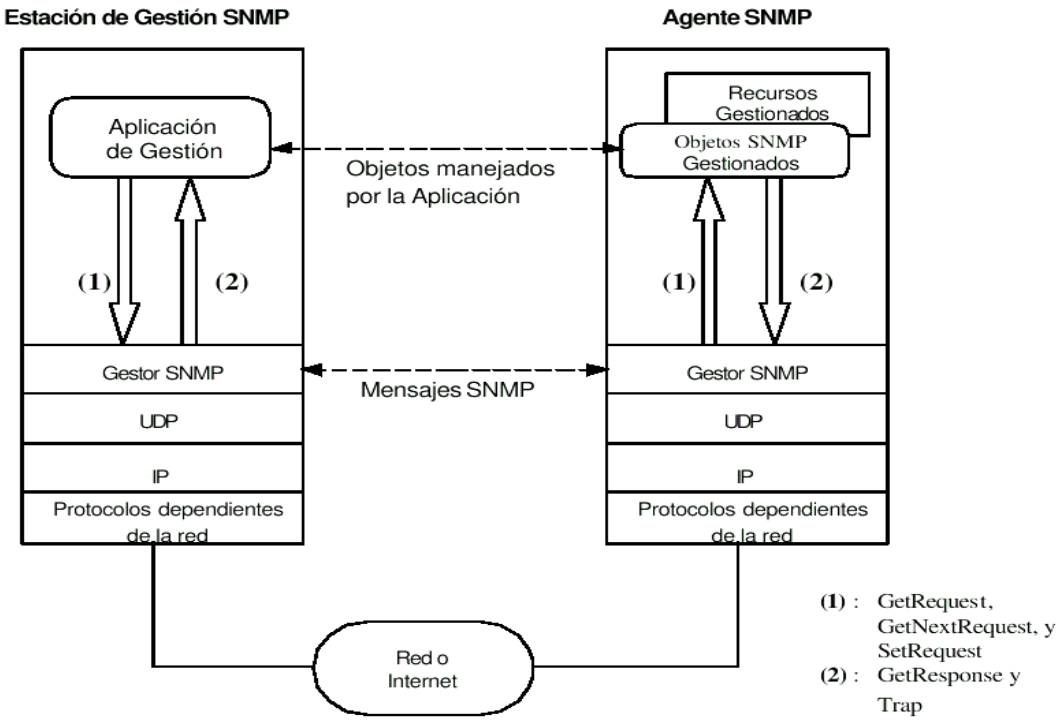


Fig. 6 Relación entre una aplicación de gestión y los objetos gestionados.

Para utilizar SNMP, es necesario que los dispositivos soporten parte de la suite de protocolos TCP/IP. Muchos dispositivos no soportan ninguna parte de dicha suite de protocolos, o bien implementan TCP/IP pero es deseable no agregar la carga adicional de SNMP. Para estos casos se desarrolló el concepto de Proxy. De esta manera un agente SNMP puede actuar en nombre de otros dispositivos. Cuando la estación de gestión desee recuperar información de dichos dispositivos, enviará un pedido al agente Proxy que se encargará de convertir los mensajes para adaptarlos al protocolo de gestión usado por los dispositivos mencionados. El agente Proxy recibirá las respuestas de los dispositivos y las convertirá al formato necesario para enviarlas a la estación de gestión. Esto mismo hará en el caso de la notificación de algún evento significativo.

1.4.4 Entidad SNMPv3.

En SNMPv3 se define una arquitectura de gestión de red que contiene:

- Una colección de entidades SNMP que interaccionan entre sí, en ésta cada entidad implementa una parte de las capacidades de SNMP y puede actuar como agente, gestor o una combinación de ambos.
- Cada entidad consiste en una colección de módulos que interaccionan entre sí para proporcionar servicios como una entidad que incluye un motor SNMP. El motor SNMP implementa funciones para enviar y recibir mensajes, autentificar y encriptar/desencriptar mensajes, y controlar el acceso a los objetos gestionados.

El papel de una entidad SNMP viene dado por el conjunto de módulos que implementa dicha entidad.

Un agente SNMP implementará un conjunto de módulos y también un gestor SNMP implementará otro conjunto de módulos.
Ésta estructura modular permite actualizar cada módulo por separado, sin tener que modificar el estándar por completo.

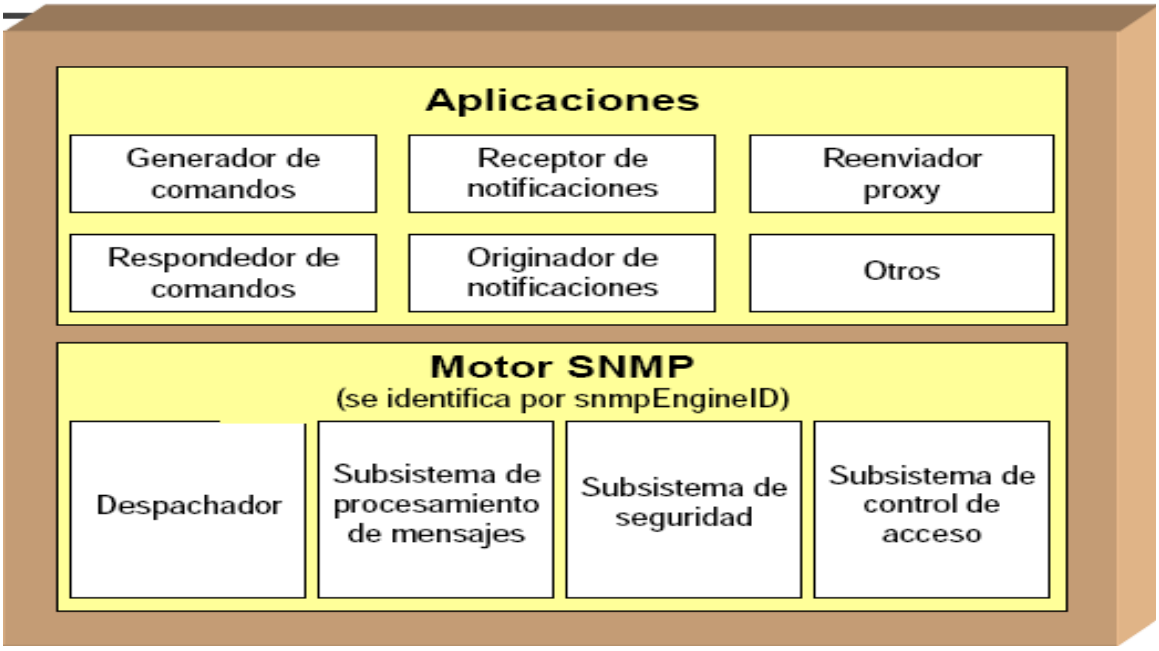


Fig. 7 Entidad SNMPv3.

Aplicaciones

- **Generador de comandos**

  Inicia la generación de PDUs Get, GetNext, GetBulk y/o Set, y procesa sus respuestas.

- **Respondedor de comandos**

  Recibe las PDUs Get, GetNext, GetBulk y/o Set dirigidas al motor local, las atiende, y genera la correspondiente respuesta.

- **Originador de notificaciones**

  Monitoriza en un sistema la ocurrencia de eventos o condiciones particulares y genera PDUs Trap y/o Inform en función de ellos.

- **Receptor de notificaciones**

  Permanece a la escucha de mensajes de notificación, y genera la respuesta a los PDUs Inform.

- **Reenviador Proxy (Proxy forwarder)**
  Reenvía mensajes SNMP

Motor SNMP

- **Despachador (dispatcher)**
  Se encarga del envío y recepción de mensajes y PDUs entre la red y los distintos módulos de la entidad
- **Subsistema de procesamiento de mensajes**
  Prepara las PDU para su envío dentro de un mensaje y extrae las PDUs de los mensajes entrantes. Puede soportar una o más versiones de SNMP
- **Subsistema de seguridad**
  Proporciona los servicios de seguridad (autentificación, encriptado) al subsistema de procesamiento de mensajes, implementa el modelo de seguridad basado en usuarios (USM). Puede implementar más modelos de seguridad, cuando se definan.
- **Subsistema de control de acceso**
  Proporciona un conjunto de servicios de autorización para comprobar los derechos de acceso a la MIB de una entidad. Aplicaciones respondedoras de comandos y generadoras de Notificaciones. Implementa el modelo de control de acceso basado en vistas (VACM).Puede implementar más modelos de control de acceso, cuando se definan.

Un gestor SNMP convencional incluye:

- Despachador.
- Subsistema de procesamiento de mensajes.
- Subsistema de seguridad.
- Aplicación generadora de comandos.
- Aplicación generadora de notificaciones.
- InformRequest.
- Aplicación receptora que notifica.

Un agente SNMP convencional incluye:

- Despachador.
- Subsistema de procesamiento de mensajes.
- Subsistema de seguridad.
- Subsistema de control de acceso.
- Aplicación respondedora de comandos.
- Aplicación generadora de notificaciones.
- Reenviador Proxy.

1.4.5 Procesamiento del Mensaje.

El RFC-2272 define en forma general el modelo para el procesamiento del mensaje en SNMPv3. Este modelo es responsable de aceptar los PDUs del Despachador, encapsularlo entonces en mensajes, e invocar el USM (Modelo de Seguridad del Usuario) para insertar los parámetros relacionados con la seguridad en el encabezado del mensaje. El modelo de procesamiento del mensaje también se encarga de aceptar mensajes entrantes, invocar el USM para procesar los parámetros de seguridad que se encuentran en el encabezado del mensaje y entrega el PDU (Unidad de Datos de Protocolo) al despachador.

En lo que a la estructura del mensaje se refiere. Los primeros cinco campos son generados por el modelo de procesamientos de mensajes entrantes / salientes. Los siguientes seis campos muestran los parámetros de seguridad usados por el USM. Finalmente el PDU, junto con el ContextEngineID y ContextName constituyen el PDU a ser procesado.

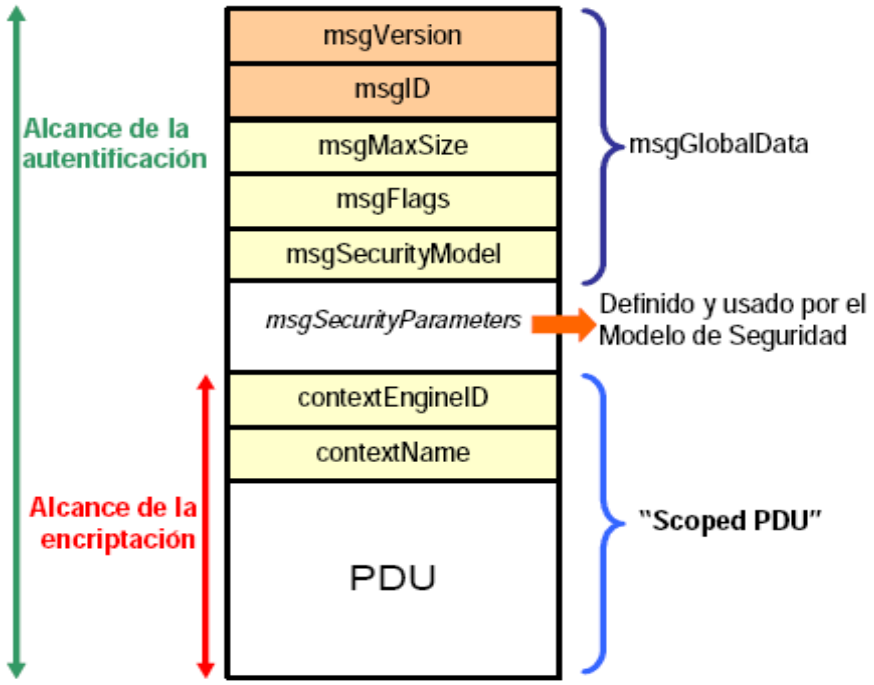


Fig. 8 Estructura del mensaje.

- **msgVersion**: Configurado para SNMPv3.
- **MsgID**: Un identificador único usado entre dos entidades SNMP para coordinar los mensajes de requisición y respuesta. Su rango es de 0 a $2^{31} - 1$.
- **MsgMaxSize**: Se refiere al tamaño máximo de un mensaje en octetos soportado por el que envía, con un rango de 484 a $2^{31} - 1$. Éste és el máximo tamaño que una entidad que envía puede aceptar de otra SNMP Engine.
- **MsgFlag**: Un arreglo de octetos que contiene tres banderas en los tres bits menos significativos:
  - ReportableFlag: Utilizada igual a 1 para los mensajes enviados conteniendo una requisición o un Inform, e igual a 0 para mensajes conteniendo una Respuesta, Trap ó Reporte PDU.

  - PriorFlag y AuthFlag: Son configuradas por el que envía para indicar el nivel de seguridad que le fue aplicado al mensaje.

- **MsgSecurityModel**: Es un identificador en el rango de $2^{31} - 1$ que indica qué modelo de seguridad utilizado por el que envió el mensaje, para que así el receptor tenga conocimiento de que modelo de seguridad deberá usar para procesar el mensaje. Existen valores reservados:
  - 1 para SNMPv1
  - 2 para SNMPv2
  - 3 para SNMPv3.
- **msgSecurityParameters**: parámetros del subsistema de seguridad.
- **contextEngineID**: identificador único de la entidad SNMP que ha de procesar el mensaje entrante.
- **contextName**: nombre único de un contexto en la entidad SNMP. Un contexto se refiere a un conjunto nombrado de instancias de objetos en la MIB local.
- **PDU**: una PDU SNMPv2.

1.4.6 Operaciones del Protocolo SNMP.

SNMP soporta el intercambio de mensajes entre Gestor y Agentes en un entorno de gestión de red (y entre Gestores).
Presenta dos modos de operación:

  - Modo “petición-respuesta”.
  - Modo “mensaje trap”.

Modo “petición-respuesta”: El Gestor envía una “petición” (solicitud) al Agente, éste desarrolla alguna acción y devuelve una “respuesta” al Gestor. Las “peticiones” típicamente se hacen para: Consultar (obtener), o Modificar (establecer) los valores de objetos MIB en un dispositivo gestionado.

Modo “mensaje trap” o modo “trap”: El agente envía al Gestor, de manera asíncrona, un mensaje no solicitado explícitamente por el Gestor. Los mensajes de esta naturaleza se les denomina genéricamente “traps”. Los mensajes “traps” se utilizan para notificar al Gestor una situación excepcional en un dispositivo gestionado, que normalmente conlleve a modificar valores de objetos MIB (uno o varios). Por ejemplo: el administrador de la red podría querer recibir notificación, mediante “traps”, cuando:

- El grado de congestión de cierto enlace alcance determinado nivel.
- Cierta interfaz en un router deje de funcionar.
- El número de paquetes descartados en un router alcance cierto valor y en general, cuando ocurra algún evento destacable desde la perspectiva de la gestión de la red.

Tipos de mensajes SNMP

- **GetRequest, GetNextRequest, GetBulkRequest**

Son mensajes que se envían de Gestor a Agente, para solicitar el valor de uno o más objetos MIB .La PDU contendrá los identificadores de tales objetos para los tres casos el Agente responde con el mensaje “Response”, contentivo de los objetos MIB solicitados y sus correspondientes valores. Los tres mensajes difieren en la granularidad de los datos solicitados:

GetRequest: para solicitar el valor de un objeto o un conjunto arbitrario de objetos.

GetNextRequest: múltiples mensajes de éste tipo se pueden emplear para “barrer” una lista o tabla de objetos.

GetBulkRequest: para solicitar un número grande de valores de objetos, evitando la necesidad de múltiples mensajes GetRequest y GetNextRequest.

- **SetRequest**

Mensaje que se envía de Gestor a Agente. Posibilita a un gestor establecer el valor de uno o más objetos MIB de un dispositivo gestionado. El Agente contesta con el mensaje "Response", confirmando así que la solicitud ha sido cursada,el valor del objeto, o valores, ha sido establecido.

- **Trap**

Mensaje que se envía del Agente al Gestor. Se genera de manera asíncrona, y NO en respuesta a petición alguna. Notifica al Gestor la ocurrencia de algún evento significativo predefinido, que requiere tal notificación. No requiere mensaje de respuesta por parte del Gestor, es un mensaje no confirmado.

En la RFC 1907 se definen tipos de "traps", por ejemplo:

- Arranque de un dispositivo de red.
- Establecimiento o pérdida de un enlace.
- Fallo de autentificación.

- **InformRequest**

Mensaje de Gestor a Gestor. Se utiliza para que un gestor notifique a otro Gestor cierta información de gestión MIB, que es remota a la entidad gestora que lo recibe. El Gestor receptor contesta con un mensaje "Response", como mecanismo de confirmación.

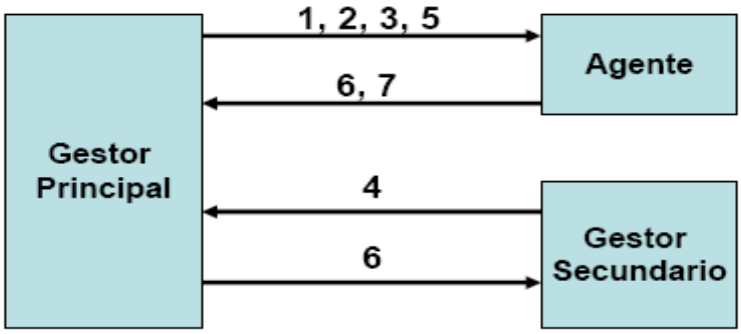


Fig. 9 Ámbito del mensaje.

1. GetRequest
2. GetNextRequest
3. GetBulkRequest
4. InformRequest
5. SetRequest
6. Response
7. Trap

1.4.6.1 Sondeo SNMP dirigido por traps (trap-directed polling).

El protocolo SNMP se basa en el sondeo o polling. El gestor sondea periódicamente a los agentes para ver si hay algo que necesite atención. Si una estación de gestión controla un gran número de agentes, y cada uno tiene un gran número de objetos, este mecanismo se vuelve poco eficiente. Por ello, se emplea la técnica del sondeo dirigido por trap:

Cuando llega un trap desde un agente, el gestor centra su atención en ese dispositivo, pero éste puede generar el siguiente problema:
Los traps no se confirman y el transporte es 'no fiable' (UDP). Por tanto, el gestor no se puede basar exclusivamente en la recepción de traps para obtener información de los dispositivos.

Para este problema una posible solución es:

- Sondeo al inicializar el sistema y a intervalos poco regulares (horas) para esto se pide alguna información clave, y algunas características básicas de funcionamiento a todos los agentes que conozca el gestor; el resto del tiempo, el gestor no sondea, y es el agente quien le avisa mediante un trap en caso de que ocurra algún evento anormal
  Ejemplos: Caída y reinicialización del agente, caída de un enlace.

Tras recibir el trap el gestor puede obtener más información del agente que envió el trap, o de otros agentes próximos a él para obtener más información y diagnosticar el problema.

Mediante el uso de comunidades, un agente puede limitar el acceso a su MIB en dos formas:

- Vista de la MIB: subconjunto de los objetos de la MIB
- Modo de acceso: READ-ONLY o READ-WRITE

La combinación de una vista de la MIB y un modo de acceso se denomina perfil de comunidad SNMP (SNMP community profile). A cada comunidad se le asigna un perfil, denominándose a esta asociación política de acceso SNMP (SNMP access policy). Cada paquete SNMP contiene el nombre de la comunidad, sin codificar. El agente sólo atiende la petición si el nombre de la comunidad es correcto para el tipo de acceso solicitado.
Se trata de un esquema de seguridad muy limitado por ello, en muchos agentes no se implementan las peticiones de escritura en la MIB (mensajes SetRequest).

1.4.7 Formato de los paquetes SNMP.

En SNMP la información se intercambia entre la estación de gestión y el agente en forma de mensajes SNMP. Cada mensaje incluye un campo Versión con el número de versión de SNMP (version = 0, para SNMPv1), un campo Community con el nombre de comunidad, y uno de los cinco tipos de unidades de datos de protocolo (PDU)3. Notar que el formato de las PDUs de GetRequest, GetNextRequest y SetRequest es el mismo que el de la PDU de GetResponse, con los campos error-status y error-index siempre en cero. Esto reduce el número de formatos de PDU con los que debe tratar la entidad SNMP.

Todos los paquetes contienen dos campos:

1. El número de versión de SNMP.
2. Un nombre de comunidad.

El resto del paquete depende del tipo del mismo, y se denomina PDU (Protocol Data Unit) de SNMP.

Versión	Comunidad	PDU de SNMP

Mensaje SNMP

Tipo PDU	Request ID	0	0	Asignaciones de Variables

PDUs de GetRequest, GetNextRequest y SetRequest

Tipo PDU	Request ID	Cod. Error	Índice error	Asignación de Variables

PDU de GetResponse

Tipo PDU	Empresa	Dir. agente	Trap genérica	Trap especifica	Time stamp	Asig. de Vbles.

PDU de Trap

Nombre 1	Valor 1	Nombre 2	Valor 2	...	Nombre N	Valor N

Asignación de variables

Request ID: identificador único por cada petición.

Código de error: indica que ha ocurrido una excepción al procesar una petición.

Posibles valores: noError (0), tooBig(1), noSuchName(2), badValue(3),readOnly(4), genErr(5).

Índice de error: indica qué variable de la lista causó la excepción, cuando el código de error no es 0.

Asignación de variables: lista de nombres de variables y sus correspondientes valores.

- Los nombres se especifican como identificadores de objetos (OIDs).
- En GetRequest, los valores son null.

Empresa (enterprise): objeto que genera el trap (valor de sysObjectID)

Dirección de agente: dirección IP del agente que genera el trap.

Trap genérica: tipo de trap genérico.

Trap específico: código de trap específico.

Time stamp: tiempo transcurrido entre la última reinicialización de la entidad y la generación del trap (valor de sysUpTime).

1.4.8 Ventajas e inconvenientes de SNMP.

Ventajas:

- Es un protocolo maduro, estándar de facto aceptado por la industria.
- Está disponible en gran cantidad de productos
- Es fácil de implementar y requiere pocos recursos del sistema.

Inconvenientes:

- Falta de seguridad:
  - ✔ Cualquier estación puede resetear variables con SetRequest, por lo que muchos fabricantes no implementan este comando.
  - ✔ No hay control de acceso: al recibir un PDU un agente no comprueba si ha sido enviado por una estación autorizada.
  - ✔ La identificación de comunidad viaja tal cual
- Mala utilización del ancho de banda:
  - ✔ No existe la posibilidad de transferir información por bloques.
- Limitaciones en el mecanismo de traps:
  - ✔ Sólo se puede informar de algunos eventos previstos.
  - ✔ No son reconocidas.
- No es apropiado para gestionar redes muy grandes (por el sondeo).

1.5.Información de Gestión SNMP.

1.5.1 Introducción.

Toda la información de gestión que se intercambia en las relaciones que se establecen entre un gestor y un agente, se describe desde dos puntos de vista complementarios: Por un lado la **Estructura de la Información de Gestión** (SMI, Structure of Management Information), que define la estructura lógica de la información, cómo se identifica y cómo se describe, es decir, la estructura sintáctica de la comunicación. Por otro lado, la **Base de Información de Gestión** (MIB, Management Information Base), que utilizando la sintáxis de la SMI describe los recursos que serán objetos de las relaciones de gestión.

1.5.2 Base de Información de Gestión (MIB-II).

La base de un sistema de gestión de redes es una base de datos que contiene información acerca de los elementos a gestionar. Esta base de datos es referida como Management Information Base (MIB). Cada recurso a gestionar se representa como un objeto. La MIB es una colección estructurada de dichos objetos. Cada sistema en la red (Workstation, Server, routers, bridges, etc.) mantiene una MIB que refleja el estado de los recursos gestionados en ese sistema. Una entidad de gestión de red puede monitorear y controlar los recursos del sistema, leyendo y modificando los valores de los objetos en la MIB. Para que la MIB cumpla con las necesidades de un sistema de gestión de redes, debe satisfacer ciertos objetivos.

- Los objetos usados para representar un recurso particular deben ser los mismos en todos los sistemas.
- Se debe usar un esquema común de representación de la información para lograr interoperabilidad.

El primer objetivo se logra definiendo los objetos y su estructuración en la MIB. El segundo objetivo se logra definiendo una estructura de la información de gestión (SMI, Structure of Management Information).

Cada objeto de un dispositivo gestionable por SNMP debe tener un nombre único con el que se le denominará en las operaciones de gestión.

SNMP utiliza el esquema jerárquico de nombres desarrollado por ISO. El espacio de nombres forma un árbol, con una raíz conectada a un conjunto de nodos etiquetados
Etiqueta = {breve descripción textual + entero} (ejemplo: iso(1))

Los nodos se agrupan por ramas de objetos relacionados: Identificador de objeto: nombre de un nodo.

Es la secuencia de enteros de las etiquetas de cada nodo, desde la raíz hasta el nodo en cuestión. El identificador es único para cada objeto. Cada nodo representa un recurso, actividad o información relacionada.

El esquema de nombres debe asegurar que dos fabricantes no emplean el mismo nombre para objetos distintos. Se define mediante el SMI (Structure of Management Information).

Obviamente, hay que utilizar algún mecanismo para identificar a los objetos. Así mismo se necesita conocer qué valores puede poseer y qué codificación utiliza. Es decir, cada tipo de objeto debe estar caracterizado por un nombre, una sintáxis y unas reglas de codificación:

1. **Nombre**: Cada ejemplar se representa de forma única por un OID.
2. **Sintaxis:** Para indicar la estructura de los valores que están mantenidos por los objetos gestionados se utiliza una macro de ASN.1.

3. **Reglas de Codificación:** Para la transmisión de valores de los objetos definidos mediante la SMI se aplican directamente las reglas BER (Basic Encoding Rules, Reglas de Codificación Básica).

1.5.3 Estructura de la Información de Gestión (SMI).

La SMI (Structure of Management Information Base), que se especifica en la RFC 1155, define el marco general dentro del cual una MIB se puede definir y construir. La SMI identifica los tipos de datos que pueden usarse en la MIB y especifica cómo se representan y denominan los recursos dentro de la MIB. La filosofía detrás de la SMI es fomentar la simplicidad y la extensibilidad dentro de la MIB. Por eso, la MIB puede solamente guardar tipos de datos simples: escalares y arreglos bidimensionales de escalares. La SMI no soporta la creación o recuperación de estructuras de datos complejas, simplificando así la implementación y mejorando la interoperabilidad.

El SMI define las reglas para describir los objetos gestionados y cómo los protocolos sometidos a la gestión pueden acceder a ellos. La descripción de los objetos gestionados se hace utilizando un subconjunto de ASN.1 ("Abstract Syntax Notation 1, estándar ISO 8824), un lenguaje de descripción de datos. La definición del tipo de objeto consta de cinco campos:

- **Objeto**: nombre textual, llamado *descriptor del objeto*, para el tipo del objeto, junto con su correspondiente *identificador de objeto*, definido abajo.
- **Sintaxis**: la sintaxis abstracta para el tipo el objeto. Las opciones son SimpleSyntax (entero, octeto de caracteres, identificador de objeto, Null), ApplicationSyntax (dirección de red, contador, escala, ticks, opaco) u otro tipo de sintaxis de aplicación (ver el RFC 1155 para más detalles).
- **Definición**: descripción textual de la semántica del tipo.
- **Acceso**: sólo lectura, sólo escritura, lectura - escritura o inaccesible.
- **Status**: obligatorio, opcional u obsoleto.

Como ejemplo, podemos tener:

OBJECT

sysDescr{system 1}

Syntax OCTECT STRING

Definition This value should include the full name and versión identification of the system's hardware type, software operating-system, and networking software. It is mandatory that this contain only printable ASCII characters.

Access read-only.

Status mandatory.

SMI no define información concreta de una tarjeta de red, de un router o de cualquier dispositivo gestionable, si no que define el lenguaje en el que esa información se especifica. Para proporcionar una forma estandarizada de representar la información de gestión, SMI proporciona una técnica estándar para: definir la estructura de una MIB, definir objetos individuales (sintaxis y su valor) y codificar los valores de los objetos.

Un objeto gestionado no sólo ha de ser descrito, también debe ser identificado. Ésto se hace utilizando el identificador de objeto ("Object IDentifier") ASN.1 como si fuera un número de teléfono, reservando grupos de números para distintas localizaciones. En el caso de la gestión de red para TCP/IP, el número reservado fue 1.3.6.1.2 y SMI lo usa como base para la definición de nuevos objetos.

Para proporcionar una forma estandarizada de representación de la información de gestión, la SMI debe proporcionar técnicas para:

- Definir la estructura de una MIB particular.
- Definir objetos individuales, incluyendo la sintaxis y el valor de cada objeto.
- Codificar los valores de los objetos.

Estructura de la MIB.

Todos los objetos gestionados en el ambiente de SNMP están ordenados en una estructura de árbol. Las hojas del árbol son los verdaderos objetos gestionados, cada uno de los cuales representa algún recurso, actividad, o información relacionada que será gestionada. La estructura de árbol por sí misma define un agrupamiento de objetos dentro de grupos relacionados lógicamente.

Asociado con cada tipo de objeto dentro de una MIB hay un identificador de objeto (OID) del tipo OBJECT IDENTIFIER de ASN.1. El OID es único para cada tipo de objeto, y sirve para nombrar al objeto. Su valor consiste en una secuencia de enteros denominados subindentificadores. Como es posible establecer un orden jerárquico a partir de los OID, además de servir para identificar los objetos, los OIDs sirven también para identificar la estructura del árbol.

Comenzando por la raíz (sin nombre) del árbol, cada subidentificador del OID identifica un nodo en el árbol. Hay tres nodos en el primer nivel: ccitt (0), iso (1), joint-iso-ccitt (2).

Bajo el nodo iso, un subárbol es para uso de otras organizaciones, una de las cuales es el U.S. Departament of Defense (dod). La RFC 1155 asume que un subárbol bajo el nodo dod será ubicado por el Internet Activities Board (IAB) como sigue:

internet OBJECT IDENTIFIER ::= {iso (1) org (3) dod (6) 1}

En la figura siguiente vemos como el nodo internet está ubicado debajo del nodo dod.

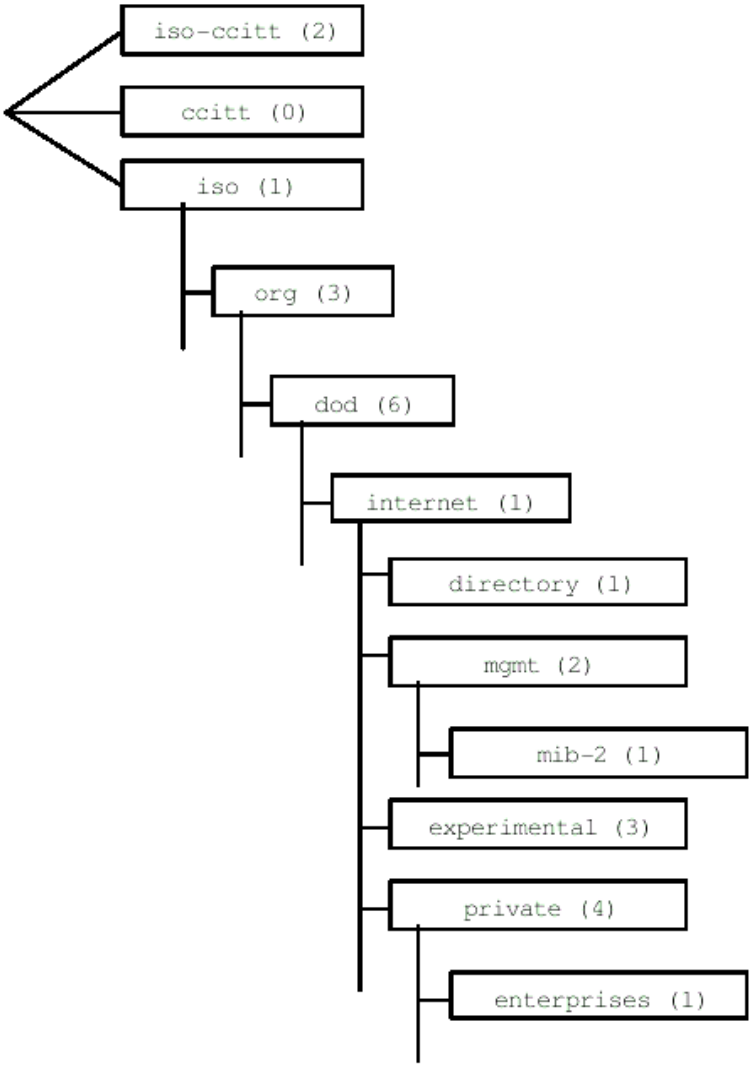


Fig. 10 Parte del árbol de OIDs.

Entonces, el nodo Internet(1) tiene un valor de OID de 1.3.6.1. Este valor sirve como prefijo para los nodos del siguiente nivel inferior del árbol. El documento SMI define cuatro nodos bajo el nodo Internet:

- **directory(1):** reservado para uso futuro con el OSI directory (X.500)
- **mgmt(2):** usado para objetos definidos en documentos aprobados por el IAB.
- **experimental(3):** usado para identificar objetos utilizados en experimentos de Internet.
- **private(4):** usados para identificar objetos definidos unilateralmente por los fabricantes.

El subárbol mgmt contiene las definiciones de las MIBs que han sido aprobadas por el IAB. Actualmente, dos versiones de la MIB han sido desarrolladas, mib-1 y mib-2. La segunda es una extensión de la primera. Ambas fueron provistas con el mismo OID en el subárbol de modo que sólo puede haber una de las MIBs presentes en cualquier configuración.

Pueden definirse objetos adicionales para una MIB en una de las siguientes tres formas:

- Se puede expandir el subárbol mib-2 o reemplazarlo por una nueva revisión (Presumiblemente mib-3). Para expandir mib-2 se define un nuevo subárbol. Por Ejemplo, la remote networking monitoring MIB se define como el decimosexto subárbol bajo mib-2 (mib-2 (16)).
- Puede construirse una MIB experimental para una aplicación particular. Los objetos definidos en ella pueden luego ser movidos hacia el subárbol mgmt. Ejemplos de este caso incluyen varias media transmisión MIBs, una de las cuales es IEEE 802.5 token ring LAN (RFC 1231).
- Pueden agregarse extensiones privadas al subárbol private. Una que esta documentada como RFC es la MUX MIB (RFC 1227).

El subárbol private tiene actualmente definido sólo un nodo hijo, el nodo enterprises(1).

Esta porción del subárbol se usa para permitirle a los fabricantes mejorar la gestión de sus dispositivos y compartir esta información con otros usuarios y fabricantes quienes pueden necesitar interoperar con sus sistemas. Se asigna una rama dentro del subárbol enterprises a cada fabricante que se registre por un OID enterprises.

1.5.4 Tipos de Módulos MIB.

Cada módulo MIB, y los objetos que relaciona, tienen una identificación normalizada llamada OID, es un identificador de objetos, secuencia de números enteros separados por puntos, según su ubicación en el "*árbol de identificación de objetos*" de ISO

Existen tres tipos de módulos MIB:

Estándar: Diseñados por un grupo de trabajo del **IETF** *(Internet Engineering Task Force)* y estandarizado por el **IESG** *(Internet Engineering Steering Group)*. Los prefijos de los identificadores de objetos se encuentran bajo el subárbol *mgmt*.

Experimental: Mientras un grupo de trabajo desarrolla un MIB, los identificadores de objetos temporales se colocan bajo el subárbol *experimental*. Si el MIB adquiere la condición de estándar, se colocan los identificadores bajo el subárbol *mgmt*.

Específico: La mayor parte de las empresas desarrollan módulos MIB propios que soporten ciertas características particulares, las cuales no son generalmente contempladas en los módulos MIB estándar.

1.5.5 Objetos definidos en la MIB.

En la MIB cada recurso se representa mediante un objeto, los tipos de objetos existentes son escalares y tablas bidimensionales.
Los objetos escalares tienen una única instancia, y se identifica con el OID (*Object IDentifiers*) del objeto concatenado con .0, un ejemplo:

nombre del sistema: *system.sysName.0*

Las Tablas tienen las siguientes características:

- Cada fila se identifica por el objeto *Entry: Table.Entry*.
- Cada columna define un objeto columnar, n: *Table.Entry.n*.
- Cada columna tiene una instancia por cada fila de la tabla => valor del objeto.
- Índices: simples o compuestos.
- Identificación de la instancia: identificación del objeto columnar + valor del índice en la fila.

Ejemplo:Tabla de conexiones TCP: *tcp.tcpConnTable* (1.3.6.1.2.1.6.13)

Las operaciones de monitorización consultan el valor de los objetos y las operaciones de control modifican el valor de los objetos.Cada objeto de un dispositivo gestionable por SNMP debe tener un nombre único con el que se le denominará en las operaciones de gestión. El esquema de nombres debe asegurar que dos fabricantes no emplean el mismo nombre para objetos distintos.

La MIB-II (Management Information Base) se divide en varios grupos, los nodos deben implementar todos los objetos del mismo grupo, o ninguno.

Group	Objects for	#
system	basic system information	7
interfaces	network attachments	23
at	address translation	3
ip	internet protocol	38
icmp	internal control message protocol statistics	26
tcp	transmission control protocol	19
udp	user datagram protocol	7
egp	exterior gateway protocol	18
transmiss.	transmission. Media-specific	0
snmp	snmp applications entities	30

#: Number of objects in the group

Fig. 11 MIB – II ("Management Information Base II") - Definición de grupo.

system (1) información sobre el sistema en el que está el agente

interfaces (2) información sobre cada interfaz en un dispositivo de red

at (3) obsoleto

ip (4) información sobre IP, incluyendo encaminamiento (*routing*)

icmp (5): información sobre ICMP (errores, mensajes descartados...)

tcp (6): información sobre TCP, incluyendo tablas de conexiones

udp (7): información sobre UDP (datagramas recibidos, erróneos...)

egp (8): información sobre EGP

cmot (9): vacío, se mantiene por razones históricas

transmission (10): vacío, grupos específicos para medios de transmisión

snmp (11): información sobre rendimiento de SNMP

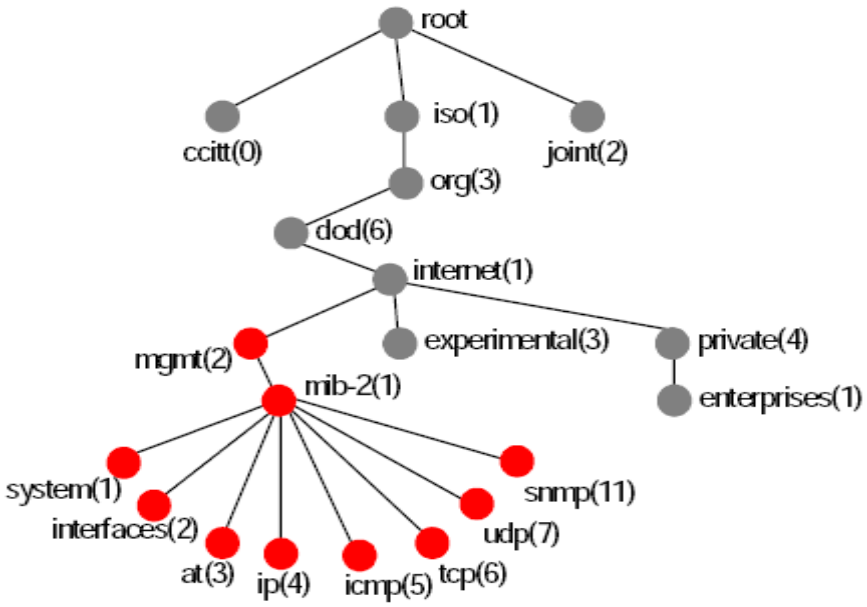


Fig. 12 Árbol OID. Grupos de MIB estándar.

Cada nodo gestionado soporta sólo los grupos apropiados. Por ejemplo, si no hay pasarela, el grupo EGP no tiene por qué estar incluido. Pero si un grupo es apropiado, todos los objetos en ese grupo deben estar soportados.

La arquitectura SNMP opera con un reducido grupo de objetos que se encuentran definido con detalle en la RFC 1066 "Base de Información de Gestión" para la gestión de redes sobre TCP/IP".

La lista de objetos gestionados definidos deriva de aquellos elementos considerados esenciales. Este enfoque, consistente en tomar sólo los objetos esenciales no es restrictivo, ya que el SMI proporciona mecanismos de extensibilidad tales como la definición de una nueva versión de MIB o de objetos privados o no estandarizados.

Debajo están los de objetos de cada grupo. Para más información referirse a la RFC 1213.

- Grupo de sistema: Posee 7 objetos con información sobre el sistema en el que esta el agente (entidad).

  - ✔ sysDescr - Descripción completa del sistema(versión, HW, OS)
  - ✔ sysObjectID - Identificación que da el distribuidor al objeto
  - ✔ sysUpTime - Tiempo desde la última reinicialización.
  - ✔ sysContact - Nombre de la persona que hace de contacto
  - ✔ sysServices - Servicios que ofrece el dispositivo.
  - ✔ sysLocation - localización física.
  - ✔ sysName - nombre del dispositivo.

- Grupo de interfaces: Contiene 23 objetos con información sobre cada interfaz en un dispositivo de red.

  - ✔ ifNumber - número de interfaces de la entidad.
  - ✔ ifTable - tabla de interfaces de la entidad.

- ifEntry - información sobre una interfaz específica
- ifIndex - Número de interfaz.
- ifDescr - Descripción de la interfaz.
- ifType - Tipo de la interfaz.
- ifMtu - Tamaño máximo del datagrama IP.
- ifSpeed - Velocidad en la interfaz en un momento dado (útil para interfaces que cambian de velocidad como modem).
- ifPhysAddress - Dirección física del interfaz.
- ifAdminisStatus - Status de la interfaz..
- ifOperStatus - Estado operacional (up1/down2/testing3).
- ifLastChange - Tiempo que lleva la interfaz en el estado actual.
- ifInOctets/ifOutOctets - Bytes recibidos/enviados.
- ifInUcastPkts/ ifOutUcastPkts - Paquetes unicast.
- ifInNUcastPkts/ ifOutNUcastPkts - Paquetes no-unicast.
- ifInDiscards/ifOutDiscards - Paquetes descartados (error, no conocido...).
- ifInErrors/ifOutErrors - Paquetes con error.
- ifInUnknownProtos - Paquetes recibidos de protocolos desconocidos.
- ifOutQLen - Paquetes en la cola de salida
- ifSpecific - Una referencia a las definiciones de MIB, más específico a los medios de comunicación particulares que se usan para comprender la interfaz.

- Grupo de traducción de direcciones
  - atTable - Tabla de traducción de direcciones
  - atEntry - Cada entrada que contiene una correspondencia de dirección de red a dirección física.
  - atPhysAddress - La dirección física dependiente del medio.
  - atNetAddress - La dirección de red correspondiente a la dirección física.

- Grupo IP: Contiene 38 objetos de gestión, con información relativa al funcionamiento del protocolo IP.
    - ✔ ipForwarding - Indicación de si la entidad es una pasarela IP.
    - ✔ ipDefaultTTL - Valor de TTL por defecto.
    - ✔ ipInReceives – Datagramas recibidos.
    - ✔ ipInHdrErrors - Número de Datagramas de entrada desechados debido a errores en sus cabeceras IP.
    - ✔ ipInAddrErrors - Número de datagramas de entrada desechados debido a errores en sus direcciones IP.
    - ✔ ipForwDatagrams - Datagramas reenviados.
    - ✔ ipInUnknownProtos - Número de datagramas de entrada desechados debido a protocolos desconocidos o no soportados.
    - ✔ ipInDiscards - Datagramas recibidos que se descartan.
    - ✔ ipInDelivers - Datagramas recibidos y entregados al nivel superior (sin errores).
    - ✔ ipOutRequests - Datagramas enviados sin contar los reenviados.
    - ✔ ipOutDiscards - Datagramas de salida descartados.
    - ✔ ipOutNoRoutes - Datagramas descartados por falta de información de rutas.
    - ✔ ipReasmTimeout - Número máximo de segundos que recibieron los fragmentos, mientras se está esperando el reensamblaje a esta entidad.
    - ✔ ipReasmReqds - Datagramas recibidos necesitando reensamblamiento.
    - ✔ ipReasmOKs - Número de datagramas IP reensamblados con éxito.
    - ✔ ipReasmFails - Fragmentos con error en el reensamblado.
    - ✔ ipFragOks - Fragmentos realizados con éxito.
    - ✔ ipFragFails - Fragmentos con errores.
    - ✔ ipFragCreates - Fragmentos generados.
    - ✔ ipAddrTable - Tabla de direcciones de la entidad.
        - ◆ ipAddrEntry – La tabla que trata la información relevante de las direcciones del IP de esta entidad.

- ipAdEntAddr – El IP address a el cual la información de dirección de esta entrada pertenece.
- ipAdEntNetMask –El subnet mask se asoció al IP address de esta entrada. El valor de la máscara es un IP address con todos los pedacitos de la red fijados a 1 y todos los pedacitos de los anfitriones fijados a 0.
- ipAdEntBcastAddr - El valor del pedacito menos significativo en el IP difunden la dirección usada para enviar datagramas en la interfaz (lógico) asociada al IP address de esta entrada.

✔ ipRouteTable - Tabla de rutas IP.

- IpRouteEntry - Una ruta a una destinación particular usada por la entidad en este interfaz (lógico).
- ipRouteDest - El IP address del destino de esta ruta.
- ipRouteIfIndex - El valor de índice que identifica únicamente interfaz local a través del cual el salto siguiente de esta ruta debe ser alcanzado.
- ipRouteMetric1 - El primer encaminamiento métrico para este router.
- ipRouteMetric2 - Una alternativa de encaminamiento para esa ruta.
- ipRouteNextHop - El IP address del salto siguiente de esta ruta.
- ipRouteType - Tipo de ruta.
- ipRouteProto - El mecanismo de encaminamiento vía el cual esta ruta fue aprendida.
- ipRouteAge - Número de segundos que ésa ruta fue puesta al día.
- ipRouteMask – Máscara de subred para el encaminamiento.
- ipRouteInfo - Una referencia a las definiciones de la MIB específicas al protocolo particular de encaminamiento que es responsable de esta ruta, según lo determinado por el valor especificado en el valor del ipRouteProto de la ruta.

✔ ipNetToMediaTable - Tabla de traducción de direcciones.

- ipNetToMediaEntry – Cada entrada contiene una equivalencia “física” de la dirección del IP address.
- ipNetToMediaIfIndex – La interfaz en la cual la equivalencia de esta entrada es eficaz. La interfaz identificada por un valor particular de este índice es la misma interfaz según lo identificado por el mismo valor del ifIndex.
- ipNetToMediaNetAddress – El IP address que corresponde a la dirección “física” dependiente de los medios.

✔ ipRoutingDiscards – Entradas de encaminamiento descartadas.

- Grupo ICMP:Contadores sobre tipos de mensajes ICMP enviados y recibidos
  - ✔ icmpInMsgs - Número de mensajes ICMP recibidos.
  - ✔ icmpInError: mensajes con errores recibidos.
  - ✔ icmpInDestUnreachs - Número de mensajes ICMP "destino inalcanzable"(destination unreachable) recibidos.
  - ✔ icmpInTimeExcds - Número de mensajes ICMP "tiempo excedido” recibidos.
  - ✔ icmpInParmProbs: mensajes “Problema de parámetros” recibidos.
  - ✔ icmpInSrcQuenchs - Número de mensajes ICMP "fuente apagada” recibidos.
  - ✔ icmpInRedirects: mensajes “Redirección” recibidos.
  - ✔ icmpInEchos: mensajes “Eco” recibidos.
  - ✔ icmpInEchoReps: mensajes “Respuesta de eco” recibidos.
  - ✔ icmpInTimestamps: mensajes “Marca de tiempo” recibidos.
  - ✔ icmpInTimestampReps: mensajes “Respuesta marca de tiempo” recibidos.
  - ✔ icmpInAddrMask: mensajes “Petición de máscara de dirección” recibidos.
  - ✔ icmpInAddrMaskReps: mensajes “Respuesta máscara de dirección” recibidos.
  - ✔ icmpIOutMsgs - mensajes icmp enviados.

- ✔ icmpOutError - Número de mensajes ICMP no enviados debido a problemas en ICMP.
- ✔ icmpOutDestUnreachs: mensajes “Destino inalcanzable” enviados.
- ✔ icmpOutTimeExcds: mensajes “Tiempo excedido” enviados.
- ✔ icmpOutParmProbs: mensajes “Problema de parámetros” enviados.
- ✔ icmpIOutSrcQuenchs: mensajes “Fuente apagada” enviados.
- ✔ icmpOutRedirects: mensajes “Redirección” enviados.
- ✔ icmpOutEchos: mensajes “Eco” enviados.
- ✔ icmpOutEchoReps: mensajes “Respuesta de eco” enviados.
- ✔ icmpOutTimestamps: mensajes “Marca de tiempo” enviados.
- ✔ icmpOutTimestampReps: mensajes “Respuesta marca de tiempo” enviados.
- ✔ icmpOutAddrMask: mensajes “Petición de máscara de dirección” enviados.
- ✔ icmpOutAddrMaskReps: mensajes “Respuesta máscara de dirección” enviados.

- Grupo TCP
  - ✔ tcpRtoAlgorithm - Algoritmo que determina el time out para retransmitir octetos para los que no se ha recibido reconocimiento
  - ✔ tcpRtoMin - El valor mínimo permitió por un TCP puesto en práctica para el descanso de la retransmisión, medida en milisegundos.
  - ✔ tcpMaxConn - Límite en el número de conexiones TCP que puede soportar la entidad.
  - ✔ tcpActiveOpens -Número de veces que las conexiones TCP han efectuado una transición directa del estado SYN-SENT al estado CLOSED.
  - ✔ tcpInSegs - Número de segmentos recibidos, incluyendo aquellos con error.
  - ✔ tcpConnRemAddress - La dirección IP remota para ésta conexión TCP .
  - ✔ tcpInErrs - Número de segmentos desechados debido a errores de formato.
  - ✔ tcpOutRsts - Número de resets generados.

- Grupo UDP
  - ✔ udpInDatagrams - Número de datagramas UDP entregados a usuarios UDP
  - ✔ udpNoPorts - Número de datagramas UDP recibidos para los que no existía aplicación en el puerto de destino
  - ✔ udpInErrors - Número de datagramas UDP recibidos que no se pudieron entregar por razones otras que la ausencia de la aplicación en el puerto de destino
  - ✔ udpOutDatagrams - Número de datagramas UDP enviados por la entidad

- Grupo EGP
  - ✔ egpInMsgs - Número de mensajes EGP recibidos sin error
  - ✔ egpInErrors - Número de mensajes EGP con error
  - ✔ egpOutMsgs - Número de mensajes EGP generados localmente.
  - ✔ egpNeighAddr - La dirección IP del vecino de esta entrada EGP
  - ✔ egpNeighState - El estado EGP del sistema local con respecto a la entrada EGP vecino

Esta no es la definición completa del MIB pero sirve de ejemplo de los objetos de cada grupo.

El grupo de interfaces contiene dos objetos de nivel superior: el número de interfaces del nodo (*ifNumber*) y una tabla con información de éstas (*ifTable*). Cada entrada de la tabla (*ifEntry*) contiene los objetos de esa interfaz. Entre ellos, el tipo de interfaz (*ifType*) se identifica en el árbol MIB con notación ASN.1 como 1.3.6.1.2.1.2.2.1.3. Para un adaptador de red en anillo, su valor sería 9.

1.5.6 Denominación de objetos según ISO.

Los objetos se nombran de forma jerárquica según su ubicación en el árbol OID. Cada punto de bifurcación del árbol OID tiene un nombre y un número. Cualquier punto del árbol OID se identifica por la secuencia de nombres (o números) que especifican el camino desde la raíz al punto en cuestión.

Ejemplo: en 1.3.6.1.2.1 se encuentran las definiciones de los módulos MIB normalizados.

Módulos MIB normalizados se encuentran en las RFC 3000

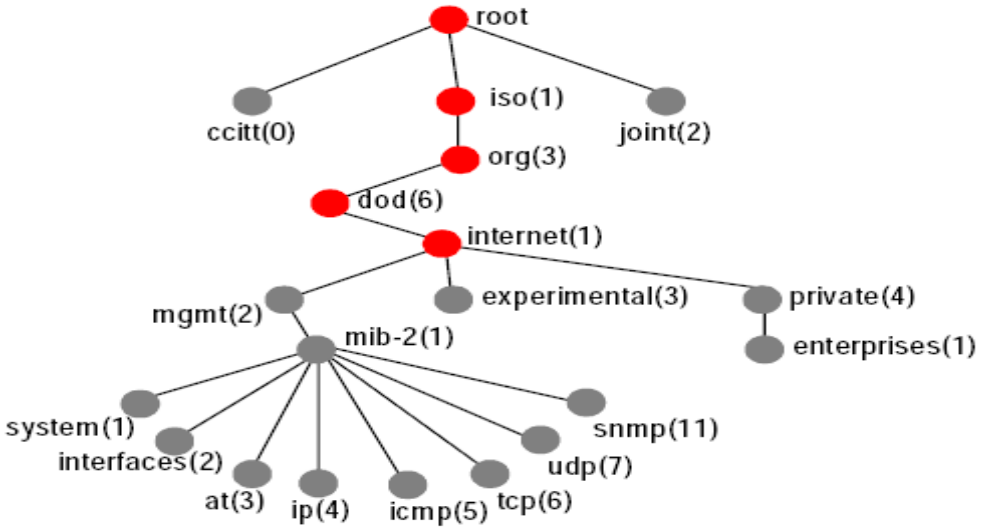


Fig. 13: Estructura de la información de gestión.

Un objeto administrado (algunas veces llamado objeto MIB, objeto, o MIB) es uno de cualquier número de características específicas de un dispositivo administrado. Los objetos administrados están compuestos de una o más instancias de objeto, que son esencialmente variables.

1.6.ASN.1

1.6.1 Fundación del proyecto.

El proyecto ASN.1 (Abstract Syntax Notation 1,Notación Sintáctica Abstracta 1), fue establecido en febrero de 2001 por el grupo de estudio de ITU-T 7 (Unión Internacional de Telecomunicaciones) para asistir a usuarios existentes de ASN.1, dentro y afuera de ITU-T, y promover el uso de ASN.1 a través de una amplia gama de industrias y de cuerpos de los estándares. Desde el 17 de septiembre de 2001, la responsabilidad del proyecto ASN.1 reside con el grupo de estudio 17 y el proyecto ahora abarca los identificadores del objeto (OIDs) y las autoridades del registro (según lo definido en el ITU-T Rec. Serie X.660).

ASN.1 fue desarrollado como parte de la capa 6 (presentación) del modelo de referencia OSI (esta capa define la forma en que los datos serán almacenados en los nodos). Esta notación proporciona un nivel de abstracción similar al ofrecido por lenguajes de programación de alto nivel. ASN.1 es una notación que ofrece un rico conjunto de *tipos de datos* y constructores que permiten definir estructuras de datos complejas a partir de tipos simples o primitivos. Al igual que cualquier lenguaje de programación, la notación es especificada utilizando gramática BNF.

1.6.2 ASN.1 y SNMP.

Uno de los principales usos de ASN.1 es en la definición de objetos SNMP (Protocolo Simple de Gestión de Redes).

ASN.1 es una notación formal usada para describir los datos transmitidos por telecomunicaciones, sin importar la puesta en práctica del lenguaje y la representación física de estos datos.

Esta estructura es recursiva: para cualquier valor ASN.1 que consista de uno o más componentes, el componente value de la TLV se codifica a sí mismo como una o más estructuras TLV. Hay tres métodos de codificar un valor:

- Codificación primitive de longitud definida.
- Codificación constructed de longitud definida.
- Codificación constructed de longitud indefinida.

El método elegido depende del tipo ASN.1 del valor a codificar y del conocimiento o no de la longitud del valor basado en dicho tipo.

1.6.2.1 Reglas de ASN.1.

En este se definen sus propias reglas como:

- Los tipos estándares se escriben en mayúsculas (p.e. INTEGER)
- Los tipos definidos por el usuario comienzan en mayúscula (p.e. Status)
- Los identificadores deben comenzar con minúsculas (p.e. count)
- Los espacios en blanco y tabuladores no son relevantes.
- Los comentarios empiezan y terminan con un guión “-”
- No se permite usar en SNMP los tipos BOOLEAN ni REAL

1.6.2.2 Convenciones en ASN.1

ASN.1 hace distinciones entre mayúsculas y minúsculas de la siguiente forma:

Elemento	Convención
Types	Inicial en mayúscula
Values	Inicial en minúscula
Macros	Todas las letras en mayúscula
Modules	Inicial en mayúscula
ASN.1 keywors	Todas las letras en mayúscula

Caracteres especiales en ASN.1:

Elemento	Nombre
-	Número con signo
--	Comentario
::=	Asignación ("definido como...")
|	Alternativa (opciones de una lista)
{}	Inicio y final de lista
[]	Inicio y final de una etiqueta (tag)
()	Inicio y final de una expresión de subtipo
..	Indica un rango

1.6.2.3 Tipos de Datos en ASN.1.

En ASN.1 se consideran cuatro tipos posibles para un dato:

Universal: INTEGER, BOOLEAN, OCTECT STRING, OBJECT IDENTIFIER, ENUMERATED y NULL.
Específico del Contexto: Definidos para el contexto local en que se usan estos tipos (normalmente el sistema operativo)
Aplicación: Definidos para la aplicación específica
Privado: Definidos por el usuario

Cada uno de los tipos es **Primitivo** (como un único entero) o **Construido** (como un vector de enteros, estructuras y listas).

La gramática para codificar una estructura de datos en ASN.1 tiene el mismo aspecto que la gramática de un lenguaje de alto nivel (C, Pascal).
La sintaxis de transferencia se define especificando cómo se codifican los distintos tipos de datos. La codificación comienza con un tag que especifica el tipo de datos. Cada tipo tiene su propia regla de codificación. Para decodificar este elemento-dato marcado la

capa de presentación destino examina el tag e invoca al procedimiento de decodificación que corresponde al tipo de dato indicado.

Tipos primitivos (simples).

Para mantener la simplicidad de SNMP, la SMI de Internet usa un subconjunto de los tipos de datos de ASN.1. Éstos están divididos en dos categorías: los tipos **primitivos** (Primitive) y los tipos **construidos** (Constructor). Los tipos de datos primitivos (también llamados tipos simples) incluyen INTEGER, OCTET STRING, OBJECT IDENTIFIER y NULL.

INTEGER es un tipo primitivo con valores diferenciados (o únicos) que son números enteros positivos o negativos incluyendo el cero. El tipo INTEGER tiene dos casos especiales. El primero es el tipo entero enumerado, en el cual los objetos tienen un número específico distinto de cero, tal como 1, 2, 3. El segundo, el tipo *integer-bitstring*, es usado por cadenas de bits de longitud corta, tal como (0..127) y muestra el valor en formato hexadecimal.

OCTET STRING es un tipo primitivo cuyos valores son una secuencia ordenada de cero, uno, o más octetos. SNMP usa tres casos especiales del tipo OCTET STRING: el DisplayString, el octetBitstring, y el PhysAddress. En el DisplayString, todos los octetos son caracteres ASCII imprimibles. El octetBitstring es utilizado para cadenas de bits que exceden 32 bits en longitud. (TCP/IP frecuentemente incluye 32 campos de bits. Esta cantidad es un valor típico para el ancho de palabra interna de varios procesadores -hosts y routers- en Internet.). La MIB-II define el PhysAddress y lo usa para representar direcciones de acceso al medio (o capa física).

OBJECT IDENTIFIER es un tipo cuyos valores son el conjunto de todos los identificadores de objetos asignados de acuerdo a las reglas de ISO 8824-1. El tipo ObjectName, es un caso especial utilizado por SNMP, es restringido a los identificadores de objetos de los objetos y subárbol en la MIB.

NULL es un tipo con un único valor, también llamado null. El null sirve como placeholder, pero actualmente no es utilizado por los objetos de SNMP. (NULL es utilizado como placeholder en varios campos de asociación de variables en el PDU GetRequest de SNMP: NULL sustituye el valor de una variable desconocida, es decir el valor que GetRequest está solicitando).

Los datos primitivos ASN.1 permitidos en SNMP son:

Tipo Primitivo	Significado	Código
INTEGER	Entero de longitud arbitraria.	2
BIT STRING	Cadena de cero o más bits.	3
OCTECT STRING	Cadena de cero o más bytes sin signo.	4
NULL	Marcador de lugar.	5
OBJECT IDENTIFIER	Tipo de datos definidos oficialmente.	6

Tipos Construidos o Estructurados

Los tipos construidos(Constructor) o estructurados (Structured), SEQUENCE y SEQUENCE OF, definen tablas y filas (*entries*) dentro de dichas tablas. Por convención, los nombres para los objetos tabla terminan con el sufijo Table, y los nombres para las filas terminan con el sufijo Entry.

SEQUENCE es un tipo estructurado definido mediante la referencia a una lista de tipos *fijos* y *ordenados*. Algunos de los tipos pueden ser opcionales y todos pueden ser diferentes tipos definidos en ASN.1. Cada valor del nuevo tipo consiste en una lista ordenada de valores, uno por cada tipo de componente. SEQUENCE como un todo, *define una fila dentro de un tabla*. Cada entrada (*entry*) en la SEQUENCE especifica una columna dentro de la fila.

SEQUENCE OF es un tipo estructurado que está definido mediante la referencia a un solo tipo existente; cada valor en el nuevo tipo es una lista ordenada de cero, uno, o más valores de dicho tipo existente. Al igual que SEQUENCE, SEQUENCE OF define las filas en una tabla; a diferencia de SEQUENCE, SEQUENCE OF solo usa *elementos del mismo tipo* en ASN.1.

El nombre de la tabla, tcpConnTable, finaliza con el sufijo *Table*. El nombre de fila, tcpConnEntry, finaliza con el sufijo *Entry*. El nombre de secuencia, TcpConnEntry, es semejante al de la fila, excepto por la primera letra que está en mayúscula. La cláusula INDEX define la construcción y orden de las columnas que conforman las filas.

También podemos hacer uso de :

Tipos Referidos: Crea nuevos tipos a partir de otros existentes. Pueden especificarse categorías: universal, aplicación, contexto y privado.

Macros: modelos genéricos que definen prototipos para tipos de datos complejos.

ASN.1 también define la forma de convertir (y decodificar en el receptor) sin ambigüedad los valores expresados con ASN.1 para su transmisión (y recuperación) por la red,
La sintaxis de transferencia se especifica usando las BER (Basic Encoding Rules)
Tras usar las reglas, que muchas veces son recurrentes, llegamos a tener un flujo de objetos primitivos organizados.

Cada valor transmitido se codifica usando 4 campos:

1. Identificador
2. Longitud del campo de datos
3. Campo de datos
4. Indicador de fin de contenido

El **identificador** es un octeto (o más) con tres campos:

- **Etiqueta** (2 bits): 00-Universal, 01-Aplicación, 10-Específico y 11-Privado.
- **Tipo** (1 bit): 0-Primitivo, 1-Construido.
- **Valor Etiqueta** (5 bits): identifican el tipo de valor si está en el rango 0..30. Si es 31 o más estarán los cinco activos (11111) y hay uno o más octetos detrás del primero. Se usan los 7 bits últimos de cada uno en caso de existir; el primer bit de cada octeto adicional es 0 en todos los octetos excepto en el último.

La longitud del campo de datos es un octeto que contiene el número de octetos de datos; el campo de datos contiene los valores de los objetos que se desean transmitir y el indicador de fin de contenido se utiliza si se desconoce el número de datos. Este campo existe en ASN.1 pero está prohibido usarlo en SNMP.

Las cadenas de octetos utilizan big endian (de izquierda a derecha, MSB a la izquierda)
El valor nulo tiene el campo de longitud 0. No transmite ningún valor numérico.

Para su mejor comprensión en la siguiente tabla se muestran Ejemplos:

	ID	LONGITUD	VALOR
Entero 49	00-0-00010	00000001	00110001
Cadena de Octetos ”XY”	00-0-00100	00000010	01111000 01111001
Nulo	00-0-00101	00000000	

Desemejante de muchas otras sintaxis que demanden ser extensibles, ASN.1 ofrece la extensibilidad de la cual trata el problema, y proporciona la ayuda para, ínter trabajar entre los sistemas previamente desplegados y más nuevas, actualizadas versiones diseñadas los años aparte.

ASN.1 envía la información en cualquier forma (audio, vídeo, datos, etc.) dondequiera que él necesita ser comunicado digitalmente. ASN.1 cubre solamente los aspectos estructurales de la información (no hay operadores para manejar los valores una vez que se definan éstos o para hacer cálculos con ellos). Por lo tanto no es un lenguaje de programación.

1.7. Reglas de Codificación Básicas, BER.

Ahora se discutirán las reglas de codificación que permiten que la información sea transmitida en una red. Las Reglas de Codificación Básicas (Basic *Encoding* Rules - **BER**) definen esta sintaxis de transferencia, y así lo especifica el estándar ISO 8825-1.

1.7.1 Descripción.

Las Reglas de Codificación Básicas fueron las reglas originales reemplazadas por el estándar ASN.1_para codificar información abstracta en una corriente de datos concreta. Las reglas, colectivamente referidas como una sintaxis de transferencia en el contexto de ASN.1, especifica las secuencias de octetos exactas las cuales se usan para codificar un elemento de datos dado. La sintaxis define tales elementos como: las representaciones para tipos de datos básicos, la estructura de la información longitud, y los medios para definir tipos complejos o compuestos basados en más tipos primitivos. La sintaxis BER, junto con dos subconjuntos de BER (Canonical Encoding Rules-CER- y Distinguished Encoding Rules-DER-), están definidas por el documento de estándares X.690 de ITU-T, el cual es parte de las series de documentos ASN.1.

El formato BER especifica un formato auto-descriptivo y auto-delimitativo para codificar las estructuras de datos ASN.1. Cada elemento de datos está codificado por un identificador de tipos, una descripción longitud, los elementos de datos actuales, y donde sea necesario, un marcador de fin-de-contenido. Estos tipos de codificaciones son

llamados comúnmente tipo-longitud-valor o codificaciones TLV. Este formato permite a un receptor decodificar la información ASN.1 desde una corriente incompleta, sin necesitar conocimiento previo del tamaño, contenido, o significado semántico de los datos.

1.7.2 Comparación con formatos alternativos.

La diferencia clave entre el formato BER y los formatos CER o DER es la flexibilidad suministrada por las reglas de codificación Básicas. Como dice en el estándar X.690, "Las codificaciones alternativas se permiten por las reglas de codificación básicas como una opción del emisor. Los receptores quienes piden conformidad con las reglas de codificación básicas deben soportar todas las alternativas". Por ejemplo, cuando codificamos un valor construido (esto es, un valor que está compuesto de múltiples valores ya codificados más pequeños), el emisor puede usar una de las tres formas diferentes para especificar la longitud de los datos. Un receptor debe estar preparado para aceptar todas las codificaciones legales para cumplir la conformidad BER. En cambio, ambos CER y DER restringen las especificaciones de longitud disponibles a una opción.

Hay una percepción común de que BER está siendo ineficiente comparado con las reglas de codificación alternativas. Esto se ha respondido argumentando que es principalmente debido a pobres implementaciones, no es algo inherente a las reglas de codificación. Estas implementaciones confían en la flexibilidad que BER suministra para usar lógica de codificación que es más fácil de implementar, pero redunda en una corriente de datos mayor de lo necesario. Si esto es considerado ineficiente, ha dejado varios esquemas de codificación alternativos, tales como Packet Encoding Rules, que intentan mejorar el rendimiento y tamaño de BER.

1.7.3 Utilización.

A pesar de sus problemas, BER es un formato popular para transmitir datos, particularmente en sistemas con codificaciones de datos nativas distintas.

- El protocolo SNMP especifica ASN.1 con BER como su esquema de codificación requerido.
- El estándar de firma digital PKCS #7 también especifica ASN.1 con BER para codificar mensajes cifrados y su firma digital.
- Muchos sistemas de telecomunicaciones, tales como ISDN, y la mayoría de los servicios de teléfono celulares móvil usan ASN.1 con BER en algún grado para transmitir mensajes de control sobre la red.
- Los mensajes LDAP son codificados usando BER.

1.7.4 Codificación de la Información de Administración.

Recordemos que cada máquina dentro de un sistema de administración puede tener su propia representación interna de la información de administración. La sintaxis ASN.1 describe esa información en una forma estándar. La sintaxis de transferencia realiza la comunicación a nivel de bits (la representación externa) entre máquinas. Por ejemplo, asuma que un nodo necesita información de administración de otro dispositivo. La aplicación de administración podría generar un requerimiento SNMP (SNMP request), que se codificaría con las reglas BER y se transmitiría sobre el medio físico de la red. La máquina destino recibiría la información desde la red, la decodificaría usando las reglas de BER y la interpretaría como un comando SNMP. La respuesta SNMP (SNMP response) retornaría la información de una manera similar, pero en forma inversa.

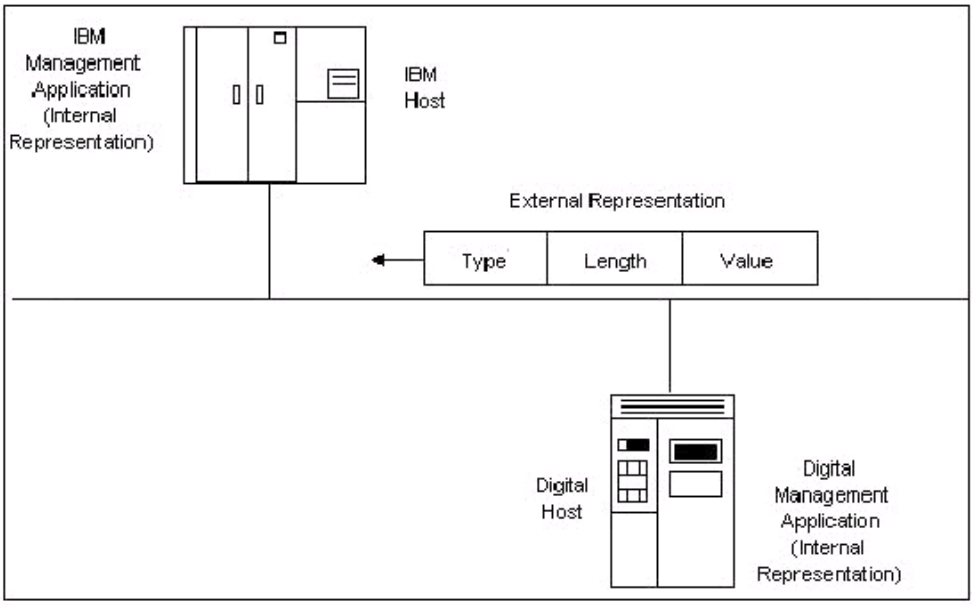


Fig. 14 Representación Interna y Externa de los Datos.

1.7.4.1 Codificación Tipo-Longitud-Valor (TLV).

Para definir la representación de los datos externos, las reglas BER especifican primero la posición de cada bit dentro de los octetos a ser transmitidos. Cada octeto transmite primero el bit más significativo (*Most Significant Bit - MSB*) y lo definen como el bit 8 del lado izquierdo del octeto. El bit menos significativo (*Least Significant Bit - LSB*) se define en el octeto como el bit 1 colocado a la derecha del mismo (figura 15).

La estructura de codificación del dato tiene tres componentes: *Tipo*, *Longitud* y *Valor* (TLV). Note que en la literatura es posible que encuentre otros nombres para TLV, incluyendo Etiqueta-Longitud-Valor (***T**ag-**L**ength-**V**alue*) o Identificador-Longitud-Valor (***I**dentifier-**L**ength-**C**ontents*) (ISO 8825-1). La estructura de una codificación TLV usada con SNMP es mostrada en la figura 16.

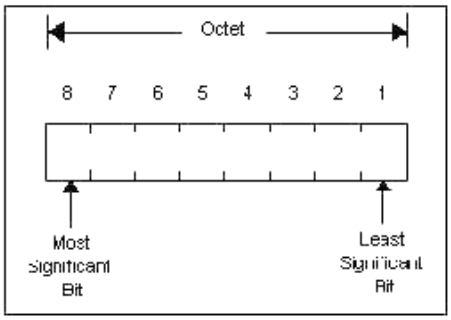


Fig. 15 Orden de los BIT en BER Definidos en ISO 8825-1.

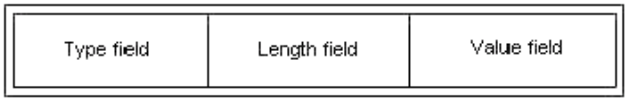


Fig. 16 Codificación Tipo-Longitud-Valor.

Mediante la definición del orden y estructura de los bits, BER garantiza que ambos extremos (nodos) interpreten el flujo de bits de una forma consistente.

Campo Tipo (*Type field*).

El campo *Tipo* va de primero e informa para qué destino es la estructura que sigue. El campo *Tipo* contiene una identificación para la estructura codificada; codifica el tag de ASN.1 (tanto la clase como el número) para el tipo de dato contenido en el campo *Valor*. Un subcampo dentro del campo *Tipo* contiene un bit designado como P/C que indica si la codificación es primitiva (*Primitive*) (P/C = 0) o estructurada (Constructed) (P/C = 1).

Hay dos tipos de campos *Tipo*; su uso depende de la magnitud del número del tag. Cuando el número del tag está entre 0 y 30, el campo Tag contiene un solo octeto. Cuando el número del tag es 31 o superior, el campo *Tipo* está construido con múltiples octetos. En cualquier caso, el primer octeto contiene tres subcampos: Clase (Class), bit P/C y número de tag (tag number). El subcampo Clase codifica la clase de tag en uso:

Clase	Bit 8	Bit 7
Universal	0	0
Application	0	1
Context-specific	1	0
Private	1	1

Las aplicaciones SNMP utilizan las tres primeras clases: ***universal***, ***application*, y *context-specific***. La clase *universal* codifica el tipo INTEGER, el tipo OCTET STRING, etcétera. La clase *application* codifica los tipos definidos (IpAddress, Counter, etcétera). La clase context-specific codifica las cinco unidades de datos de protocolo (PDUs) de SNMP , GetRequest, GetResponse, etcétera.

El subcampo P/C (bit 6) indica la forma del elemento. Codificación primitiva (*Primitive encoding*) (P/C = 0) quiere decir que el contenido de los octetos representan el valor directamente. Una codificación estructurada (*Constructor encoding*) (P/C = 1) significa que el contenido de los octetos codifican uno o más valores de datos adicionales, tal como un SEQUENCE.

¡SNMP utiliza números de tag entre 0 y 30! El número del tag aparece en el tercer subcampo y es representado en binario. El bit 5 es el MSB (bit más significativo) del tag; el bit 1 es el LSB.

ISO 8824-1 contiene números de tag para la clase *universal* (por ejemplo, UNIVERSAL 2 representa el tipo INTEGER). La especificación de la SMI, RFC1155, contiene números de tag para la clase application (por ejemplo, IpAddress es un tipo primitivo con tag [0]). La especificación de SNMP, RFC1157, contiene números de tag para la clase context-specific (por ejemplo, el PDU GetRequest es un tipo construido con tag [0]).

La siguiente lista muestra un resumen de las tres clases de *campos Tipo* usados con SNMP y la codificación para éstos: class, P/C, y tag number. Estas codificaciones

aparecen en notación binaria y hexadecimal, donde la H representa la notación hexadecimal:

Universal Class	Valor del campo Tipo (Type)
INTEGER	00000010 = 02H
OCTET STRING	00000100 = 04H
NULL	00000101 = 05H
OBJECT IDENTIFIER	00000110 = 06H
SEQUENCE	00110000 = 30H
SEQUENCE-OF	00110000 = 30H
Application Class	**Valor del campo Tipo**
IpAddress	01000000 = 40H
Counter	01000001 = 41H
Gauge	01000010 = 42H
TimeTicks	01000011 = 43H
Opaque	01000100 = 44H
Context-Specific Class	**Valor del campo Tipo**
GetRequest	10100000 = A0H
GetNextRequest	10100001 = A1H
GetResponse	10100010 = A2H
SetRequest	10100011 = A3H
Trap	10100100 = A4H

Aunque BER también permite números de tag como 31 o superiores, *SNMP no los utiliza*. Para los números de tag mayores de 31, el campo *Tipo* usa un formato diferente . El número de tag *en el primer octeto* es el binario 11111 (decimal 31), y octetos adicionales se agregan para "trasportar" el número de tag. El *bit 8* = 1 (MBS = 1) de un octeto indica que siguen más octetos; el *bit 8* = 0 (MBS = 0) de un octeto especifica que es el último octeto. Los bits 7 hasta el 1 de cada octeto subsiguiente lleva el entero binario sin signo

del número del tag. El *bit 7* del primer octeto subsiguiente representa el MSB del número del tag.

Campo Longitud (*Length field*).

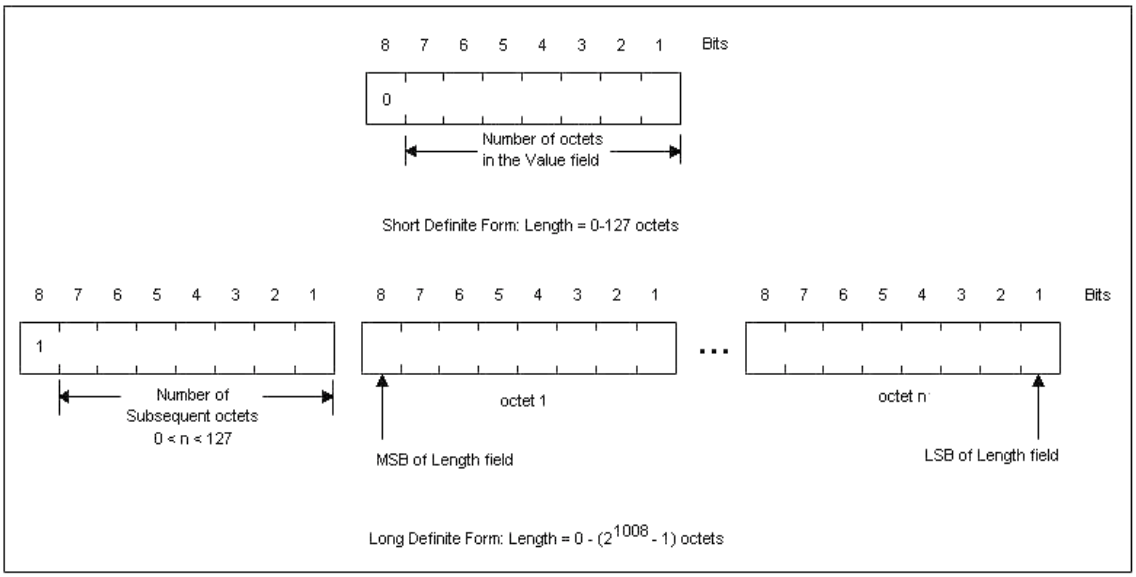


Fig.17 Codificación del Campo Longitud.

El campo *Longitud* sigue al campo *Tipo* y determina el número de octetos que contendrá el campo *Valor* (*Value field*). El campo *Longitud* puede tomar tanto la forma *definida corta* como la *definida larga* (figura 17). (Otra forma, llamada "indefinida", no es utilizada por SNMP). Definido indica que la longitud de la codificación es conocida antes de la transmisión; indefinido indica lo contrario.

La forma *definida corta* indica una longitud de entre 0 y 127 octetos en el campo *Valor*; la forma *definida larga* indica 128 o más octetos en el campo *Valor*, aunque esta puede representar longitudes más cortas.

La forma larga usa múltiples octetos para representar la longitud total: el primer octeto del campo *Longitud* tiene el *bit 8* = 1, seguido por un número binario indicando el número de octetos que siguen. Ese número debe estar entre 1 y 126; el valor 127 está reservado

para extensiones futuras. El *bit 8* del segundo octeto es considerado el MSB del campo *Longitud*, y los octetos siguientes conforman el resto de la *longitud*. De esta manera, la forma *definida larga* puede representar una longitud hasta de 2^{1008}-1 octetos. (El 1008 viene del producto entre 126 y 8: 126 octetos subsecuentes a razón de 8 bits por octeto).

Campo Valor (*Value field*)

El campo *Valor* contiene cero o más octetos, los cuales trasportan los valores de los datos. Ejemplos incluyen un entero, un carácter ASCII, ó un OBJECT IDENTIFIER, tal como { 1.3.6.1.2. }.

1.7.4.2 Codificación del tipo INTEGER.

El tipo INTEGER es un tipo simple que toma valores de cero, enteros positivos o negativos. Es un tipo primitivo codificado con un campo *Valor* que contiene uno o más octetos. Los octetos contenidos son números binarios complemento-2, iguales al valor entero, y pueden usar tantos octetos como sea necesario. Por ejemplo, el entero "75" sería codificado como:

campo Tipo = 02H

campo Longitud = 01H

campo Valor = 4BH

1.7.4.3 Codificación del tipo OCTET STRING.

El **OCTET STRING** es un tipo simple cuyos valores son una secuencia ordenada de cero, uno, o más octetos, cada uno de los cuales debe ser múltiplo de 8 bits. La codificación para valores OCTET STRING es primitiva, con el *campo Tipo* = 04H. El *campo Longitud* y el *campo Valor* dependen de la información codificada.

Se usará la cadena "BBM" en el ejemplo para mostrar la codificación del tipo OCTET STRING (figura 18). El *campo Tipo* contiene 04H, indicando un tipo primitivo OCTET STRING (número de tag igual a 4). El *campo Longitud* indica 3 octetos en el campo Valor. La codificación del *campo Valor* utiliza ASCII.

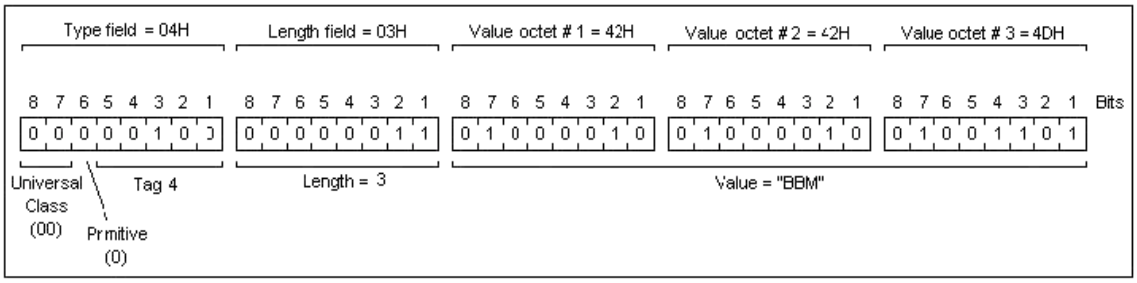


Fig. 18 Codificación del tipo OCTECT STRING,Valor "BBM".

1.7.4.4 Codificación del tipo OBJECT IDENTIFIER.

El **OBJECT IDENTIFIER** nombra (o identifica) ítems. (En SNMP, estos identifican objetos administrados). Su campo *Valor* contiene una lista ordenada de subidentificadores. Para salvar los esfuerzos de codificación y transmisión, se puede tomar ventaja del hecho de que el primer subidentificador es un número pequeño, tal como 0, 1 o 2 y lo combina matemáticamente con el segundo subidentificador, el cual puede ser más grande. El número total de subidentificadores es, por lo tanto, menor que el número de componentes de identificador de objetos en el valor OID a ser codificado. Este número reducido (uno menos) resulta de una expresión matemática que usa los primeros dos componentes OID para producir otra expresión:

Sea X el valor del primer OID, y Y el valor del segundo; El valor del primer subidentificador será: (X * 40) + Y.

Los valores de estos subidentificadores son codificados y localizados en el *campo Valor*. El bit 8 de cada octeto indica si el octeto es el último en la serie de octetos requeridos para describir completamente el valor. Si el bit 8 = 1, al menos uno de los octetos sigue; el

bit 8 = 0 indica el último (o único) octeto. Los bits 7 hasta el 1 de cada octeto codifica subidentificadores. Utilizando un ejemplo del árbol de objetos de la MIB-II, en el grupo System, el OBJECT IDENTIFIER tiene un valor de:

{ iso org(3) dod(6) internet(1) mgmt(2) mib-2 (1) 1 }

Del árbol de objetos, esto está representado por: { 1.3.6.1.2.1.1. }

Observe que, por convención, los subidentificadores son separados por puntos para mayor claridad.

Usando los valores de X=1 y Y=3, el primer valor del subidentificador es: (1 * 40) + 3 = 43

Esto nos lleva a saber que el primer valor de subidentificador es 43, el segundo valor del subidentificador es 6, el tercero es 1, etcétera. El primer valor (43) necesita seis bits para su codificación (que cabe en un octeto: 00101011). El segundo valor (6) necesita 3 bits para la codificación (110), y requiere sólo un octeto. Los valores subsiguientes también requieren un octeto. La codificación tiene de esta forma: *campo Tipo* = 06H (OBJECT IDENTIFIER, tag = 6); campo Longitud = 06H; y campo Valor = 2B 06 01 02 01 01 H.

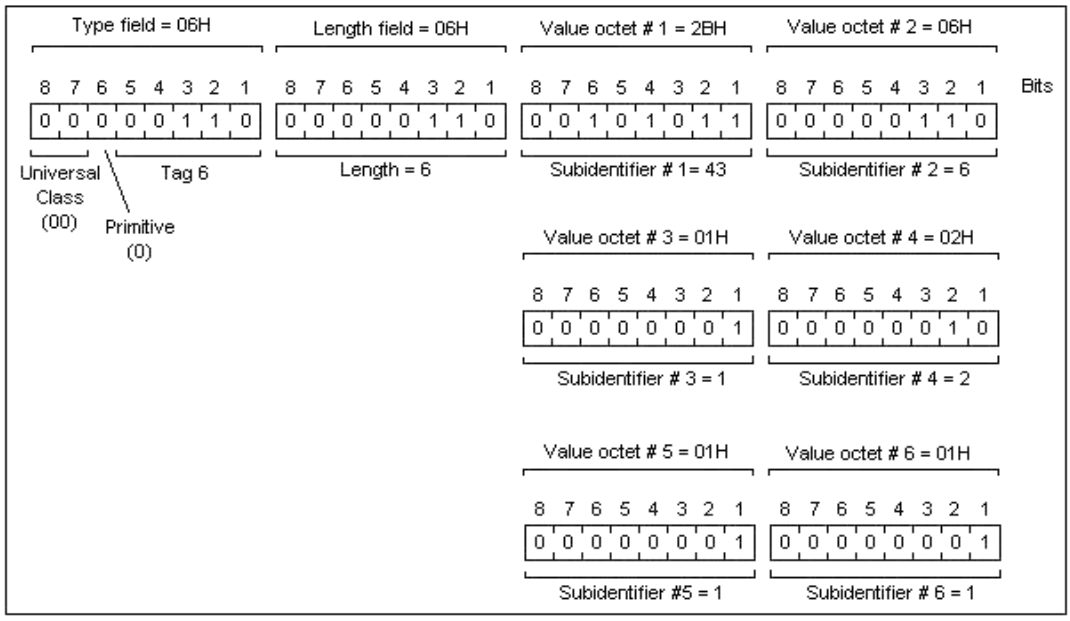


Fig. 19 Codificación para el tipo OBJECT IDENTIFIER,Valor={1,2,3,4,5,6}.

1.7.4.5 Codificación del tipo NULL.

El tipo NULL es un preservador (*placeorder*) que comunica la ausencia de información. Por ejemplo, cuando un administrador requiere el valor de una variable, éste utiliza el tipo NULL como un preservador en la posición que el agente ocupará en la respuesta.

La codificación para el tipo NULL es primitiva. El campo Tipo = 05H, el campo Longitud = 00H. El campo valor es vacío (no hay octetos de valor), como muestra la figura.

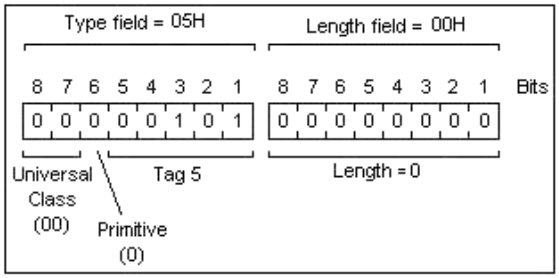


Fig. 20 Codificación del Tipo NULL, Valor NULL.

1.7.4.6 Codificación del tipo SEQUENCE.

Es necesario recordar que el tipo SEQUENCE es una lista de tipos pertenecientes a ASN.1. Un valor de SEQUENCE siempre es codificado en forma estructurada (*Constructed*). Los enlaces de variable usados en los mensajes SNMP proveen un buen ejemplo de SEQUENCE. Los enlace de variable (o VarBind) enlazan un nombre de objeto con su valor, el cual es transmitido dentro del campo Valor, como se muestra a continuacion SNMP (RFC 1157, Pág. 32) define el VarBind:

VarBind ::=
SEQUENCE {
name
ObjectName,
value
ObjectSyntax,

```
}
VarBindList ::=
SEQUENCE OF
VarBind
END
```

Como lo muestra esta sintaxis, el VarBind es un SEQUENCE (enlace) de un nombre, un valor, y la VarBindList es una lista de nombres y valores.

Aquí va un ejemplo: Suponga que Usted necesita la descripción del sistema para un objeto particular cuyo nombre es sysDescr. Para obtener la descripción del sistema, el administrador transmite un SNMP GetRequest al agente preguntando por el valor del objeto sysDescr. El agente responde con un mensaje SNMP GetResponse que contiene el valor, tal como “Retix Local Ethernet Bridge Model 2265M.” El VarBind asocia el objeto (sysDescr) y su valor (“Retix...”).

El primer campo Tipo (30H) indica un tipo estructurado, con Tag = 16 (SEQUENCE). El primer campo Longitud contiene 33H, indicando que sigue el octeto de valor 51. El BER es aplicado entonces para todos los tipos en SEQUENCE. La primera secuencia identifica un tipo primitivo con Tag = 6 (OBJECT IDENTIFIER) y Longitud = 08H. El campo valor contiene la representación numérica del objeto sysDescr {1.3.6.1.2.1.1.1.0}. La segunda secuencia identifica un tipo primitivo con Tag = 4 (OCTET STRING), y Longitud = 27H (39 en decimal). El segundo campo Valor representa el valor del objeto sysDescr (Retix Local Bridge ...”).

1.7.4.7 Codificación del tipo SEQUENCE-OF.

El valor del tipo SEQUENCE-OF es codificado en forma estructurada y de la misma manera que el tipo SEQUENCE.

1.7.4.8 Codificación del tipo IPADDRESS.

Esta discusión se traslada ahora a las codificaciones que hacen uso de la clase application. Puesto que todas estas codificaciones son de clase application (01) y primitivas (P/C = 0) ,con números de tag entre 0 y 4, los campos Tipo estarán en un rango de 40 a 44H (ver figuras 21 a la 24).

La SMI define el tipo IpAddress. El IpAddress lleva una dirección IP de 32 bits, la cual está representada en cuatro octetos. Saltando a la discusión sobre la MIB, el grupo IP contiene objetos que relacionan los procesos IP en un router o un host. Un objeto llamado IpAdEntAddr identifica la dirección IP que está relacionada a la información subsiguiente. Para codificar la IpAdEntAddr (ver figura 21), el campo Tipo es puesto a 40H (clase solicitud, Tag = 0). El campo Longitud = 4, representando los cuatro octetos en la dirección IP. El campo Valor contiene cuatro octetos, los cuales transportan la dirección IP en notación decimal punteada. Para la dirección mostrada en el ejemplo (128.150.161.8), el primer octeto en el campo Valor contiene el binario equivalente a 128 (10000000), el segundo, el binario equivalente a 150, y así sucesivamente.

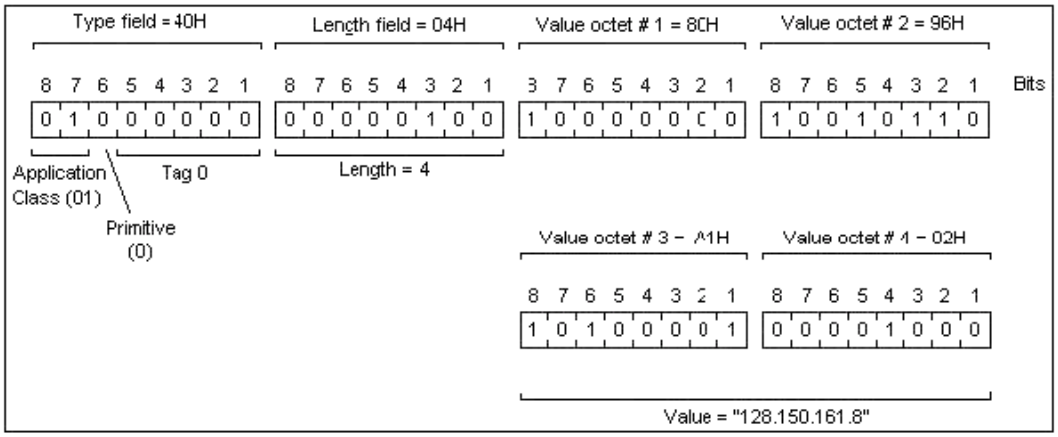


Fig. 21 Codificación del Tipo IpAddress, Valor= "128.150.161.8".

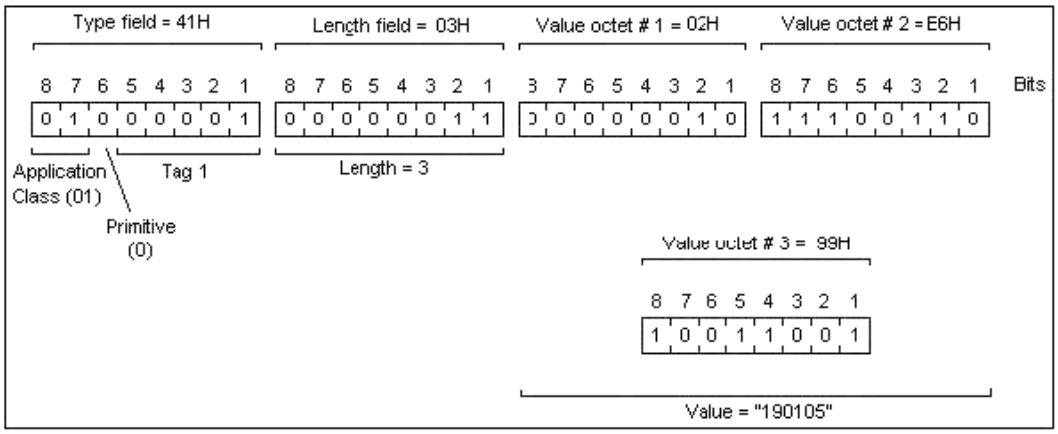


Fig. 22 Codificación del Tipo Counter, Valor "190105".

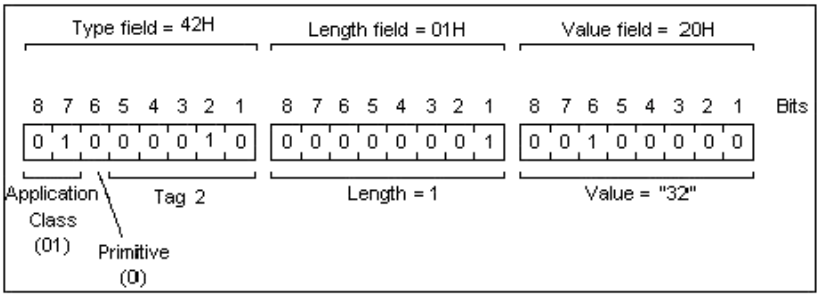


Fig. 23 Codificación del Tipo Gauge, Valor "32".

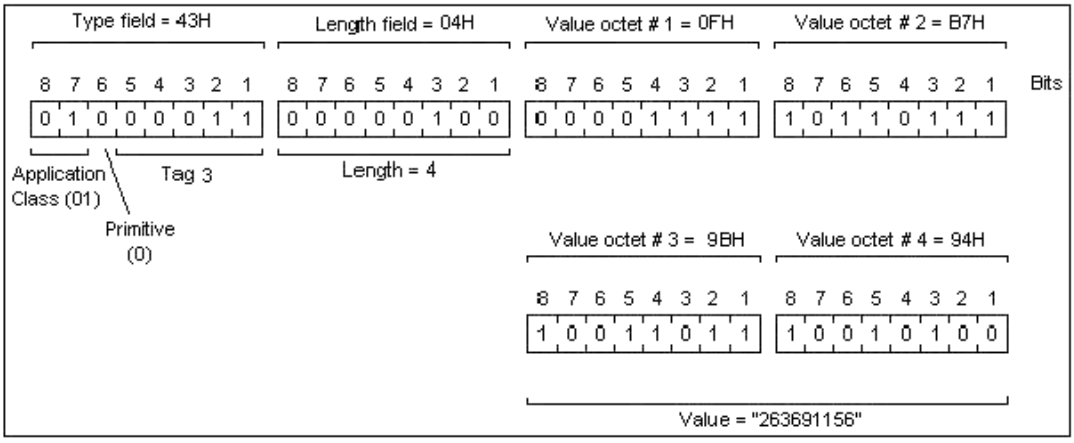


Fig. 24 Codificación del Tipo Times Ticks, Valor "263691156".

1.7.4.9 Codificación del tipo COUNTER.

Un tipo Counter representa un entero no negativo que se incrementa monótonamente hasta alcanzar un valor máximo de 4'294.967.295 luego se reinicia y vuelve a comenzar desde cero. El grupo ICMP utiliza muchos contadores para registrar estadísticas de mensajes. Un objeto, icmpInMsgs, registra el número de mensajes que el proceso ICMP ha recibido en un router o un host. Una codificación de muestra debería tener un campo Tipo = 41H, representando la clase solicitud, la codificación primitiva y Tag = 1. El valor (190.105) requiere tres octetos. El campo Longitud es, por lo tanto, 03H, y el campo Valor contiene 02 E6 99H, representando los 190.105 mensajes.

1.7.4.10 Codificación del tipo GAUGE.

Un tipo Gauge es un entero no negativo que puede incrementar o decrementar su valor, pero hasta un valor máximo de 4'294.967.295 . El Gauge no es usado frecuentemente. La MIB-II lo define para los objetos ifSpeed, ifOutQLen y tcpCurrEstab únicamente. Por ejemplo, la figura 23 asume que la máxima longitud de cola de salida de una interfaz particular es 32 paquetes. Para codificar este valor Gauge, el campo Tipo es fijado a 42H (clase solicitud, Primitiva, Tag = 2). Un octeto codifica el decimal 32; por lo tanto:

El campo Longitud = 01H
El campo Valor contiene 20H
El valor 32 en decimal deseado.

1.7.4.11 Codificación del tipo TIMETICKS.

El tipo TimeTicks contiene una marca de tiempo (*time-stamp*) que mide el tiempo transcurrido (en centésimas de segundos) desde algún evento. El objeto sysUpTime mide el tiempo desde que la entidad de administración de la red sobre un dispositivo fue

reinicializado. El campo Tipo debería ser fijado a 43H (clase solicitud, Primitiva, Tag = 3). Cuatro octetos representan un valor igual a 263691156. Por lo tanto, El campo Longitud contiene 04H. Los cuatro octetos en el campo Valor contienen la representación binaria del valor TimeTicks.

1.7.4.12 Codificación de CONTEXT-SPECIFICS para SNMP.

La clase final de codificación discutida es la codificación de context-specific, la cual es utilizada en el contexto de SNMP. Cinco unidades de dato de protocolo (PDUs), transportan la información de SNMP. Los PDUs son GetRequest, GetNextRequest, GetResponse, SetRequest y Trap. Estos PDUs tienen números de tag de 0 a 4 respectivamente. Estas codificaciones son todas de la clase context-specific (10) y estructurada (P/C = 1). El campo Tipo, de esta manera, tiene valores en el rango de A0 hasta A4H. Los campos Valor y Longitud dependen de la información transportada.

1.7.5 Ejemplos de codificación y decodificación.

1. Codifique la petición de un mensaje SNMPv1 snmpget con los siguientes parámetros:

   Comunidad: publica.

   Request-id: 33 46 2A 3B en hexadecimal.

   OIDs: icmpInMsgs, icmpInErrors, icmpInDestUnreachs.

Solución:

30 46 —-long 70
  02 01 --versión
    00
  04 07 -- Comunidad
    70 75 62 6C 69 63 61
  A0 38 —- GetRequest
    02 04 —-request-id
      33 46 2A 3B
    02 01 —-error-status
      00
    02 01 – error-index
      00

30 2A -- SEQUENCE OF VarBinds
  30 0C -- VarBinds
    06 08 -- OID
      2B 06 01 02 01 05 01 00
    05 00 --NULL
  30 0C -- VarBinds
    06 08 -- OID
      2B 06 01 02 01 05 02 00
    05 00 --NULL
  30 0C – VarBinds
    06 08 -- OID
      2B 06 01 02 01 05 03 00
    05 00 –NULL

2. Un agente recibe un mensaje SNMP solicitando los valores de 1.3.6.1.2.1.1.2.0 y 1.3.6.1.2.1.4.1.0; utiliza la comunidad public (61 6c 75 6d 6e 6f en ASCII) y el request-id del mensaje es: 35467A5C h. Con la comunidad public, la vista a la que tiene acceso el agente incluye 1.3.6.1.2.1.1 y no incluye 1.3.6.1.2.1.4. Genere la respuesta codificada en BER según la norma SNMPv1. Utilice formato largo para indicar la longitud del mensaje, de la PDU y de variable-bindings.

Solución:

30 82 00 3B --SEQUENCE
02 01 -- INTEGER
00 –- SNMP VERSION 1
04 06 –- OCTECT STRING
61 6c 75 6d 6e 6f –- COMUNIDAD
A2 82 00 2C –- GET_RESPONSE
02 04 –- INTEGER
35 46 7A 5C – REQUEST_ID
02 01 –- INTEGER
02 –-ERROR STATUS = NOSUCHNAME
02 01 --INTEGER
02 –-ERROR INDEX=2
30 82 00 1C -- SEQUENCE
30 OC --SEQUENCE
06 0A --OID
2B 06 01 02 01 01 02 00 –- NAME = 1.3.6.1.2.1.1.2.0 -- udpInDatagrams
05 00 –NULL
30 0C --SEQUENCE
06 0A --OID
2B 06 01 02 01 04 01 00 –- NAME = 1.3.6.1.2.1.4.1.0
-- udpInDatagrams
05 00 –NULL

3. Decodifique la siguiente secuencia correspondiente a un mensaje SNMP: 30 82 00 32 02 01 00 04 06 70 75 62 6C 69 63 A2 82 00 23 02 04 59 57 65 EF 02 01 00 02 01 00 30 82 00 13 30 82 00 0F 06 08 2B 06 01 02 01 04 09 00 41 03 01 92 7C

Solución:

30 82 00 32 -- UNIVERSAL SEQUENCE, constru, long 50

02 01 --UNIVERSAL INTEGER, primitivo, long 1

00 --valor 0 (versión SNMP 1)

04 06 -- UNIVERSAL OCTECT STRING, primitivo, long 6

70 75 62 6C 69 63 –- valor “public”

A2 82 00 23 –- [2] long 35,constru, get-response

02 04 – UNIVERSAL INTEGER , long 4,

59 57 65 EF -- request-id 1498899951

02 01 –- UNIVERSAL INTEGER , long 1

00 -- error-status = noError

02 01 –-UNIVERSAL INTEGER , long 1

00 –- error-index = 0

30 82 00 13 –- UNIVERSAL SEQUENCE, constru, long 19

30 82 00 0F -- UNIVERSAL SEQUENCE, constru, long 15

06 08 -- UNIVERSAL OBJECTS IDENTIFIER, long 8

2B 06 01 02 01 04 09 00 --1.3.6.1.2.1.4.9.0

4103 –- [APPLICATION 1], primitivo, long 3

01 92 7C –- Counter 103036 ver RFC 1155-SMI

Respuesta a una petición del valor de ipInDelivers.

4. Decodifique la siguiente secuencia correspondiente a un mensaje SNMP:
   3082 0049 0201 0004 0670 7562 6C69 63A1 8200 3A02 0403 A0BD FE02 0100 0030 8200 2A30 8200 0A06 062B 0601 0201 0405 0030 8200 0A06 062B 0601 0201 0105 0030 8200 0A06 062B 0601 0201 0B05 00

Solución:

3082 0049
--et UNIVERSAL, construido; SEQUENCE, longitud formato largo
--73 octetos de longitud
0201 00
--et tipo primitivo INTEGER, et UNIVERSAL longitud 1, valor 0, SNMP versión 1
04 06
--et UNIVERSAL tipo primitivo OCTECT STRING, longitud 6 octeto
70 7562 6C69 63 --PUBLIC

A1 8200 3A
--et especifica de contexto 1, GetNextRequest-PDU, longitud 58
02 04
--tipo primitivo INTEGER, et UNIVERSAL, longitud 4
03 A0BD FE --Request_id
02 0100
--tipo primitivo INTEGER, et UNIVERSAL, longitud 1, valor 0. Error_status valor 0
0201 00
----tipo primitivo INTEGER, et UNIVERSAL, longitud 1, valor 0. Error_index valor 0
30 8200 2A
--tipo primitivo, construido; SEQUENCE OF(VarBindList). longitud formato largo
--42 octetos
30 8200 0A
--et UNIVERSAL, tipo construido; SEQUENCE (VarBind), longitud formato largo
--10octetos
06 06
--et UNIVERSAL, tipo OBJECT IDENTIFIER, primitivo, longitud 6 octetos
2B 0601 0201 04
--valor 1.3.6.1.2.1.4
05 00 --valor NULL
30 8200 0A
--et UNIVERSAL, tipo construido; SEQUENCE (VarBind), longitud formato largo
--10octetos
06 06
--et UNIVERSAL, tipo OBJECT IDENTIFIER, primitivo, longitud 6 octetos
2B 0601 0201 01
--valor 1.3.6.1.2.1.1

05 00 --valor NULL
30 8200 0A
--et UNIVERSAL, tipo construido; SEQUENCE (VarBind), longitud formato largo
--10octetos
06 06
--et UNIVERSAL, tipo OBJECT IDENTIFIER, primitivo, longitud 6 octetos
2B 0601 0201 0B
--valor 1.3.6.1.2.1.11
05 00 --valor NULL

1.7.6 Comparación de ASN.1 y BER.

ASN.1 es como un lenguaje de programación (como C), mientras que BER es como un compilador para ese lenguaje. Los compiladores son específicos de la plataforma, mientras que muchos lenguajes de programación de alto nivel no lo son. C define las reglas y el lenguaje para escribir un programa. Un programa no es C; está escrito en C. El programa no es útil hasta que no se compila para una plataforma determinada (como Intel x86). Lo mismo ocurre con ASN.1 y BER. ASN.1 es el lenguaje para escribir un estándar. Un estándar no es ASN.1; está escrito en ASN.1. Los datos que genera un programa que cumple el estándar pueden denominarse "datos ASN.1". Los datos ASN.1 no son útiles (es decir, no se pueden transmitir a través de una LAN) hasta que no se codifican en una secuencia de octetos que se puede descodificar fácilmente en el destino.

2. Realización de Prácticas para Gestión de Redes.

2.1 Instructivo para la realización de las prácticas.

Antes de realizar la práctica se dará una explicación de como poder realizarla:

Como realizar el archivo de configuración:

1. Se deben crear las listas de control de acceso correspondientes que definirán quién tendrá acceso al servicio snmpd. Por razones de seguridad solo la interfaz 127.0.0.1 se le da permiso de lectura escritura. Un ejemplo es el siguiente:

   com2sec local 127.0.0.1/32 Cl4v3Acc3s0

   En lo anterior la línea significa que habrá una lista de control de acceso denominada «local» y que corresponderá solo a 127.0.0.1/32, asignando Cl4v3Acc3s0 como clave de acceso.

2. Se crean grupos: ***MyRWGroup***. Éste será un grupo al que se le asignará mas adelante permisos de lectura escritura. Por cada grupo se asignan tres líneas que especifican el tipo de acceso que se permitirá en un momento dado a un grupo en particular. Es decir, ***MyRWGroup*** se asocia a ***local.***

   #Se asigna local al grupo de lectura escritura

   group MyRWGroup v1 local

   group MyRWGroup v2c local

   group MyRWGroup usm local

3. Se especifican las ramas que se van a permitir ver a través del servicio. Lo mas común, para, por ejemplo, utilizarse con MRTG, es lo siguiente:

   view all included .1 80

4. Se debe especificar que permisos tendrán los dos grupos, en éste caso MyRWGroup. Son de especial interés las últimas columnas.

   ## group context sec.model sec.level prefix read write notif

   access MyRWGroup "" any noauth exact all all all

5. Se definen dos parámetros de carácter informativo para que cuando utilicen aplicaciones cliente como MRTG se incluya algo de información acerca de que sistema se está accediendo.

   syslocation Servidor Linux en AMDK6.linuxparatodos.com.mx

   syscontact Administrador (fulano@algun-dominio.net)

Ahora se dará una explicación de las ordenes utilizadas

- **snmpget**

La orden snmpget se puede utilizar para obtener datos de un host remoto dado su nombre de host, la información y un OID. Ejemplo:

```
# snmpget -c public -v 1 localhost system.sysUpTime.0
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (57693) 0:09:36.93
```

public: la comunidad SNMP.
1: version snmp a utilizar.
localhost: es el nombre del equipo con el que queremos hablar,
system.sysUpTime.0: corresponde al OID.

- **snmgetnext**

Esta órden es similar al comando snmpget se utiliza para obtener el siguiente OID en el árbol de datos del MIB. En lugar de obtener los datos que se solicitan directamente, devuelve el siguiente OID en el árbol y su valor:

Ésta órden también se puede utilizar para recorrer de forma manual el árbol de las MIB en un host remoto, especificando siempre el último OID aparecido en la línea de órdenes para la siguiente órden:

 # snmpgetnext -v 1 -c public localhost system.sysUpTime.0

SNMPv2-MIB::sysContact.0 = STRING: Roxana

El siguiente OID de sysUpTime es sysContact.0 que es el que muestra en el resultado

- **snmpwalk**

Ésta órden realiza una serie completa de getnexts automáticamente y se detiene cuando devuelve resultados que no están en el rango del OID especificado originalmente. Por ejemplo, si quisiéramos obtener toda la información almacenada en el grupo system del MIB de una máquina podríamos hacerlo utilizando esta órden:

 # snmpwalk -v 1 -c public localhost system

SNMPv2-MIB::sysDescr.0 = STRING: Linux linux-roxana 2.6.18.2-34-default #1 SMP Mon Nov 27 11:46:27 UTC 2006 i686

SNMPv2-MIB::sysObjectID.0 = OID: NET-SNMP-MIB::netSnmpAgentOIDs.10

DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (126563) 0:21:05.63

NMPv2-MIB::sysContact.0 = STRING: Roxana

SNMPv2-MIB::sysName.0 = STRING: linux-roxana

SNMPv2-MIB::sysLocation.0 = STRING: Nerlyng

SNMPv2-MIB::sysORLastChange.0 = Timeticks: (6) 0:00:00.06

- **snmptable**

Ésta órden muestra una tabla SNMP en un formato de filas y columnas de manera que su visionado y comprensión es más sencillo que si utilizamos la órden snmpwalk.

 # snmptable -v 1 -Cw 80 -c prueba localhost iftable

SNMP table: IF-MIB::ifTable

ifIndex	ifDescr	ifType	ifMtu	ifSpeed	.....
1	lo	softwareLoopback	16436	10000000	.....
2	eth0	ethernetCsmacd	1500	0	.....

3	sit0	tunnel	1480	0	.....

- **snmpset**

Ésta órden se utiliza para modificar información en un host. Por cada una de las variables que se quiere establecer, es necesario el OID a actualizar, el tipo de datos de la variable y el valor al que queramos poner la variable. Podemos obtener los tipos de datos válidos del siguiente modo:

#snmpset -h | tail

TYPE: one of i, u, t, a, o, s, x, d, b, n

i: INTEGER, u: unsigned INTEGER, t: TIMETICKS, a: IPADDRESS

o: OBJID, s: STRING, x: HEX STRING, d: DECIMAL STRING, b: BITS

U: unsigned int64, I: signed int64, F: float, D: double

2.2 Resolución de práctica 1.

Configuración de un agente de gestión.

Herramientas necesarias

- Ordenador con sistema operativo Linux (se utiliza Suse10.2)
- Paquete de gestión Net-Snmp.

Ejercicio1.

Hacer un fichero de configuración local snmpd.conf. Puede basarse en el fichero de ejemplo.

Ejemplo de configuración sencilla:

```
com2sec  npublic      172.29.20.0/24        public
group    gpublic      v1                    npublic
view     todo         included              mib-2
```

access gpublic "" v1 noauth exact todo none none

El fichero de configuración final (snmpd.conf) deberá tener la siguiente configuración:
Acceso permitido a cualquier gestor con dirección 192.168.1.0
Vistas:

- Una vista denominada **todo** donde se vea las mib-2 excepto snmp.
- Una vista denominada **protocolos** donde se vea interfaces, ip, snmp, icmp, tcp y udp.
- Una vista denominada **sistema** donde se vea system.

Acceso de comunidades:

- Comunidad **private** con acceso de lectura y escritura en la vista sistema.
- Comunidad **public** con acceso de solo de lectura en la vista todo.
- Comunidad **adminet** con acceso lectura y escritura en la vista protocolos.

Configurar el valor de las instancias del grupo system:

- syscontact
- syslocation

Solución

```
# Reglas de control de acceso al agente, establece quién puede conectarse, permisos
# de lectura, escritura, qué ramas puedes ver, etc.

# Sólo será posible acceder al agente SNMP desde el host
#            sec.name          source        community
com2sec    private_sec       localhost     private
com2sec    public_sec        default       public
com2sec    adminet_sec       localhost     adminet
com2sec    prueba_sec        default       prueba
```

```
group MyRWGroup   v1          private_sec
group MyRWGroup   v2c         private_sec
group MyRWGroup   usm         private_sec
group MyROGroup   v1          public_sec
group MyROGroup   v2c         public_sec
group MyROGroup   usm         public_sec
group MyRWGroup2 v1         adminet_sec
group MyRWGroup2 v2c        adminet_sec
group MyRWGroup2 usm        adminet_sec
group MyPrueba        v1      prueba_sec
group MyPrueba v2c            prueba_sec
group MyPrueba usm            prueba_sec

# Ramas MIB que se permiten ver
#          incl/excl subtree                          mask
view all   included  .1                               80
view  todo included  .1.3.6.1.2.1
view  todo excluded  .1.3.6.1.2.1.11
view  protocolos included .1.3.6.1.2.1
view  protocolos excluded .1.3.6.1.2.1.1
view  protocolos excluded .1.3.6.1.2.1.3
view  protocolos excluded .1.3.6.1.2.1.8
view  protocolos excluded .1.3.6.1.2.1.9
view  protocolos excluded .1.3.6.1.2.1.10
view sistema included .1.3.6.1.2.1.1
#Establece permisos de lectura y escritura
#        group    context   sec.model  sec.level  match    read          write          notif
access MyPrueba  ""         any       noauth    exact      all           all            all
access MyROGroup ""         any       noauth    exact      todo          none           none
access MyRWGroup2 ""        any       noauth    exact      protocolos    protocolos     none
```

access MyRWGroup "" any noauth exact sistema sistema none

System contact information
syscontact Roxana
syslocation Nerling

Este archivo de configuración deberá ser copiado en /etc/snmp/, y deberá ser llamado snmpd.conf

Ejercicio 2.

Arrancar el agente.

Existen 2 formas de arrancar el agente:

- Esta es la manera mas común de arrancar el agente, solo deberá ejecutar el siguiente comando, por defecto el puerto que utiliza es el 161. El fichero de configuración que utiliza es el que se encuentra en /etc/snmp/snmpd.conf

linux-snmp:~ #/etc/init.d/snmpd start

- Arrancar el agente (snmpd) con los ficheros de configuración locales y en el puerto 1500 (172.29.20.xx:1500). /usr/sbin/snmpd (ver parámetros en la ayuda).

linux-snmp:~ #/usr/sbin/snmpd -c ./snmpd.conf -f -d 127.29.20.xx:1500

Se recomienda con la opción –f para que el gestor no devuelva el control al sistema, y –d para visualizar el formato de los mensajes recibidos de las respuestas generadas. Si tiene problemas con el agente utilice la opción –D. El fichero de configuración local se carga con la opción –c ./mysnmpd.conf. Forzar a la una dirección y puerto172.29.20.xx:1500.

Ejercicio 3.

Realizar consultas con el comando snmpget de los objetos del grupo system y el valor que tienen.

En localhost sucede esto:

linux-snmp:~ # snmpget -v 1 -c public localhost system.1.0

SNMPv2-MIB::sysDescr.0 = STRING: Linux linux-snmp 2.6.18.2-34-default #1 SMP Mon Nov 27 11:46:27 UTC 2006 i686

linux-snmp:~ # snmpget -v 1 -c private localhost system.1.0

SNMPv2-MIB::sysDescr.0 = STRING: Linux linux-snmp 2.6.18.2-34-default #1 SMP Mon Nov 27 11:46:27 UTC 2006 i686

Aquí no puede acceder debido a que la comunidad adminet solo puede ver los grupos interfaces, ip, snmp, icmp, tcp y udp. Ver el archivo de configuración snmpd.conf

linux-snmp:~ # snmpget -v 1 -c adminet localhost system.1.0

Error in packet

Reason: (noSuchName) There is no such variable name in this MIB.

Failed object: SNMPv2-MIB::sysDescr.0

En otra máquina de la red

linux-snmp:~ # snmpget -v 1 -c public 192.168.1.120 system.1.0

SNMPv2-MIB::sysDescr.0 = STRING: Linux linux-snmp 2.6.18.2-34-default #1 SMP Mon Nov 27 11:46:27 UTC 2006 i686

A continuación el siguiente comando da un error debido a la configuración que se tiene de snmpd.conf, la comunidad private solo pude ser vista por el localhost, no por otra maquina

linux-snmp:~ # snmpget -v 1 -c private 192.168.1.120 system.1.0

Timeout: No Response from 192.168.1.120.

El error siguiente da debido a 2 motivos, primero adminet solo pude ser vista por la propia máquina (localhost), y segundo adminet no tiene como grupos a system

linux-snmp:~ # snmpget -v 1 -c adminet 192.168.1.120 system.1.0

Timeout: No Response from 192.168.1.120.

Ejercicio 4.

Cambiar el valor de system.sysName.0 con el comando snmpset.

linux-snmp:~ # snmpset -v 1 -c private localhost sysName.0 s "ELKIS"

SNMPv2-MIB::sysName.0 = STRING: ELKIS

Ejercicio 5.

Comprobar qué sucede si intenta:

- Leer una instancia de una vista con una comunidad que no tiene permisos de lectura.

  Con la comunidad private únicamente se puede leer y modificar los datos del grupo system. Ejemplo:

  linux-snmp:~ # snmpwalk -v 1 -c private localhost snmp

  End of MIB

- Modificar una instancia de una vista con una comunidad que no tiene permisos de escritura ((p.e. system.sysName.0)).

  La comunidad **public** que tiene acceso de solo lectura a la vista **todo,** podrá ver sus objetos menos modificarlos. Ejemplo

  linux-snmp:~ # snmpset -v 1 -c public localhost sysName.0 s "Linux"

  Error in packet.

  Reason: (noSuchName) There is no such variable name in this MIB.

- Modificar una instancia que no es de escritura, con una comunidad con la que se tiene permiso de escritura (p.e.system.sysDescr.0).

  l**inux-snmp:~ # snmpset -v 1 -c private localhost sysDescr.0 s "Elkis"**

Error in packet.
Reason: (noSuchName) There is no such variable name in this MIB.
Failed object: SNMPv2-MIB::sysDescr.0

- Leer o modificar una instancia con una comunidad no definida.
  Intentaremos modificar el valor de s**ysName** con la comunidad **nueva**, la cual no esta definida en el archivo de configuración snmpd.conf.
  Ejemplo:
  linux-snmp:~ # snmpset -v 1 -c nueva localhost sysName.0 s "ELKIS"
  Timeout: No Response from localhost

Ejercicio 6.

Consultar la tabla de interfaces (interfaces.ifTable) con el comando snmpgetnext.

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.1
IF-MIB::ifIndex.1 = INTEGER: 1

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.2
IF-MIB::ifDescr.1 = STRING: lo

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.3
IF-MIB::ifType.1 = INTEGER: softwareLoopback(24)

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.4
IF-MIB::ifMtu.1 = INTEGER: 16436

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.5
IF-MIB::ifSpeed.1 = Gauge32: 10000000

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.6

IF-MIB::ifPhysAddress.1 = STRING:

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.7
IF-MIB::ifAdminStatus.1 = INTEGER: up(1)

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.8
IF-MIB::ifOperStatus.1 = INTEGER: up(1)

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.9
IF-MIB::ifLastChange.1 = Timeticks: (0) 0:00:00.00

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.10
IF-MIB::ifInOctets.1 = Counter32: 5315916

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.11
IF-MIB::ifInUcastPkts.1 = Counter32: 12875

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.12
IF-MIB::ifInNUcastPkts.1 = Counter32: 0

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.13
IF-MIB::ifInDiscards.1 = Counter32: 0
l**inux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.14**
IF-MIB::ifInErrors.1 = Counter32: 0

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.15
IF-MIB::ifInUnknownProtos.1 = Counter32: 0

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.16
IF-MIB::ifOutOctets.1 = Counter32: 5355182

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.17
IF-MIB::ifOutUcastPkts.1 = Counter32: 12920

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.18
IF-MIB::ifOutNUcastPkts.1 = Counter32: 0

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.19
IF-MIB::ifOutDiscards.1 = Counter32: 0

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.20
IF-MIB::ifOutErrors.1 = Counter32: 0

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.21
IF-MIB::ifOutQLen.1 = Gauge32: 0

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.22
IF-MIB::ifSpecific.1 = OID: SNMPv2-SMI::zeroDotZero

linux-snmp:~ # snmpgetnext -v 1 -c public 192.168.1.120 interfaces.2.1.23
RFC1213-MIB::atIfIndex.3.1.192.168.1.1 = INTEGER: 3

Ejercicio 7.

Abrir conexiones (ftp, telnet, http...) y consultar la tabla de conexiones abiertas con el comando snmpwalk: tcp.tcpConnTable.

Utilizar snmptable para consultar la tabla. Observar la diferencia.

linux-snmp:~ # snmpwalk -v 1 -c prueba localhost tcp.tcpConnTable
TCP-MIB::tcpConnState.0.0.0.0.111.0.0.0.0.0 = INTEGER: listen(2)
TCP-MIB::tcpConnState.0.0.0.0.1984.0.0.0.0.0 = INTEGER: listen(2)

TCP-MIB::tcpConnState.0.0.0.0.5666.0.0.0.0.0 = INTEGER: listen(2)
TCP-MIB::tcpConnState.127.0.0.1.25.0.0.0.0.0 = INTEGER: listen(2)
TCP-MIB::tcpConnState.127.0.0.1.199.0.0.0.0.0 = INTEGER: listen(2)
TCP-MIB::tcpConnState.127.0.0.1.631.0.0.0.0.0 = INTEGER: listen(2)
TCP-MIB::tcpConnState.127.0.0.1.2544.0.0.0.0.0 = INTEGER: listen(2)
TCP-MIB::tcpConnState.192.168.151.190.45698.209.85.135.104.80 = INTEGER: established(5)
TCP-MIB::tcpConnLocalAddress.0.0.0.0.111.0.0.0.0.0 = IpAddress: 0.0.0.0
TCP-MIB::tcpConnLocalAddress.0.0.0.0.1984.0.0.0.0.0 = IpAddress: 0.0.0.0
TCP-MIB::tcpConnLocalAddress.0.0.0.0.5666.0.0.0.0.0 = IpAddress: 0.0.0.0
TCP-MIB::tcpConnLocalAddress.127.0.0.1.25.0.0.0.0.0 = IpAddress: 127.0.0.1
TCP-MIB::tcpConnLocalAddress.127.0.0.1.199.0.0.0.0.0 = IpAddress: 127.0.0.1
TCP-MIB::tcpConnLocalAddress.127.0.0.1.631.0.0.0.0.0 = IpAddress: 127.0.0.1
TCP-MIB::tcpConnLocalAddress.127.0.0.1.2544.0.0.0.0.0 = IpAddress: 127.0.0.1
TCP-MIB::tcpConnLocalAddress.192.168.151.190.45698.209.85.135.104.80 = IpAddress: 192.168.151.190
TCP-MIB::tcpConnLocalPort.0.0.0.0.111.0.0.0.0.0 = INTEGER: 111
TCP-MIB::tcpConnLocalPort.0.0.0.0.1984.0.0.0.0.0 = INTEGER: 1984
TCP-MIB::tcpConnLocalPort.0.0.0.0.5666.0.0.0.0.0 = INTEGER: 5666
TCP-MIB::tcpConnLocalPort.127.0.0.1.25.0.0.0.0.0 = INTEGER: 25
TCP-MIB::tcpConnLocalPort.127.0.0.1.199.0.0.0.0.0 = INTEGER: 199
TCP-MIB::tcpConnLocalPort.127.0.0.1.631.0.0.0.0.0 = INTEGER: 631
TCP-MIB::tcpConnLocalPort.127.0.0.1.2544.0.0.0.0.0 = INTEGER: 2544
TCP-MIB::tcpConnLocalPort.192.168.151.190.45698.209.85.135.104.80=INTEGER: 45698
TCP-MIB::tcpConnRemAddress.0.0.0.0.111.0.0.0.0.0 = IpAddress: 0.0.0.0
TCP-MIB::tcpConnRemAddress.0.0.0.0.1984.0.0.0.0.0 = IpAddress: 0.0.0.0
TCP-MIB::tcpConnRemAddress.0.0.0.0.5666.0.0.0.0.0 = IpAddress: 0.0.0.0
TCP-MIB::tcpConnRemAddress.127.0.0.1.25.0.0.0.0.0 = IpAddress: 0.0.0.0
TCP-MIB::tcpConnRemAddress.127.0.0.1.199.0.0.0.0.0 = IpAddress: 0.0.0.0

TCP-MIB::tcpConnRemAddress.127.0.0.1.631.0.0.0.0.0 = IpAddress: 0.0.0.0
TCP-MIB::tcpConnRemAddress.127.0.0.1.2544.0.0.0.0.0 = IpAddress: 0.0.0.0
TCP-MIB::tcpConnRemAddress.192.168.151.190.45698.209.85.135.104.80 = IpAddress: 209.85.135.104
TCP-MIB::tcpConnRemPort.0.0.0.0.111.0.0.0.0.0 = INTEGER: 0
TCP-MIB::tcpConnRemPort.0.0.0.0.1984.0.0.0.0.0 = INTEGER: 0
TCP-MIB::tcpConnRemPort.0.0.0.0.5666.0.0.0.0.0 = INTEGER: 0
TCP-MIB::tcpConnRemPort.127.0.0.1.25.0.0.0.0.0 = INTEGER: 0
TCP-MIB::tcpConnRemPort.127.0.0.1.199.0.0.0.0.0 = INTEGER: 0
TCP-MIB::tcpConnRemPort.127.0.0.1.631.0.0.0.0.0 = INTEGER: 0
TCP-MIB::tcpConnRemPort.127.0.0.1.2544.0.0.0.0.0 = INTEGER: 0
TCP-MIB::tcpConnRemPort.192.168.151.190.45698.209.85.135.104.80 = INTEGER: 80

linux-snmp:~ # snmptable -v 2c -Cw 80 -c prueba localhost tcp.tcpConnTable
SNMP table: TCP-MIB::tcpConnTable

tcpConnState	tcpConnLocalAddress	tcpConnLocalPort	tcpConnRemAddress
listen	0.0.0.0	111	0.0.0.0
listen	0.0.0.0	1984	0.0.0.0
listen	0.0.0.0	5666	0.0.0.0
listen	127.0.0.1	25	0.0.0.0
listen	127.0.0.1	199	0.0.0.0
listen	127.0.0.1	631	0.0.0.0
listen	127.0.0.1	2544	0.0.0.0
timeWait	127.0.0.2	60910	127.0.0.2
established	192.168.151.190	45698	209.85.135.104

SNMP table TCP-MIB::tcpConnTable, part 2

tcpConnRemPort

0
0
0
0
0
0
0
1984
80

Observación: La orden snmptable genera los datos en formato de filas y columnas en cambio snmpwalk no.

2.3 Desarrollo de un programa codificador/decodificador de mensajes ASN:1 en lenguaje C.

Para la realización del programa y para mejor comprensión de éste se deja documentado la declaración del formato del mensaje SNMP.

El formato está declarado en ASN.1 (RFC1157).

El mensaje se codifica con BER antes de enviarlo.

```
RFC1157-SNMP DEFINITIONS ::= BEGIN
      IMPORTS
       ObjectName, ObjectSyntax, NetworkAddress, IpAddress, TimeTicks
      FROM RFC1155-SMI;
      Message ::= SEQUENCE {
        version INTEGER {version-1(0)}, -- Version-1 for this RFC
        community OCTET STRING,          -- Community name
        data ANY -- E.g., PDUs if trivial authentication is being used
        }
PDUs ::= CHOICE {
        get-request GetRequest-PDU,
        get-next-request GetNextRequest-PDU,
        get-response GetResponse-PDU,
        set-request SetRequest-PDU,
        trap Trap-PDU
        }
    GetRequest-PDU ::= [0] IMPLICIT PDU
    GetNextRequest-PDU ::= [1] IMPLICIT PDU
    GetResponse-PDU ::= [2] IMPLICIT PDU
    SetRequest-PDU ::= [3] IMPLICIT PDU
PDU ::= SEQUENCE {
```

```
request-id INTEGER,
error-status INTEGER {noError(0), tooBig(1), noSuchName(2),
        badValue(3), readOnly(4), genErr(5)},
error-index INTEGER,        -- Sometimes ignored
variable-bindings VarBindList -- Values are sometimes ignored
}
```

Código en C del Decodificador.

Nota: La secuencia correspondiente al mensaje **SNMP**, que se codificará deberá estar en un fichero **.txt.**

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <math.h>   //para las funciones matemáticas debes utilizar la opción -lm al final del gcc -g ...
#include <fcntl.h>
//-------------------------------------------------------------------------------------------------
//Estructura que guarda los OID's
typedef struct Identificador
{
        int name[15];
        char value[15];
}VarBind;

//Estructura que guarda los datos decodificados
typedef struct snmp_pdu
{
        char comunidad[20];
        char tipo_pdu[20];
        long request_id;
        VarBind list;
}snmp;
```

```
snmp pdu;  //Variable que contendra los datos
int tam=0; //Variable que contendra la longitud del OID
//-------------------------------------------------------------------------------------------------------
//Clase UNIVERSAL -> Etiquetas
const char *construido[]={"UNIVERSAL SECUENCE","UNIVERSAL SET"};

const char *primitivo[]={"UNIVERSAL BOOLEAN","UNIVERSAL INTEGER","UNIVERSAL BIT
STRING","UNIVERSAL OCTECT STRING", "UNIVERSAL NULL","UNIVERSAL OBJECTs
IDENTIFIER","UNIVERSAL OBJECTDESCRIPTOR","UNIVERSAL EXTERNAL","UNIVERSAL
REAL","UNIVERSAL ENUMERATED","UNIVERSAL NUMERICSTRING", "UNIVERSAL
PRINTABLESTRING","UNIVERSAL TELETEXSTRING","UNIVERSAL VIDEOSTRING", "UNIVERSAL
IA5STRING","UNIVERSAL UTCTIME","UNIVERSAL GENERALIZEDTIME","UNIVERSAL GRAPHICSTRING",
"UNIVERSAL VISIBLESTRING","UNIVERSAL GENERALSTRING"};

//Valor en Hexadecimal
const char *construidoVal[]={"30","31"};
const char
*primitivoVal[]={"01","02","03","04","05","06","07","08","09","0A","12","13","14","15","16","17","18","19","1A","1B"
};
//-------------------------------------------------------------------------------------------------------
//Clase APPLICATION -> Etiquetas
char *Aprimitivo[]={"APPLICATION 0","APPLICATION 1","APPLICATION 2","APPLICATION 3"};
char *Aconstruido[]={"APPLICATION 1"};
char *AprimitivoVal[]={"40","41","42","43"};
char *AconstruidoVal[]={"61"};
//-------------------------------------------------------------------------------------------------------
char buffer2[2];  //buffer de datos
int fd; //descriptor de archivo
char buffer[2];  //buffer de lectura de datos
//-------------------------------------------------------------------------------------------------------
//Funcion para limpiar la variable lectora del fichero
void limpiar()
{
        int j;
        for(j=0;j<2;j++)
                buffer[j]='\0';
```

```
        return;
}

//Funcion para limpiar el buffer de entrada
void limpiar_buffer()
{
        int j;
        for(j=0;j<2;j++)
        buffer2[j]='\0';
        return;
}

//Calcular la longitud de cada tipo
int longitud(char *buf)
{
        int A=10,B=11,C=12,D=13,E=14,F=15; //Simular los valores en Hexadecimal
        int i=0;
        int temp=0, resu=0;

        for(i=0;i<2;i++){
        temp=buf[i]-48;  //temp guarda el valor en decimal del caracter

        if(buf[i]=='A') temp=A;
        else if(buf[i]=='B') temp=B;
                else if(buf[i]=='C') temp=C;
                        else if(buf[i]=='D') temp=D;
                                else if(buf[i]=='E') temp=E;
                                        else if(buf[i]=='F') temp=F;

        resu+=temp;

        if(resu>0 && i==0)  //si es el primer octeto multiplicarlo por 16
        resu*=16;
        }
```

```
        return(resu);  //retornar el valor de la longitud
}
//Funcion que pasa de Hexadecimal a binario y calcula si el bit de mayor de peso del octeto es 1 o 0
int HexaBin(char *buf)
{
        int i=0,j=0,value=0;
        int num[]={0,0,0,0};

        value=buf[i]-48; //value guarda el valor decimal del caracter

        //mientras el valor del caracter sea letras o numeros(excepto 0)
        while(value>0)
        {
                num[j]=value%2;
                value=value/2;
                j++;
        }
        if(num[3]==1) return 1;
        else return 0;
}
//Función para decodificar el fichero: dirección de una zona de memoria en la que se apuntara al mensaje
//codificado en BER
void decodificador()
{
        //Declaracion de variables
        char c;  //indicara el tipo de mensaje
        //tamanyo de cada arreglo de etiqueta (construido y primitivo)
        int tamC=2,tamP=20;
        //obtener el valor del tipo de dato
        int octect=0,oid=0,integer=0;
        //variables para capturar el request-id y la comunidad
        long value2=0;
        int p_oct=0,p_in=0;
        //variables para el object identifier
```

```
int k=-1,poid=0,aux=1;
int num[]={0,0,0,0,0,0};
int leer=1;
//variables extras
int buf[20],bin=0,input=0;
int i,value=0,leng,j;
int n,cont=0,ant=0;
limpiar();   //limpiar buffer de lectura
while((input=read(fd,buffer,1))>0)   //Leer si no ha llegado al final de fichero
{
        i=0, value=0;
        limpiar_buffer(buffer2);  //limpiar buffer de datos
        if(isxdigit(buffer[0]))   //Si es un valor hexadecimal
        {
                buffer2[i]=buffer[0];  //guardar caracter leido

                //Si es una clase Especifica de Contexto del tipo construido
                if(buffer[0]=='A')
                {
                        i++;
                        limpiar();
                        input=read(fd,buffer,1);
                        if(isxdigit(buffer[0]))  //leer el siguiente caracter
                buffer2[i]=buffer[0];
                        printf("\n%s ",buffer2);  //mostrar el octeto leido
                        c=buffer[0];  //capturar el tipo de mensaje
                        limpiar_buffer();
                        limpiar();
                        input=read(fd,buffer,1);
                        if(isspace(buffer[0]))  //Despues de dos octetos viene un espacio en blaco
                                i=0;
                        while(i<2){
                                limpiar();
                                input=read(fd,buffer,1);
```

```
                    if(isxdigit(buffer[0]))
                            buffer2[i++]=buffer[0];
                    }
                    printf("%s",buffer2);
                    leng=longitud(buffer2); //Valor de la longitud del tipo
                    printf("\t--[%c], construido, ",c);
                    //Tipo de Mensaje
                    if(c=='0'){ printf("get-Request");
bcopy("get-request",pdu.tipo_pdu,strlen("get-request"));}
                    if(c=='1'){printf("get-nextrequest");
bcopy("get-nextrequest",pdu.tipo_pdu,strlen("get-nextrequest"));}
                    if(c=='2'){printf("get-response");
bcopy("get-response",pdu.tipo_pdu,strlen("get-response"));}
                    if(c=='3'){printf("get-setRequest");
bcopy("get-setrequest",pdu.tipo_pdu,strlen("get-setrequest"));}
                    printf(",longitud %d",leng);  //longitud del tipo
                    limpiar_buffer(buffer2);
                    continue;
            }
            //Si es una clase Especifica de Contexto del tipo primitivo
            if(buffer[0]=='8')
            {
                    i++;
                    limpiar();
                    input=read(fd,buffer,1);  //leer el siguiente caracter
                    if(isxdigit(buffer[0]))
                          buffer2[i]=buffer[0];
                    printf("\n%s ",buffer2);  //mostrar el octeto leido
                    c=buffer[0];
                    limpiar_buffer();
                    limpiar();
                    input=read(fd,buffer,1); //leer el siguiente caracter
                    if(isspace(buffer[0]))  //Despues de dos octetos viene un espacio en blaco
                            i=0;
```

```
		while(i<2){
			limpiar();
		input=read(fd,buffer,1);  //leer el siguiente octeto
		if(isxdigit(buffer[0]))
				buffer2[i++]=buffer[0];
		}
		printf("%s",buffer2);  //mostrar el octeto leido
		leng=longitud(buffer2); //Valor de la longitud del tipo
		printf("\t--[%c], primitivo, ",c);
		printf(",longitud %d",leng);  //longitud del tipo
		limpiar_buffer();
		continue;
}
//Si es una clase Aplicacion del tipo primitivo
if(buffer[0]=='4')
{
		i++;
		k=-1;
		limpiar();
		input=read(fd,buffer,1);  //leer el siguiente caracter
		if(isxdigit(buffer[0]))
			buffer2[i]=buffer[0];
		printf("\n%s ",buffer2);  //mostrar el octeto leido
		if(strcmp(buffer2,AprimitivoVal[0])==0)  //Es un IpAddress
			k=0;
		if(strcmp(buffer2,AprimitivoVal[1])==0)   //Es un Counter
			k=1;
		if(strcmp(buffer2,AprimitivoVal[2])==0)  //Es un Gauge
			k=2;
		if(strcmp(buffer2,AprimitivoVal[3])==0)   //Es un TimeTicks
			k=3;
		limpiar_buffer();
		limpiar();
		input=read(fd,buffer,1);  //leer el siguiente caracter y
```

```
if(isspace(buffer[0]))    //ver si es un espacio en blanco
        i=0;
while(i<2){
        limpiar();
    input=read(fd,buffer,1);  //leer el siguiente octeto
    if(isxdigit(buffer[0]))
                buffer2[i++]=buffer[0];
}
printf("%s",buffer2);  //mostrar el octeto leido
leng=longitud(buffer2); //Valor de la longitud del tipo
if(k>=0)
        printf("\t--[%s],primitivo,longitud %d",Aprimitivo[k],leng);
j=0;
n=leng*2-2;  //calcula la potencia correspondiente al octeto leido
value2=0;
value=0;
printf("\n");
limpiar_buffer(buffer2);
while(j<leng){  //mientras no alcance la longitud
        limpiar();
        input=read(fd,buffer,1);  //leer el siguiente caracter y
        if(isspace(buffer[0]))    //ver si es un espacio en blanco
                i=0;
        while(i<2){
        limpiar();
                input=read(fd,buffer,1);   //leer el siguiente octeto
                if(isxdigit(buffer[0]))
                        buffer2[i++]=buffer[0];
        }
        printf("%s ",buffer2);    //mostrar el octeto leido
        if(n>1)  //si la potencia es mayor a 1
        {
                value=longitud(buffer2);       //calcula el valor en decimal
                value2+=(long)(value*pow(16,n));  //del octeto leido
```

```
                                        n-=2;
                                }
                                else  //si no
                                        value2+=(long)longitud(buffer2);
                                j++;
                        }
                        //Mostrar el tipo de dato
                        if(k==0)  printf("--IpAddress %ld",value2);
                        if(k==1)  printf("--Counter %ld",value2);
                        if(k==2)  printf("--Gauge %ld",value2);
                        if(k==3)  printf("--TimeTicks %ld",value2);
                        limpiar_buffer();
                        continue;
                }
                i++;
                limpiar();
                input=read(fd,buffer,1);  //leer el siguiente caracter
                if(isxdigit(buffer[0]))   //si es un digito hexadecimal
    buffer2[i]=buffer[0];
printf("\n%s",buffer2);  //mostrar el octeto leido
                //Si es una clase UNIVERSAL del tipo construido
                for(i=0;i<tamC;i++)
                {
                        //si pertenece a los tipos construidos
                        if(strcmp(buffer2,construidoVal[i])==0){
                                //Longitud del tipo
                                limpiar();
                                input=read(fd,buffer,1);  //leer el siguiente caracter
                if(isspace(buffer[0]))   //si es un espacio en blanco
                                        j=0;
                                while(j<2){
                                        limpiar();
                                        input=read(fd,buffer,1); //leer el siguiente octeto
                                        if(isxdigit(buffer[0]))
```

```
                                        buffer2[j++]=buffer[0];
                                }
                                leng=longitud(buffer2); //Valor de la longitud del tipo
                                printf(" %s",buffer2);   //mostrar el octeto leido
                                printf("\t--[%s], ",construido[i]);
                                printf(" construido");
                                printf(",longitud %d",leng);
                                limpiar_buffer();
                        }
                }
        //Si es una clase UNIVERSAL del tipo Primitivo
        for(i=0;i<tamP;i++)
                {
                        //si pertenece a los tipos primitivos
                        if(strcmp(buffer2,primitivoVal[i])==0){
                                if(strcmp(primitivoVal[i],"06")==0) oid=1;   //si es un tipo OID's
                                //Si es un tipo INTEGER
                                if(strcmp(primitivoVal[i],"02")==0) {integer=1; p_in++;}
                                //si es un tipo OCTECT STRING y derivados
                                if((strcmp(primitivoVal[i],"04")==0) ||
(strcmp(primitivoVal[i],"13")==0) ||(strcmp(primitivoVal[i],"12")==0) || (strcmp(primitivoVal[i],"16")==0))
                                        {octect=1; p_oct++;}
                                //Longitud
                                limpiar();
                                input=read(fd,buffer,1);  //leer el siguiente caracter y
                        if(isspace(buffer[0]))   //ver si es un espacio en blanco
                                        j=0;
                                while(j<2){
                                        limpiar();
                                        input=read(fd,buffer,1);   //leer el siguiente octeto
                                        if(isxdigit(buffer[0]))
                                                buffer2[j++]=buffer[0];
                                }
                                leng=longitud(buffer2);  //Valor de la longitud del tipo
```

```
                printf(" %s",buffer2);    //mostrar el octeto leido
                if((strcmp(primitivoVal[i],"05")==0)  &&  (strcmp(pdu.tipo_pdu,"get-
setrequest")==0) || (strcmp(pdu.tipo_pdu,"get-response")==0) && ant==1)
                        ant=0;
                else{
                printf("\t--[%s], ",primitivo[i]);
                printf(" primitivo");
                printf(",longitud %d",leng);
                limpiar_buffer();
                }
                        //oid=0;
                //Valor del Tipo


                if(integer){
                        j=0;
                        n=leng*2-2;//calcula la potencia correspondiente al octeto
                        //leido
                        k=0;
                        value2=0;
                        value=0;
                        printf("\n");
                        limpiar_buffer(buffer2);
                        while(j<leng){  //mientras no alcance la longitud
                                limpiar();
                                input=read(fd,buffer,1);//leer el siguiente
                                //caracter y
                        if(isspace(buffer[0]))//ver si es un espacio en blanco
                                i=0;
                        while(i<2){
                                        limpiar();
                                        input=read(fd,buffer,1);//leer el
                                //siguiente octeto
                                if(isxdigit(buffer[0])){
                                                buffer2[i++]=buffer[0];
                                        }
```

```
                }
                printf("%s ",buffer2);    //mostrar el octeto leido
                if(n>1)  //si la potencia es mayor a 1
                {
                        value=longitud(buffer2);//calcula el
                        //valor en decimal
                        value2+=(long)(value*pow(16,n));//del
                        //octeto leido
                        n-=2;
                }
                else  //si no
                        value2+=(long)longitud(buffer2);
                j++;
        }
        if(p_in==1 || p_in>4) printf("\t--valor %d",value2); //otros
        //si es el primer INTEGER
        if(p_in==1 && value2==0) printf(" (version SNMP 1)");
        if(p_in==2){   //si es el segundo ..
                pdu.request_id=value2;
                printf("--request-id %ld",value2);
        }
        if(p_in==3 && (strcmp(pdu.tipo_pdu,"get-
response")==0)){ //si es el tercero
                //Error-Status
                if(value2==0) printf("\t--error-status=noError");
                if(value2==1) printf("\t--error-status=tooBig");
                if(value2==2) printf("\t--error-
status=noSuchName");
                if(value2==3) printf("\t--error-status=badValue");
                if(value2==4) printf("\t--error-status=readOnly");
                if(value2==5) printf("\t--error-status=genErr");
        }
        else if(p_in==3 && value2==0) //para los demas PDU's
                        printf("\t--error-status=noError");
        if(p_in==4 && value2==0)
```

```
                        //Error-Index debe valer 0
                         printf("\t--error-index=%ld",value2); // cuarto?
                limpiar_buffer();
                integer=0;
                break;
        }

        //Si es un OCTECT STRING y derivados
        if(octect){
                j=0;
                printf("\n");
                while(j<leng){
                        limpiar();
                        input=read(fd,buffer,1);//leer siguiente caracter
                        if(isspace(buffer[0]))//espacio en blanco?
                                i=0;
                        limpiar_buffer();
                        while(i<2){
                                limpiar();
                                input=read(fd,buffer,1); //leer sigte  oct
                                if(isxdigit(buffer[0]))
                                        buffer2[i++]=buffer[0];
                        }
                        printf("%s ",buffer2);  //mostrar el octeto leido
                        buf[j]=longitud(buffer2);
                //si es el primer OCTECT STRING: es la comunidad
                        if(p_oct==1) pdu.comunidad[j]=buf[j];
                        j++;
                }
                printf("\t--valor ");
                for(j=0;j<leng;j++)//mostrar el valor correspondiente
                        printf("%c",buf[j]);
                octect=0;
                break;
```

```
}
//Si es un OBJECT's IDENTIFIER
if(oid){
        j=0;
        printf("\n");
        n=(2*leng)-2;
        while(j<leng+1){
                limpiar();
                input=read(fd,buffer,1);//leer siguiente caracter y
                if(isspace(buffer[0]))// espacio en blanco?
                        i=0;
                limpiar_buffer();
                while(i<2){
                        limpiar();
                        input=read(fd,buffer,1);//leer sigte oct
                        if(isxdigit(buffer[0]))
                                buffer2[i++]=buffer[0];
                }
                printf("%s ",buffer2);  //mostrar el octeto leido
                if((bin=HexaBin(buffer2))) //Si el bit de mayor
                //peso del octetot es 1
                {
                        k=j;
                        j=0;
                        num[j++]=longitud(buffer2); //guardar el
                        //valor del octeto
                        while(k<leng)
                        {
                        limpiar();
                        //siguiente caracter
                        input=read(fd,buffer,1);
                        if(isspace(buffer[0]))//espacio en blanco
                                i=0;
                                limpiar_buffer();
                                while(i<2){
```

```
                    limpiar();
                    //leer el siguiente octeto
                    input=read(fd,buffer,1);
                    if(isxdigit(buffer[0]))
                    buffer2[i++]=buffer[0];
                }
                poid++; //viene otro octeto
                printf("%s ",buffer2); //ver octeto leido
                if(poid==1){
        //almacena la potencia para el octeto leido
                    aux++;
                    num[j++]=longitud(buffer2);}
                else
                //guardar el valor del octeto
                buf[k]=longitud(buffer2);
                k++;
                //si viene otro octeto
                if((bin=HexaBin(buffer2))){
                    poid=0;
                    k--;   //decrementar el contador
                    leer=1;
                    leng--;//decrementar la long
                    }
                else
                    leer=0;
    //si ya no vienen mas octetos calcular el valor real
                if(!leer && aux>1){
                    k--;
                    j=0;
                    //calcular la potencia real
                    aux=(aux*2)-2;
                    buf[k]=0;
                    while(aux>0) //lpotencia > a 0
                    {//adicionar el valor de c/octeto
                buf[k]+=(int)(num[j]*pow(16,aux));
                    aux-=2;
```

```
                                        j++;
                                        }
                                buf[k]+=num[j];//adicionar el ultimo oct
                                leer=1;
                                k++;  //incrementar el contador
                                j=0;
                                aux=1;  //reiniciar la potencia
                                }
                        }
                        printf(" --");
                        for(i=0,j=0;j<leng;j++,i++)
                        {pdu.list.name[j]=buf[j]; //guardar el valor del OID
                        printf(".%d",buf[j]);  //mostrar la ruta del OID}
                        tam=leng;   //tamanyo del arreglo buf
                        oid=0;
                        break;
                }
                buf[j]=longitud(buffer2);  //almacenar el valor del octeto
                if(j==0){   //si es el primer octeto
                        buf[j++]=(buf[j]-3)/40;//calc los dos prim indices
                        buf[j]=3;}
                j++;
        }  //Fin del primer while
        if(oid==1){ //si no hay octetos que indican la adicion de otro
                printf(" --");
                for(j=0;j<leng+1;j++){
                        pdu.list.name[j]=buf[j];  //guardar el valor del OID
                        printf(".%d",buf[j]);  //mostrar la ruta del OID
                        }
                tam=leng+1; //tam del arreglo buf para mostrar los datos
        }
        oid=0;
        ant=1;
        break;
```

```
                                        }
                                }
                        }
                }//Fin del if

                else if(isspace(buffer[0])) continue;   //si el caracter es un espacio en blanco

                } //Fin del while
} //Fin de la funcion decodificador()

//-------------------------------------------------------------------------------------------------
int main()
{
        char nombre[20];
        int i;

        printf("SI ALGUNOS DE LOS CAMPOS NO ESTA DECODIFICADO, SE ENTIENDE QUE LA
        CODIFICION ESTA MAL..!!!\n");
        printf("NOMBRE DEL FICHERO FUENTE:");
        scanf("%s",&nombre);
        fd=open(nombre,O_RDONLY);  //abrir fichero solo para lectura
        /* Comprobación de errores */
        if (fd==-1){
        perror("Error al abrir fichero:");
        exit(1);}
        //Decodificar el contenido de fichero
        printf("\nDECODIFICANDO FICHERO.....\n");
        //Llamada a la funcion Decodificador
        decodificador();
        //Contenido Decodificado
        printf("\n\nFICHERO DECODIFICADO!!!\n");
        /*Mostrar resultados obtenidos*/
        printf("\nDATOS RECOPILADOS:\n");
        printf("\tComunidad:%s\n",pdu.comunidad);
```

```
        printf("\tTipo de PDU:%s\n",pdu.tipo_pdu);
        printf("\tNumero Identifacador de la Peticion:%ld\n",pdu.request_id);
        printf("\tOID's solicitado:");
        for(i=0;i<tam;i++) printf(".%d",pdu.list.name[i]);
        //Cerrar fichero
        close(fd);
        printf("\n");
        return 0;
}
```

3. Instalación y Configuración de Herramientas de Gestión de Redes.

Las herramientas de monitorización que recogen información de la red y de equipos que en ella habitan son hoy en día casi imprescindibles y su importancia crece de forma exponencial con el número de hosts que tenga que mantener. Algunas de ellas permiten, además, hacer pequeñas (o grandes) actuaciones sobre las fuentes de la información que recogen.

Las herramientas de gestión que vienen siendo utilizadas son principalmente los analizadores de protocolos y las plataformas de gestión SNMP.

Las funciones de la monitorización de red se llevan a cabo por agentes que realizan el seguimiento y registro de la actividad de red, la detección de eventos y la comunicación de alertas al personal responsable del buen funcionamiento de la red.

Los eventos típicos que son monitorizados suelen ser:

- Ejecución de tareas como pueden ser realización de copias de seguridad.
- Registro del estado de finalización de los procesos que se ejecutan en la red.
- Registro de los cambios que se producen en el inventario de hardware.
- Registro de las entradas y salidas de los usuarios en la red.
- Registro del arranque de determinadas aplicaciones.
- Errores en el arranque de las aplicaciones.
- Número de usuarios en el sistema, porcentaje de usuarios no atendidos y tiempos de respuesta.
- Cuellos de botella en el sistema.
- Fallos de sistema o de mala configuración.
- Registro de los intentos fallidos de acceso al sistema y de apertura de puertos.

En función de la prioridad que tengan asignados los eventos y de la necesidad de intervención se pueden utilizar diferentes métodos de notificación como son:

- Mensajes en la consola: se suelen codificar con colores en función de su importancia.
- Mensajes por correo electrónico: conteniendo el nivel de prioridad y el nombre e información del evento.
- Mensajes a móviles: cuando el evento necesita intervención inmediata se suele comunicar a los técnicos de guardia a través de este método.

3.1 MRTG (Multi Router Traffic Graphics).

3.1.1 Descripción.

En ciertas ocasiones, puede ser interesante monitorizar gráficamente ciertos estados de nuestro servidor, como la transferencia de datos a través de una interface de red, el consumo de CPU o de memoria RAM, además de muchas otras cosas. Todo esto es posible hacerlo gracias a una herramienta gráfica que permite la monitorización de las conexiones de red mediante la generación de gráficas que reflejan el uso del ancho de banda. Se comunica con los distintos dispositivos mediante el protocolo SNMP y suele utilizarse en la monitorización de equipos de red, principalmente en routers y switches.

MRTG es una herramienta gratuita, estable y que funciona prácticamente con cualquier sistema Unix.
A diferencia de otras herramientas de monitorización como el IPaudit, MRTG simplemente genera una gráfica diaria, semanal, mensual y anual con el total del ancho de banda utilizado en la conexión a Internet. De esta forma, es el complemento perfecto para el IPaudit, ya que nos permite obtener un control exhaustivo de los segmentos de la red.

3.1.2 Instalación y Configuración.

Instalar desde la consola del SUSE:

#rpm -ihv mrtg-2.12.2-2.i586.rpm

El siguiente paso es configurar el mrtg para monitorizar los dispositivos de red:

Asegurarse que el net-snmp este instalado, configurado y corriendo.

Crear el directorio para el mrtg.

#mkdir /srv/www/htdocs/mrtg

Crear el archivo de configuración y ejecutarlo:

#pico /etc/mrtg/makecfg.sh

```
#!/bin/bash
# host list HOSTLIST=" 192.168.1.104 192.168.1.110 "
# Make configuration file
for HOST in $HOSTLIST
do
        /usr/bin/cfgmaker \
                        --ifref=eth \
                         --global 'WorkDir: /srv/www/htdocs/mrtg' \
                        --global 'Options[_]: bits,growright' \
                        --output /etc/mrtg/$HOST.cfg \
                        prueba@$HOST
                        done
```

Éste ejemplo monitoriza dos dispositivos (192.168.1.104 y 192.168.1.110) y crea dos archivos de configuración 192.168.1.104.cfg y 192.168.1.110.cfg.

Crear un archivo ejecutable y ejecutarlo:

#pico /etc/mrtg/mrtg.cron

```
#!/bin/bash

DIRCFG=/etc/mrtg
DIRHTML=/srv/www/htdocs/mrtg
LISTCFG=" 192.168.1.104.cfg
                192.168.1.110.cfg "

for CFG in $LISTCFG
do
        env LANG=C /usr/bin/mrtg $DIRCFG/$CFG
done

# Make index file
 cd $DIRCFG
/usr/bin/indexmaker --columns=1 $LISTCFG > $DIRHTML/index.html
```

Para ejecutarlo cada 5 minutos, es decir, para que los gráficos MRTG se actualicen cada 5 minutos es necesario agregar esta línea en el /etc/crontab:

```
 0-59/5 * * * * root /etc/mrtg/mrtg.cron >/dev/null 2>&1
```

En la configuración por defecto, solamente el localhost puede ver el análisis de **http://localhost/mrtg**. Sin embargo, si deseas permitir que la otra computadora la vea, corregir /etc/httpd/conf.d/mrtg.conf como abajo,

```
Order deny,allow
# Deny from all
Allow from localhost
```

Allow from .example.com

Nota: éste paso no fue necesario en el momento de la instalación pues nosotros hicimos la prueba desde el servidor y nos dio resultado.
Si se desea monitorizar la cpu de algún host(en nuestro caso 192.168.1.104):

Agregar las siguientes líneas al archivo de configuración host.cfg (192.168.1.104.cfg):

This section creates CPU Load monitoring
CPU Load Average
Target[192.168.1.104.cpu]:.1.3.6.1.4.1.2021.10.1.5.1&.1.3.6.1.4.1.2021.10.1.5.2:prueba@192.168.1.104
MaxBytes[192.168.1.104.cpu]: 100
Unscaled[192.168.1.104.cpu]: dwmy
Options[192.168.1.104.cpu]: gauge, absolute, growright, noinfo, nopercent
YLegend[192.168.1.104.cpu]: CPU Load(%)
ShortLegend[192.168.1.104.cpu]: (%)
LegendI[192.168.1.104.cpu]: 1min ave
LegendO[192.168.1.104.cpu]: 5min ave
Legend1[192.168.1.104.cpu]: 1min ave(%)
Legend2[192.168.1.104.cpu]: 5min ave(%)
Title[192.168.1.104.cpu]: CPU Load ($HOST)
PageTop[192.168.1.104.cpu]: <H1>CPU Load Average – suse10</H1>

Si se desea monitorizar la memoria utilizada por algún host (en nuestro caso 192.168.1.104), agregar las siguientes líneas al archivo de configuración host.cfg (192.168.1.104.cfg):

LoadMIBs: /usr/share/snmp/mibs/HOST-RESOURCES-MIB.txt

Target[192.168.1.104.mem]:.1.3.6.1.4.1.2021.4.6.0&.1.3.6.1.4.1.2021.4.6.0:prueba@192.168.1.104
PageTop[192.168.1.104.mem]: <H1>Memoria RAM</H1>
Options[192.168.1.104.mem]: nopercent,growright,gauge,noinfo
Title[192.168.1.104.mem]: Memoria Libre
MaxBytes[192.168.1.104.mem]: 1000000
kMG[192.168.1.104.mem]: k,M,G,T,P,X
YLegend[192.168.1.104.mem]: bytes
ShortLegend[192.168.1.104.mem]: bytes
LegendI[192.168.1.104.mem]: Free Memory:
LegendO[192.168.1.104.mem]:
Legend1[192.168.1.104.mem]: Free memory, not including swap, in bytes

En principio esto es todo, aunque por supuesto, puede ampliarse muchísimo añadiendo otros gráficos.

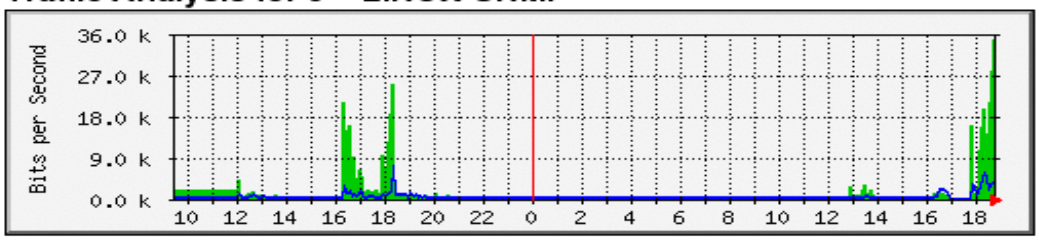
La instalación efectuada es la que se recomienda por defecto. Sin embargo se han de tener en cuenta los siguientes requisitos:
Los dispositivos monitorizados con MRTG deben tener activado el protocolo SNMP para que el software pueda conectarse a ellos y leer los datos.

Es necesario un servidor WWW y las librerías LIBPNG que permiten la creación y manipulación de imágenes en formato PNG. El sistema debe permitir la ejecución planificada (CRON) de scripts a los usuarios. Este punto es esencial ya que el programa genera gráficas cada cierto intervalo de tiempo, lo que le obliga a ejecutar determinados comandos cada 5 minutos.

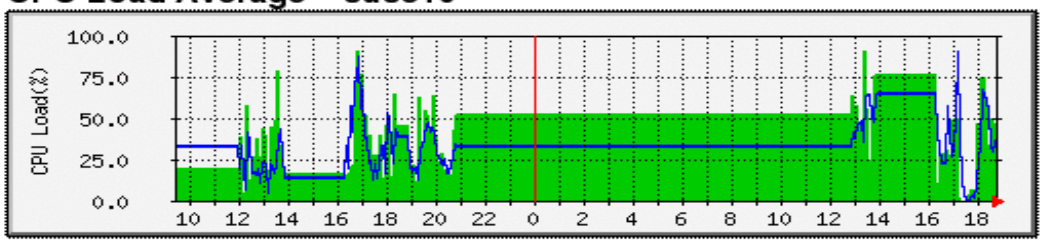
Ésta es la imagen de nuestro MRTG en funcionamiento:

MRTG Index Page

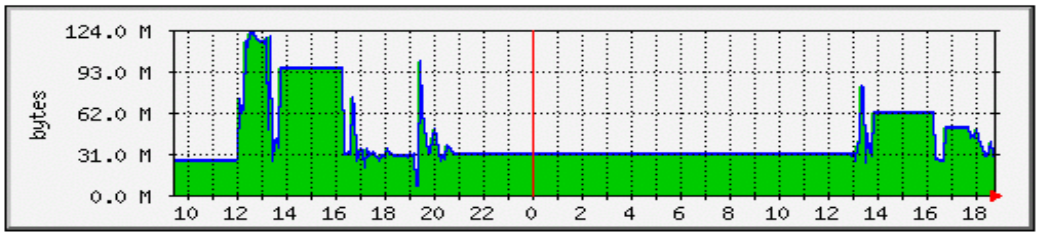
Traffic Analysis for 3 -- LINUX-SNMP



CPU Load Average -- suse10



Memoria RAM



MRTG MULTI ROUTER TRAFFIC GRAPHER
version 2.14.7
Tobias Oetiker <tobi@oetiker.ch>
and Dave Rand <dlr@bungi.com>

Fig. 25 MRTG en funcionamiento.

3.2 CACTI.

3.2.1 Descripción.

Cacti es un sistema de monitorización de redes, esto quiere decir, que podemos tener controlados los servicios que presta nuestra red en todo momento casi en tiempo real. Es un programa que representa mediante gráficos las estadísticas de la red, de fácil utilización para que administradores de sistemas relativamente inexpertos lo utilicen, mientras que al mismo tiempo es de bastante gran alcance y puede ser utilizado en redes complejas.

Cacti es una completa solución de graficado en red, diseñada para aprovechar el poder de almacenamiento y la funcionalidad de graficar que poseen las RRDtool(Round Robin Database tool). Esta herramienta, desarrollada en PHP, provee un pooler ágil, plantillas de gráficos avanzadas, múltiples métodos para la recopilación de datos, y manejo de usuarios. Tiene una interfaz de usuario fácil de usar, que resulta conveniente para instalaciones del tamaño de una LAN, así como también para redes complejas con cientos de dispositivos

RRDtool es el acrónimo de Round Robin Database tool, o sea que se trata de una herramienta que trabaja con una Base de Datos que maneja planificación según Round-Robin. Esta técnica trabaja con una cantidad fija de datos y un puntero al elemento actual. El modo en que trabaja una base de datos utilizando Round Robin es el siguiente; se trata la BD como si fuera un círculo, sobrescribiendo los datos almacenados, una vez alcanzada la capacidad de la BD. La capacidad de la BD depende de la cantidad de información como historial que se quiera conservar.

En una RRD se puede almacenar cualquier tipo de dato, siempre que se trate de una serie temporal de datos. Esto significa que se tiene que poder realizar medidas en algunos puntos de tiempo y proveer esta información a la RRDtool para que la almacene.

Un concepto ligado a las RRDtool es el de SNMP(Simple Network Management Protocol). Éste protocolo puede ser usado para realizar consultas a dispositivos acerca del valor de los contadores que ellos tienen (Ej.: una impresora). El valor obtenido de esos contadores es el que queremos guardar en la RRD.

Para manejar la recopilación de datos, se le puede pasar a Cacti la ruta a cualquier script o comando junto con cualquier dato que el usuario necesite ingresar; Cacti reunirá estos datos, introduciendo este job en el cron (para el caso de un sistema operativo Linux) y cargará los datos en la BD MySQL y los archivos de planificación según Round-Robin que deba actualizar.

Una fuente de datos también puede ser creada. Por ejemplo, si se quiere graficar los tiempos de ping de un host, se podría crear una fuente de datos, utilizando un script que haga ping a un host y devuelva el valor en milisegundos. Después de definir opciones para la RRDtool, como la forma de almacenar los datos, uno puede definir cualquier información adicional que la fuente de entrada de datos requiera, como en este caso, la IP del host al cual hacer el ping. Luego que una fuente de datos es creada, es automáticamente mantenida cada 5 minutos.

Una vez que una o más fuentes de datos son definidas, una gráfica de RRDtool puede ser creada usando los datos obtenidos. Cacti permite crear prácticamente cualquier gráfica, utilizando todos los estándares de tipos de gráficas de RRDtool y funciones de consolidación.

Cacti está diseñada para monitorear dispositivos de red y servidores; utiliza al máximo RRDTool. Ya trae incluidos todos los scripts para monitorear los dispositivos y nos da la posibilidad de indicar cuáles deseamos monitorear y qué aspecto en especial queremos estudiar.

Algunas de las cosas que se pueden monitorear con Cacti son:

- Memoria utilizada.
- Procesos.
- Utilización del disco.
- Usuarios logueados.
- Latencia del Ping.

3.2.2 Instalación y Configuración.

En Suse 10.2 los paquetes necesarios para poder instalar cacti son:

- cacti-0.8.6j.tar.gz
- MySQL-server-standard-5.0.27 0.sles9.i586.rpm.
- Apache2.
- PHP5.
- módulos de php para Apache, Net- snmp y mysql
- RRDtool
- Net-SNMP
- mysql-devel

Es necesario arrancar los siguientes procesos:

snmp---->**/etc/init.d/snmpd start**

apache2--->**/etc/init.d/apache2 start**

mysql------>**/etc/init.d/mysql start**

Crear un usuario llamado cactiuser:

#useradd cactiuser -g users

Si se accede por primera vez a Mysql, entonces ejecutar el siguiente comando, para de ésta forma entrar a la base de datos y establecer la contraseña de root:

#mysqladmin -u root password 'pasword_root'

Si ya había accedido por primera vez, ejecutar el siguiente comando, nos pedirá la contraseña de root , y entrará al pront de mysql.

#mysql -u root -p

Allí ejecutar los siguientes comandos:

mysql> create database cactidb;
mysql> grant all on cactidb.* to root;
mysql> grant all on cactidb.* to root@localhost;
mysql> grant all on cactidb.* to cactiuser;
mysql> grant all on cactidb.* to cactiuser@localhost;
mysql> set password for cactiuser@localhost=password('qw123');
mysql> flush privileges;
mysql> quit

Ahora se instalará el paquete Cacti

#tar xzvf cacti-0.8.6j.tar.gz -C /srv/www/htdocs/
#mv /srv/www/htdocs/cacti-0.8.6j /srv/www/htdocs/cacti
#cd /srv/www/htdocs/cacti/

Copiar la base de datos de cacti llamada cacti.sql en la base de datos que creamos

#mysql --user=cactiuser --password='qw123' cactidb < cacti.sql

o ejecutar: **# mysql -u cactiuser -p cactidb < cacti.sql**

Las carpetas **rra** y **log** tendrán como usuario cactiuser

#chown -R cactiuser rra/ log/

acceder a la carpeta scripts y cambiar el usuario a cactiuser

#cd scripts
#chown cactiuser:users *

Editar el archivo de configuración de cacti

 # pico /srv/www/htdocs/cacti/include/config.php
$database_defaut =“cactidb”;
$database_hostname = “localhost”;
$database_username =“cactiuser”;
$database_password =“qw123”;
$database_port = "3306";

Ahora se agregará una tarea para que se ejecute cada 5 minutos
#crontab -u cactiuser -e

y agregar la siguiente línea
***/5 * * * * cactiuser /usr/bin/php5 /srv/www/htdocs/cacti/poller.php > /dev/null 2>&1**

Acceder al siguiente enlace,

http://localhost/cacti

Seguir las indicaciones hasta mostrar una ventana así:

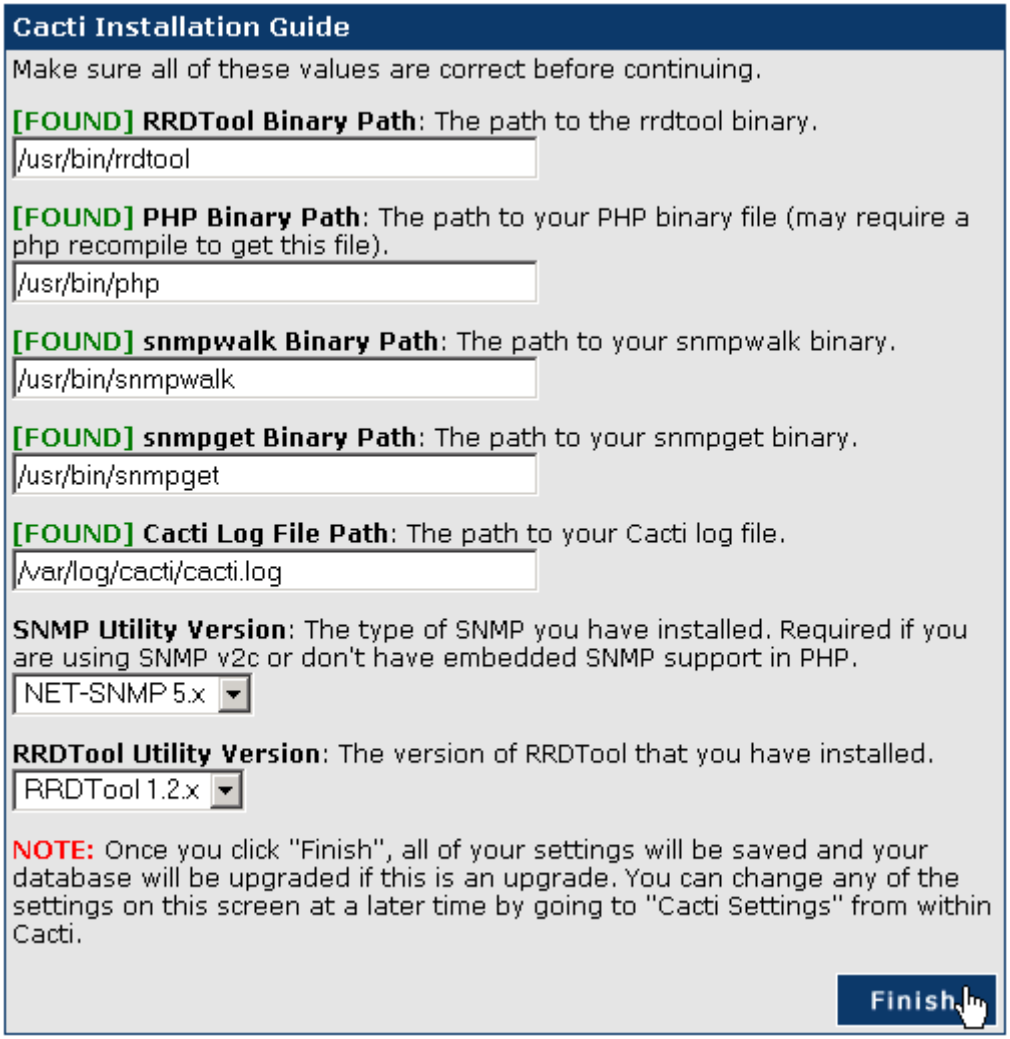


Fig. 26 Instalación del CACTI.

En este punto hay que fijarnos en que todos los apartados estén marcados en verde con la palabra [FOUND], para poder seguir la instalación. Si no hemos tenido problema nos mostrará el aspecto de la captura, todo correcto.

Ahora pedirá un usuario y un password, por defecto el usuario es **admin** y el pasword es **admin.**

Luego aparecerá la pantalla principal de cacti.

Se modificarán algunas cosas para poder ver los gráficos.

Primero click en Settings que se encuentra en el menú de la izquierda y modificar los campos, según su configuración SNMP.

SNMP Community = prueba, o la comunidad que se tenga para leer las Mib.

SNMP Port = por defecto es el 161, pero también puede cambiar.

Guardar los cambios con el botón **Save.**

Segundo dar click en Devices

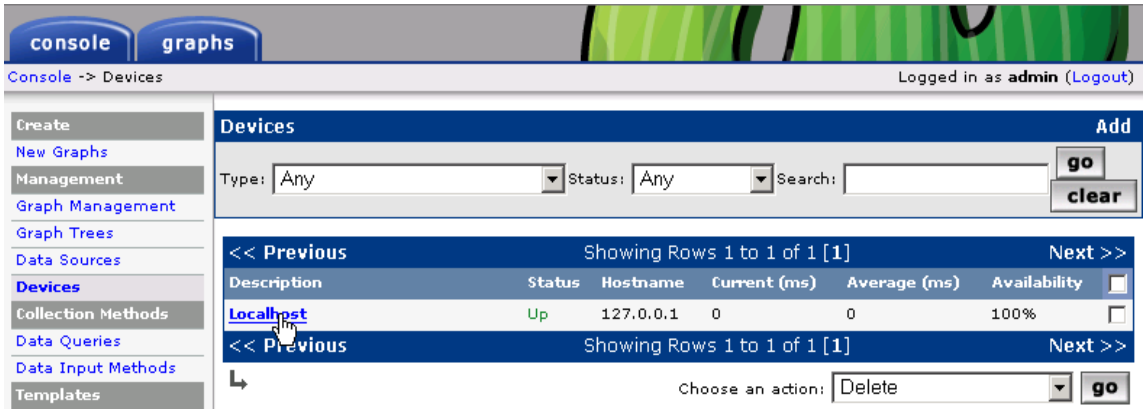


Fig. 27 CACTI en funcionamiento.

El único dispositivo que se verá será localhost, daremos click en Localhost

Allí configurar el snmp a nuestro gusto, podremos agregar la comunidad que utilizaremos para que lea las variables de las Mib y el puerto en donde tenemos corriendo el agente.

SNMP Community = prueba, o la comunidad que se tenga para leer las Mib.

SNMP Port =por defecto es el 161, pero también puede cambiar.

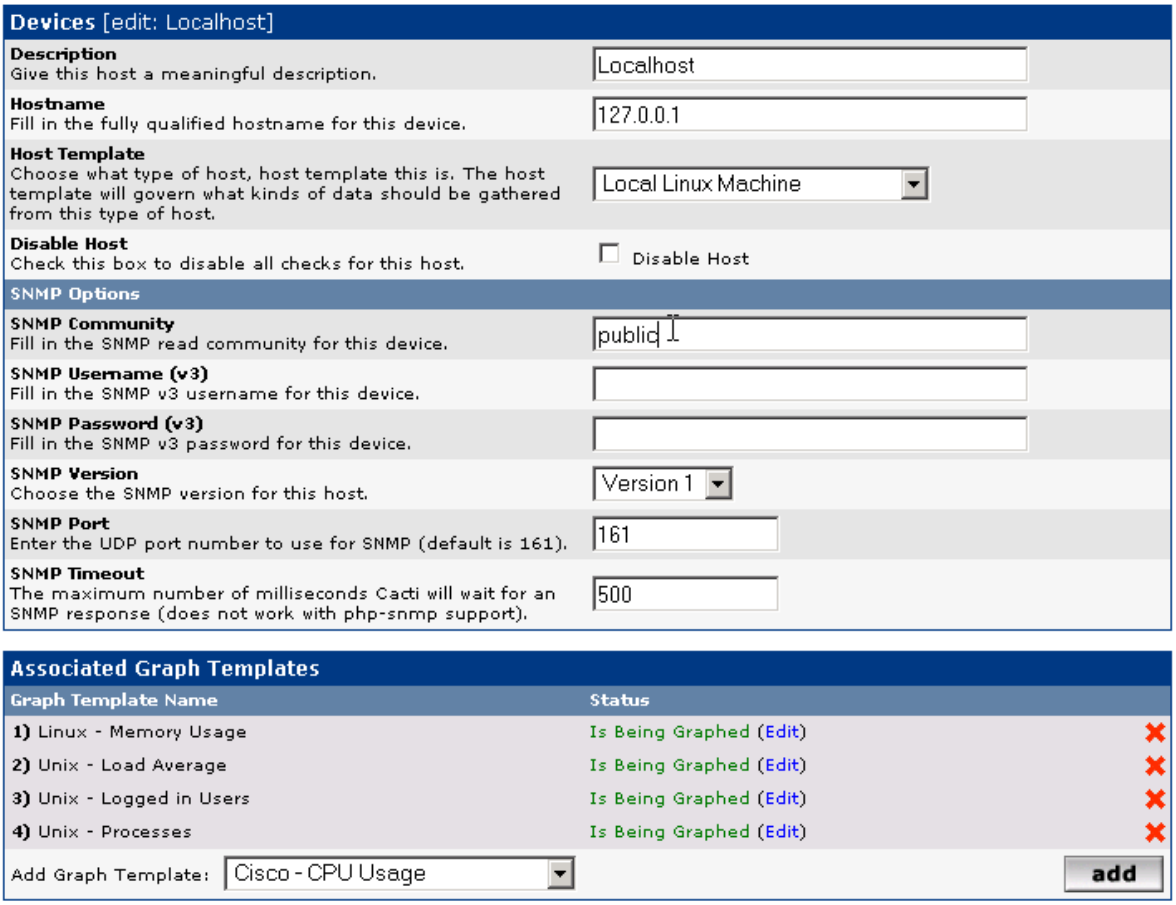


Fig. 28 Ventana de Configuración del CACTI.

Guardar los cambios en el botón **Save** que se encuentra mas abajo, aparecerá una página semejante a la figura 27 ahí volver a dar click en Localhost, y si toda la configuración es correcta mostrará información snmp.

Ejemplo:

Localhost (127.0.0.1)

SNMP Information

System: Linux linux-roxana 2.6.18.2-34-default #1 SMP Mon Nov 27 11:46:27 UTC 2006 i686

Uptime: 266084 (0 days, 0 hours, 44 minutes)
Hostname: linux-roxana
Location: Nerlyng
Contact: Roxana

Para saber si funciona correctamente ejecutar el comando
linux-snmp:/srv/www/htdocs/cacti # php poller.php
OK u:0.00 s:0.01 r:0.13
03/29/2007 08:09:50 PM - SYSTEM STATS: Time:1.3530 Method:cmd.php Processes:1 Threads:N/A Hosts:2 HostsPerProcess:2 DataSources:5 RRDsProcessed:5
OK u:0.00 s:0.01 r:0.20
OK u:0.00 s:0.01 r:0.20
OK u:0.00 s:0.01 r:0.20
OK u:0.00 s:0.01 r:0.20
OK u:0.00 s:0.01 r:0.20
OK u:0.00 s:0.01 r:0.20
OK u:0.00 s:0.01 r:0.20
OK u:0.00 s:0.02 r:0.20
OK u:0.00 s:0.02 r:0.20

3.2.3 Uso de CACTI.

Agregar un dispositivo:

Para crear un nuevo dispositivo dar click en el menú **Devices**, luego dar click en **Add**

Mostrará la ventana siguiente:

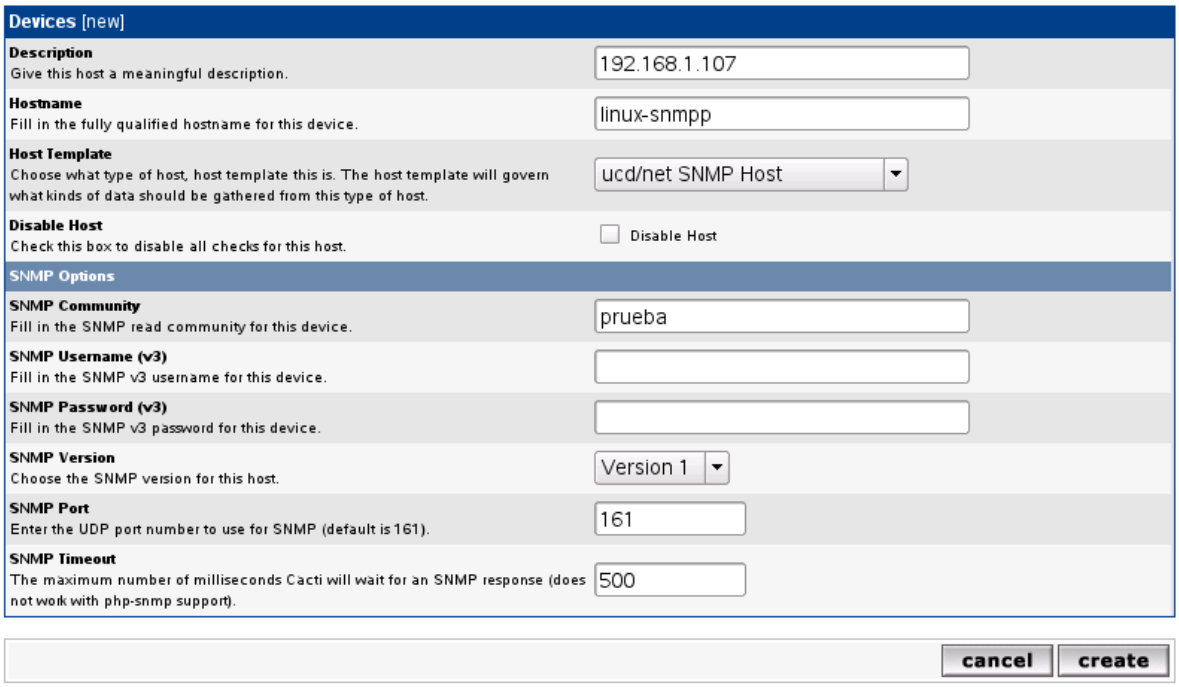


Fig. 29 Ventana para Agregar un nuevo dispositivo en CACTI.

Los únicos campos que requieren entrada son:

Description = que será la dirección Ip del dispositivo a monitorizar

Hostname = un nombre que identifique al dispositivo de manera fácil

Si el tipo de host que se está creando se encuentra en la lista de **Host Template**, se debe seleccionar, en caso de que no se encuentre, seleccionar Generic SNMP-enabled Host. También se deberá copiar el nombre de la comunidad que el dispositivo tiene para que se puedan leer las Mib, así como también seleccionar la versión y el puerto que utiliza.

Una vez terminado todo, click en el botón **create.**

Agregar gráficos.

Seleccione la opción del menú **New Graphs** que está bajo el título **Create**. Usted verá una pantalla similar a la imagen de abajo.

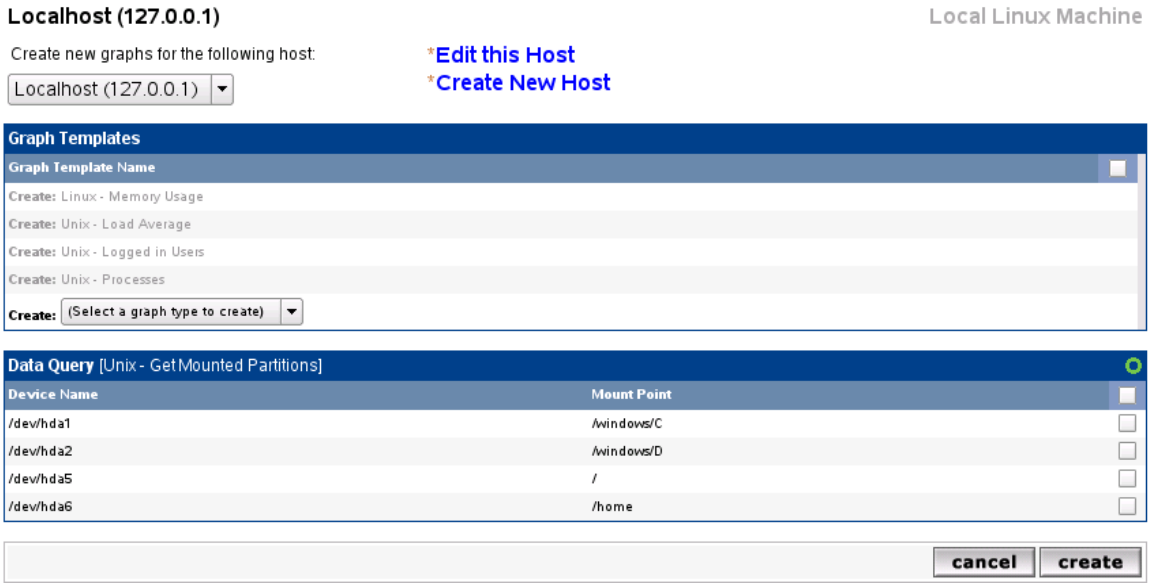


Fig. 30 Ventana para Agregar un nuevo gráfico en CACTI.

El dropdown que tiene como etiqueta (Create new host for the following host:) debe usarse para seleccionar el dispositivo al cuál se le quiere crear un gráfico. El concepto básico en ésta página es simple, en la caja Graph Templates seleccionará el tipo de gráfico que desea con el dropdown que tiene como etiqueta **Creates** y luego click en el botón create que se encuentra más abajo.

Agregar gráfico de tráfico.

Del menú seleccionar Devices, click en el dispositivo al cual queremos crearle el gráfico, seleccionar en la caja **Associated data Queries** en el dropdown **Add Data Query** (SNMP-Interface Statistics), click en **add.**

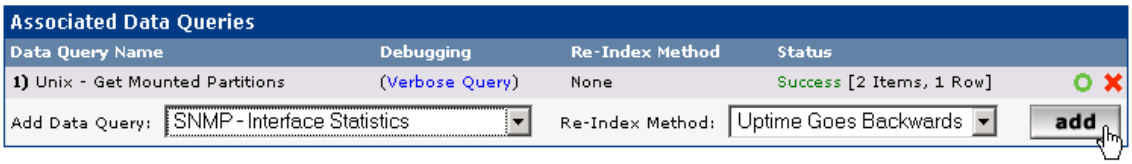


Fig. 31 Agregar un gráfico de Trafico.

Guardar los cambios en **Save.**

Ahora en el menú **New Graphs,** verificar la caja Data Query[SNMP-Interfaces Statistic], colocar un check al interfaz al cual se le desea hacer un gráfico. Luego seleccionar el tipo de gráfico en el dropdown que tiene como etiqueta **Select graph type**, por último click en el botón create que se encuentra abajo.

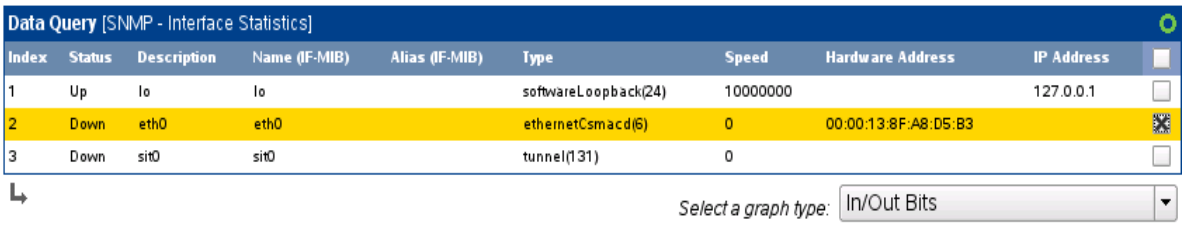


Fig. 32 Verificación del Gráfico generado.

3.3 Wireshark.

3.3.1 Descripción.

Para observar y analizar el comportamiento de los protocolos de red es preciso disponer de una herramienta capaz de monitorear el tráfico en la red y mostrarlo en una forma legible. Las herramientas que capturan y muestran el tráfico existente en una interfaz de red se denominan analizadores de protocolos de red, analizadores de paquetes, "packet sniffers" o simplemente "sniffers" (del inglés sniff, olfatear). Para visualizar el tráfico los analizadores de protocolo colocan la tarjeta de red en modo promiscuo, una modalidad en la cual es capturado todo el tráfico visible para la tarjeta de red. En una red Ethernet una interfaz de red en modo promiscuo puede ver todo el tráfico generado por todos los equipos que comparten el mismo conjunto de cables y concentradores (hubs). El modo promiscuo implica riesgos evidentes de seguridad, por lo que su uso suele limitarse al supervisor.

Ethereal, ahora Wireshark, es un popular software de código abierto que ayuda en el análisis de protocolos de red, desarrollado por expertos en redes de todo el mundo y disponible en diferentes plataformas como Linux, OS X y el sistema Windows. El fundador de Ethereal ha decidido dar un giro al proyecto comenzando por el nombre y web oficial de producto.

Wireshark es un proyecto de software open source, liberado bajo la Licencia Pública GNU (GPL-General Public License).Todo el código fuente está gratuitamente disponible bajo la GPL. Se puede modificar Wireshark para adecuarlo a las propias necesidades.

Wireshark es una herramienta muy útil para analizar las comunicaciones, tanto a nivel de conexión como a nivel del tráfico que se intercambia. Proporciona información muy útil a

la hora de entender los protocolos, pudiendo incluso detallar con mucha precisión algunos protocolos que reconoce, como el de DNS, IRC y muchos otros.

Es un analizador de protocolos con interfaz gráfica capaz de reconocer muchos protocolos distintos. Permite tanto revisar los paquetes de datos en una red activa como desde un archivo de captura previamente generado; es capaz de comprender diversos formatos de archivo propios de otros programas de captura, en particular el clásico tcpdump.

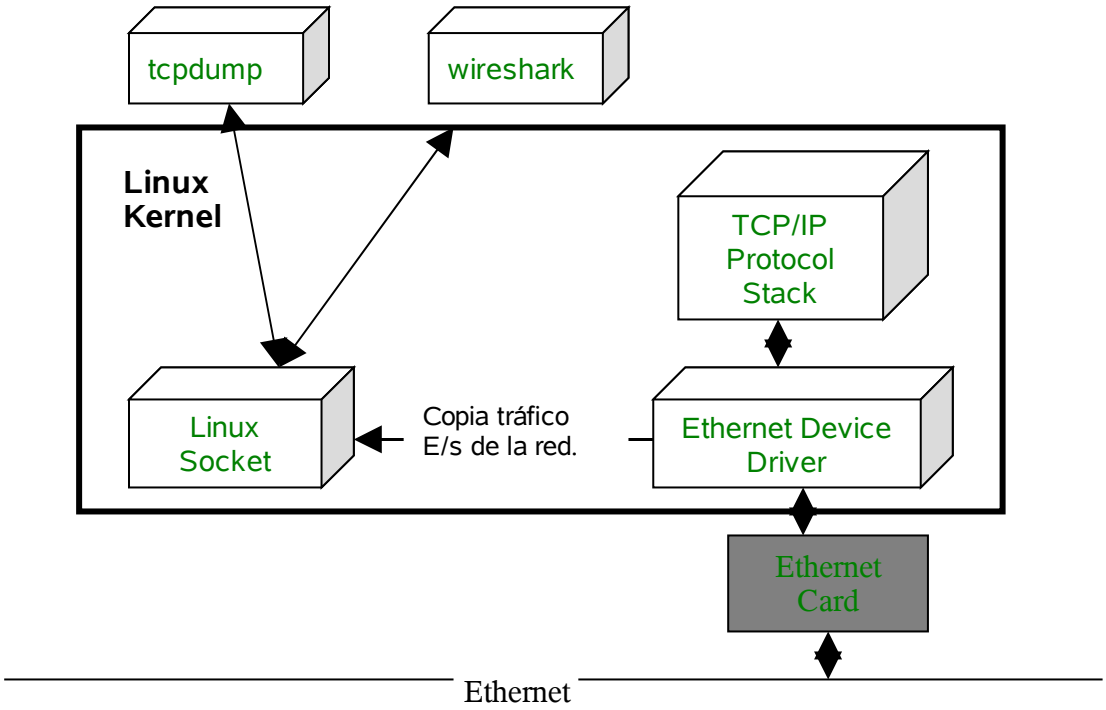


Fig. 33 Arquitectura de software para un analizador de protocolo en Linux.

La arquitectura de software para un protocolo de red en una máquina Linux con tarjeta Ethernet aparece en la figura. El analizador de protocolo corre como una aplicación, comunicándose con un componente del kernel Linux llamado Linux Socket Filter. El kernel de un sistema operativo es la parte central o núcleo del sistema, el socket (significa

enchufe) es una forma de comunicación entre procesos propia de los sistemas Unix, un proceso es un programa en ejecución. La figura 33 muestra dos analizadores de protocolo distintos, tcpdump y wireshark; tcpdump funciona en línea de comando, wireshark es una aplicación gráfica, pero ambos hacen más o menos lo mismo. El Linux Socket Filter actúa como intermediario entre el analizador de protocolo y el controlador de la tarjeta de red Ethernet (Ethernet device driver); coloca la tarjeta de red en modo promiscuo y obtiene una copia de todo el tráfico entrante desde la red y todo el tráfico saliente hacia la red. El socket de filtro procesa este tráfico y lo transfiere al analizador de protocolo, que lo presenta al usuario.

El analizador interpreta los protocolos desde el nivel de enlace al nivel de aplicación. La lista de protocolos que abarca para cada uno de los niveles es muy completa.

3.3.2 Intalación de Wireshark.

Para usar Wireshark, se debe:

1. Obtener un paquete binario para su sistema operativo, o
2. Obtener el código fuente y compilar Wireshark para su sistema operativo.

Actualmente, una de las distribuciones de Linux que trae integrado el Wireshark es Suse. Ninguna otra versión de UNIX incluye Wireshark hasta ahora, y Microsoft no lo incluye en ninguna versión de Windows. Por ésta razón, es necesario saber dónde conseguir la última versión de Wireshark y cómo instalarlo.

La versión actual de Wireshark es la 0.10.9.

En general, para tener Wireshark funcionando se deberían seguir los pasos siguientes:

1. Descargue el paquete relevante para sus necesidades, por ejemplo, la distribución fuente o binaria.
2. Compilar el código fuente a un binario, si se ha descargado el fuente. Esto puede implicar compilar y/o instalar cualquier otro paquete necesario.
3. Instalar los binarios en sus destinos finales.
4. Si estamos trabajando con la distribucion de Suse como es nuestro caso solo tiene que:

   4.1 Desde el menú de invocación de ambiente gráfico ejecutar el Yast. Si no está como administrador le pedirá la contrasena de administrador.

   4.2 En el menú Software elija la opción Actualización en línea.

   4.3 Introduzca el DVD de instalación de Suse o el primer CD de instalación.

   4.4 En la categoría Filtro elija Buscar y escriba en la caja de texto Wireshark.

   4.5 Marque el software que quiere instalar, luego Aceptar y Finalizar; y ya tiene instalado Wireshark.

3.3.3 Uso de Wireshark.

Hay dos métodos que se pueden usar para capturar paquetes con Wireshark:

1. Desde la línea de comandos introduciendo:
   linux-snmp:~ # wireshark -i eth0 -k
   o bien
   linux-snmp: wireshark&

Esta orden arranca el programa y devuelve el control de la terminal al usuario para poder continuar ingresando comandos. El símbolo & arranca el programa como proceso independiente de la terminal. La figura muestra la ventana principal de Wireshark luego de una captura de datos. Inicialmente, ésta ventana aparece vacía:

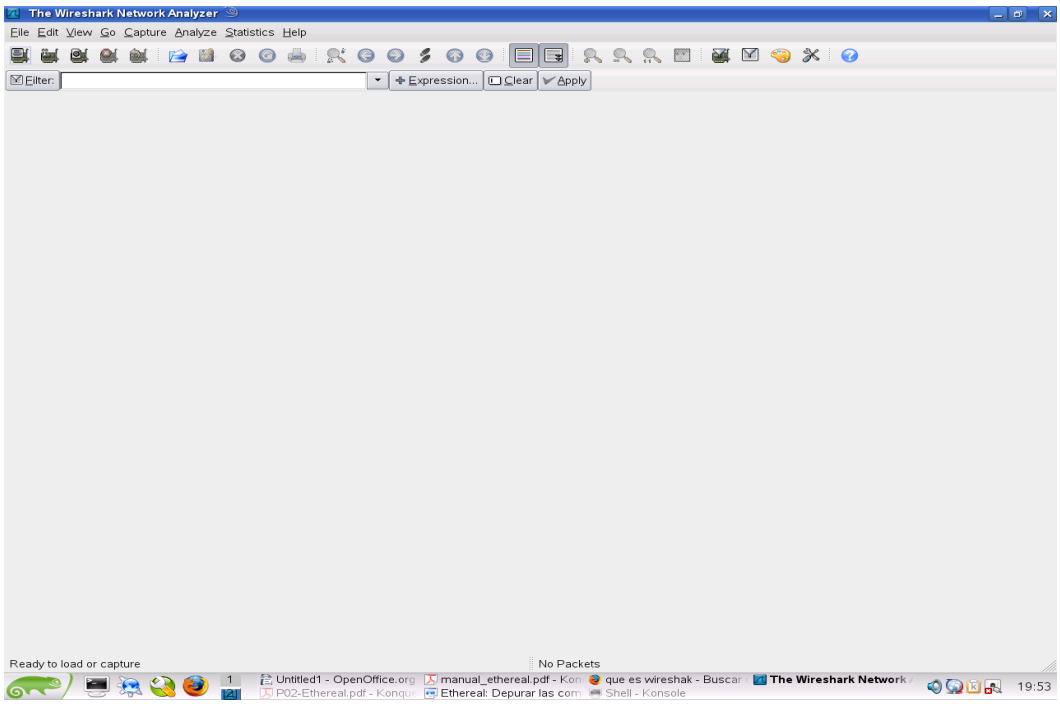


Fig. 34 Ventana Principal de Wireshark.

2. A través del menú de invocación del ambiente gráfico

En la ventana principal de Wireshark se reconocen tres áreas de despliegue:

- El panel superior es el panel de la lista de paquetes. Muestra un resúmen de cada paquete capturado. Pulsando en los paquetes de este panel se controla lo que se muestra en los otros dos paneles. Resúmen de paquetes capturados, un paquete por línea. Al desplazarse en la lista y cambiar el paquete actual se actualizan las otras dos ventanas, donde se despliega en dos formatos diferentes el contenido del paquete.
- El panel intermedio es el panel de vista en árbol. Muestra el paquete seleccionado en el panel superior en más detalle. Detalles de encabezado de protocolos para el

paquete seleccionado; los encabezados pueden abrirse (clic en >) para ver mayor detalle, o cerrarse (clic en v) para ocupar sólo una línea.

- El panel inferior es el panel de vista de datos. Muestra los datos del paquete seleccionado en el panel superior, y resalta el campo seleccionado en el panel de vista en árbol. Datos crudos del paquete, representación hexadecimal y ASCII del encabezado del paquete seleccionado en el campo del medio.

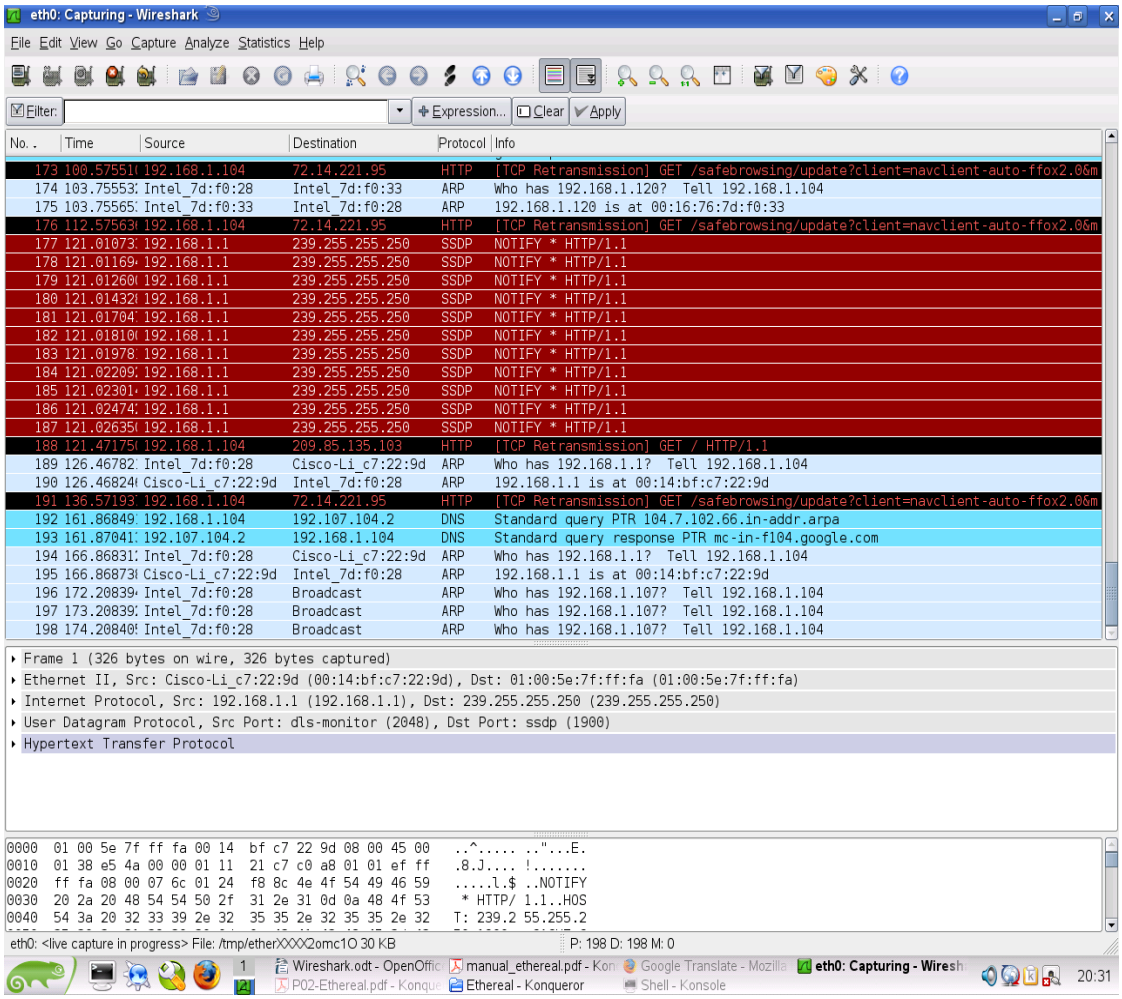


Fig. 35 Ventana principal de Wireshark luego de una captura.

Para iniciar la captura de datos, elegir las opciones de menú Capture: Start (Capturar, Comienzo). En la ventana de opciones de captura (ver figura), debe fijarse al menos la interfaz sobre la que se quiere realizar la captura, los nombres varían según los sistemas operativos; o bien dar click en el botón que aparece mas a la izquierda y se nos mostrará una caja de diálogo como la siguiente:

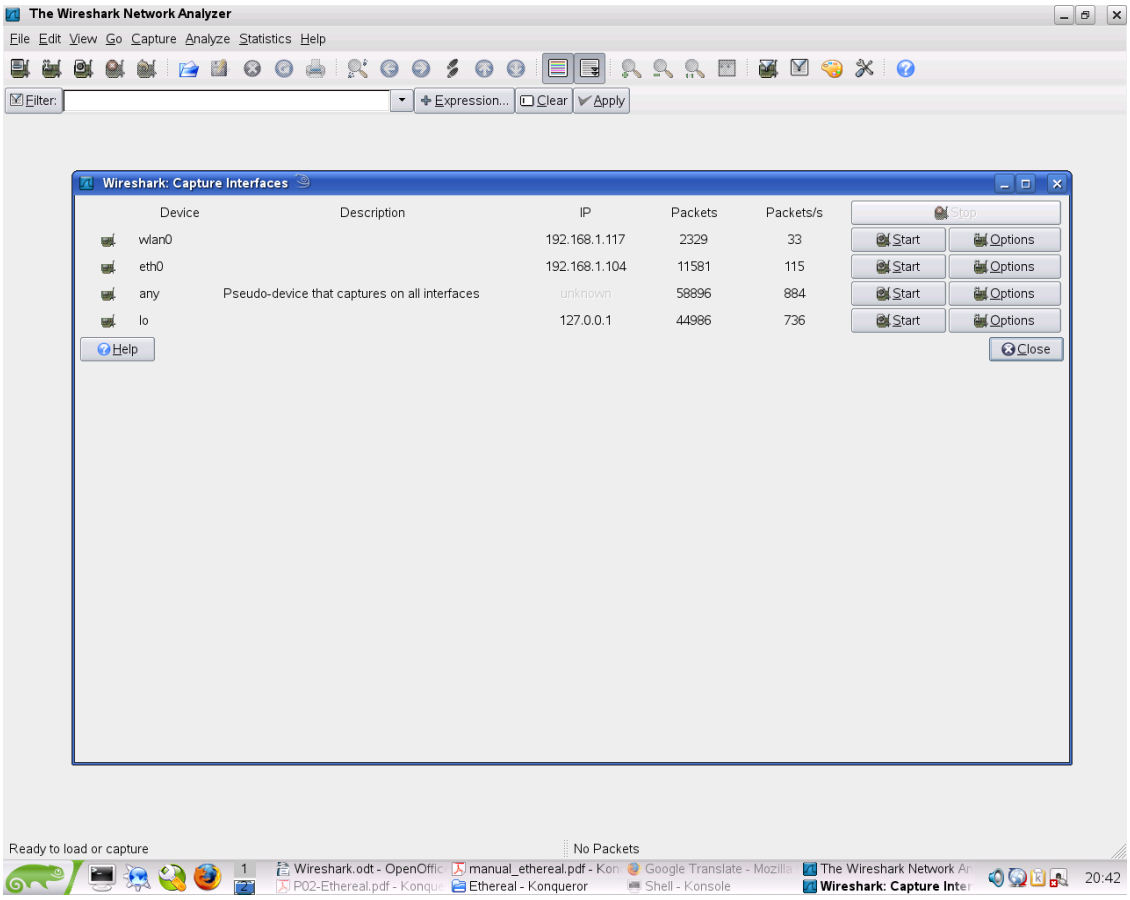


Fig. 36 Interfaces disponibles para la captura.

Filtrado de paquetes.

El filtrado de paquetes permite capturar o desplegar sólo aquellos paquetes de interés para el estudio en curso, desconociendo la existencia de otros.

Wireshark tiene dos modos de filtro distintos:

- Filtro de captura: sólo se retienen los paquetes que cumplen la expresión filtro. Define lo que se guarda.
- Filtro de despliegue: de los paquetes capturados, sólo se muestran los paquetes que cumplen la expresión filtro. Define lo que se ve de lo que hay guardado.

La sintaxis de escritura de ambos tipos de filtro es diferente. Los filtros de captura siguen la sintáxis del comando tcpdump. Este tipo de captura la conseguimos en el menú Capture opción Capture Filter y deben ser escritos en el cuadro Filter de la ventana de opciones de captura, antes de iniciar la captura.
Los filtros de despliegue se fijan en el botón Filter de la barra de herramientas de la ventana principal de Wireshark.

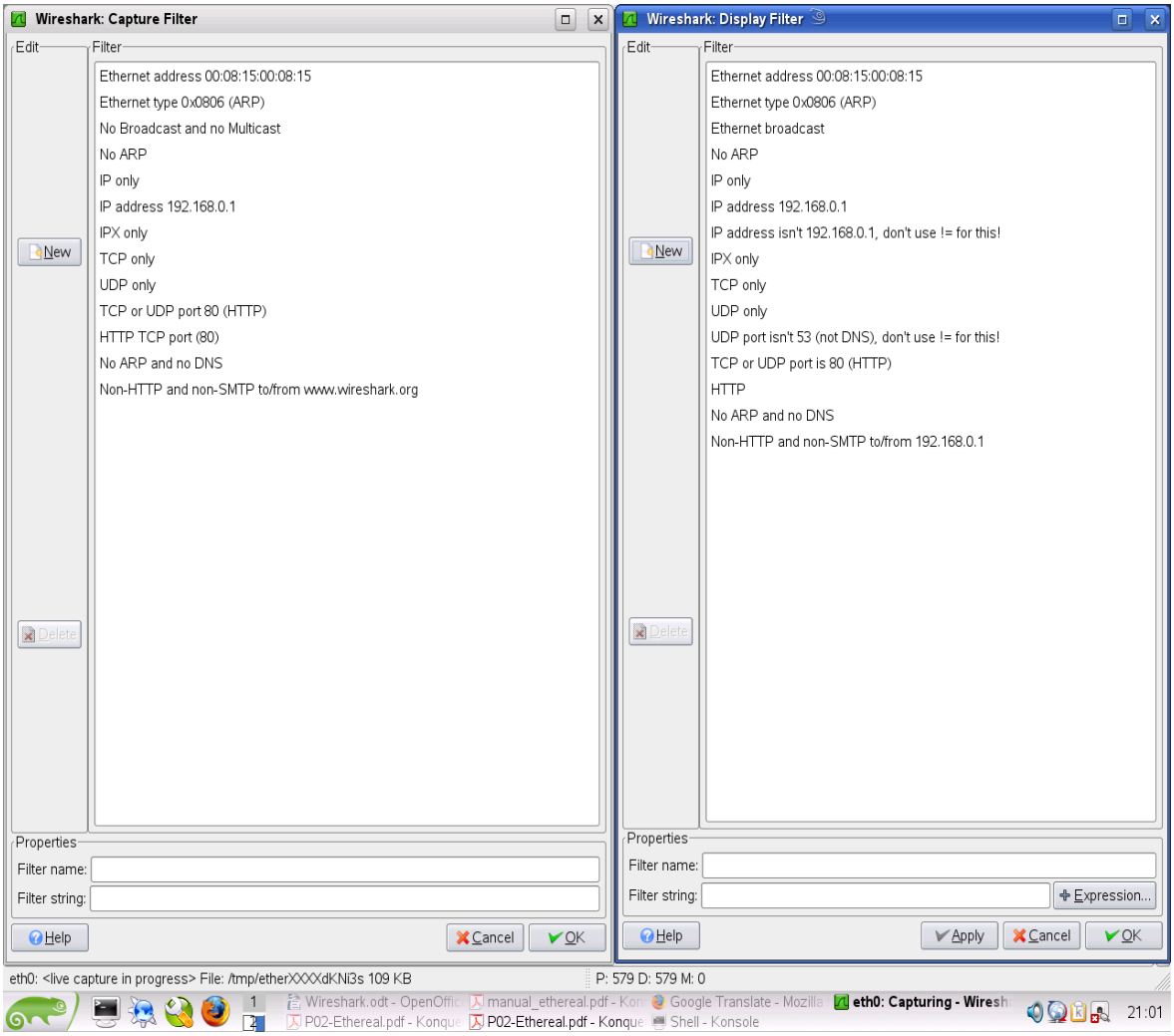


Fig. 37 Dialogo para construir un Filtro de Captura y un Filtro de Despliegue.

Además de los tres paneles principales, hay seis elementos de interés en la barra de filtros, situada en la parte inferior de la ventana principal de Wireshark.

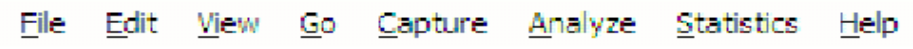
A) El botón inferior situado más a la izquierda, etiquetado "Filter:", se puede pulsar para que aparezca la ventana de construcción de filtros.

B) El cuadro de texto de la parte izquierda proporciona un área para introducir o editar expresiones de filtro. Es también donde se muestra el filtro que está actualmente en efecto. Se puede pulsar en la flecha desplegable para seleccionar expresiones de filtro anteriores de una lista.
C) El botón del medio etiquetado "+ Expresión...", lanza el asistente para crear expresiones de filtro.
D) El botón del medio a la derecha, etiquetado "Clear", borra el filtro actual.
E) El botón del medio a la derecha, etiquetado "Apply", aplica el filtro tecleado en el cuadro de texto anterior.
F) El cuadro de texto de la derecha muestra mensajes informativos. Estos mensajes pueden indicar si se está capturando o no, qué fichero se ha leído en el panel de lista de paquetes si no se está capturando. Si se ha seleccionado un campo de protocolo del panel de vista de árbol y es posible filtrar por ese campo entonces la etiqueta de filtro para ese campo de protocolo se mostrará.

Los menus de Wireshark.

El menú de Wireshark descansa a lo largo de la parte superior de la ventana de Wireshark, como se muestra en el ejemplo de la siguiente figura:

El Menú de Wireshark



Contiene los siguientes elementos:

File

Este menú contiene elementos de menú para abrir y releer ficheros de captura, guardar ficheros de captura, exportar bytes, imprimir ficheros de captura, y salir de Wireshark.

Edit

Este menú contiene elementos de menú para encontrar una trama, marcar una o más tramas, establecer referencias de tiempo y establecer las preferencias (cortar, copiar y pegar no están actualmente implementados).

View

Este menú contiene elementos de menú para modificar opciones de visualización, ampliar/reducir la fuente de los paneles, colorear tramas, expandir todas las tramas, colapsar todas las tramas, mostrar un paquete en una ventana aparte, y recargar el fichero de captura actual.

Go

Este menú contiene elementos de menú para desplazarse entre las tramas.

Capture

Este menú permite iniciar y detener capturas, y crear filtros de captura.

Analyze

Este menú contiene elementos de menú para seleccionar y filtrar paquetes coincidentes, seguir una sesión TCP, crear filtros de visualización, habilitar o deshabilitar la disección de protocolos, y configurar decodificadores especificados por el usuario.

Statistics

Este menú contiene elementos de menú para obtener un resumen de los paquetes que han sido capturados, mostrar estadísticas de jerarquía de protocolos, gráficos de entrada/salida, listas de conversaciones y hosts, tiempos de respuestas y distintos análisis de protocolos particulares.

Help

Este menú permite mostrar los plugins cargados, contiene el elemento de menú : **About Wiresharp...** y acceso a alguna Ayuda básica.

El menú File de Wireshark.

El menú File (Archivo) de Wireshark contiene los campos que se muestran en la siguiente tabla:

Elemento.	**Acelerador.**	**Descripción.**
Open...	Ctrl-O	Este elemento abre el cuadro de diálogo abrir archivo que permite cargar un fichero de captura para su visualización.
Open Recent		Permite volver a abrir los últimos archivos abiertos.
Merge…		Este elemento permite anexar un archivo de captura al actual.
Close	Ctrl-W	Este elemento cierra la captura actual. Si no se ha guardado la captura, se pierde.
Save	Ctrl-S	Este elemento guarda la captura actual. Si no se ha establecido un nombre de fichero de captura por defecto (quizás con la opción -w <capfile>), Wireshark lanza cuadro de diálogo Guardar Fichero de captura Como . Nota!: Si ya ha guardado la captura actual, este menú estará deshabilitado. Nota!: No se puede guardar una captura mientras está en progreso. Hay que detener la captura para poder aguardarla..
Save As...	Mayús- Ctrl-S	Este elemento permite guardar el fichero de captura actual a cualquier fichero que se desee. Lanza el cuadro de diálogo Guardar Fichero de captura Como (que se describe más adelante en la sección El cuadro de diálogo Guardar Fichero de captura Como)
Export		Este elemento permite exportar los bytes

		seleccionados de un paquete a un archivo binario de su elección.
Print...		Este elemento permite imprimir todos o una selección de los paquetes del fichero de captura. Lanza el cuadro de diálogo Imprimir de Wireshark (que se describe más adelante en la sección Imprimiendo paquetes).
Quit	Ctrl-Q	Este elemento permite salir de Witreshark. Si no se ha guardado el fichero de captura actual, pregunta si se desea hacerlo antes de salir.

El menú Edit de Wireshark.

El menú Edit (Editar) de Wireshark contiene los campos que se muestran en la siguiente tabla:

Elemento	**Acelerador**	**Descripción**
Find Packet...	Ctrl-F	Este elemento muestra un cuadro de diálogo que permite encontrar una trama introduciendo un filtro de visualización de Wireshark.
Find Next	Ctrl-N	Continúa la última búsqueda hacia delante.
Find Previous	Ctrl-B	Continúa la última búsqueda hacia atrás.
Time Reference		Esta opción permite fijar referencias de tiempo en la captura actual (opción Set Time Referente (toggle)). En ésta referencias se pone a 0 el contador de tiempos del formato “Segundos desde el comienzo de la captura” (ver la sección El menú View). También permite desplazarse entre referencias de tiempo (con las opciones Find Next y Find Previous)
Mark Packet	Ctrl-M	Este elemento "marca" la trama actualmente seleccionada. Vea la sección El cuadro de diálogo Guardar Fichero de captura Como

El menú View de Wireshark.

El menú View (Ver) de Wireshark contiene los campos que se muestran en la siguiente tabla:

Elemento	**Descripcion.**
Main Toolbar	Permite mostrar u ocultar la barra de herramientas principal.
Filter Toolbar	Permite mostrar u ocultar la barra de herramientas de filtros.
Statusbar	Este elemento permite mostrar u ocultar la barra de estado.
Packet List	Este elemento permite mostrar u ocultar el panel de lista de paquetes.
Packet Details	Permite mostar u ocultar el panel de detalles de paquetes.
Packet Bytes	Este elemento permite mostrar u ocultar el panel de bytes de paquete.
Time Display Format	Este elemento controla el modo en que Wireshark muestra las marcas de tiempo. Ofrece la siguientes opciones: **Time of day** Seleccionando este botón de radio se indica a Ethereal que muestre las marcas de tiempo en formato Hora del día. Este campo, "Date and time of day", "Seconds since beginning of capture" y "Seconds since previous frame" son mútuamente exclusivos. **Date and time of day** Seleccionando este botón de radio se indica a Wireshark que muestre las marcas de tiempo en formato Fecha y hora. "Time of day", este campo, "Seconds since beginning of capture" y "Seconds since previous frame" son mútuamente exclusivos. Seconds since beginning of capture Seleccionando este botón de radio se indica a Wireshark que muestre las marcas de tiempo en formato Segundos desde el comienzo de la captura. "Time of day", "Date and time of day",

	este campo y "Seconds since previous frame" son mútuamente exclusivos. Seconds since previous packet Este botón de radio indica a Wireshark que muestre marcas de tiempo en formato Segundos desde la trama anterior "Time of day", "Date and time of day", "Seconds since beginning of capture" y éste campo son mútuamente exclusivos.
Name Resolution	Este elemento controla el modo en que Wireshark muestra alguna información sobre los paquetes. En concreto, si las direcciones y otros números son traducidos. Ofrece la siguientes opciones: Resolve Name Este campo activa la resolución de nombres mientras se visualizan paquetes. Enable for MAC Layer Este campo, cuando se selecciona, le indica a Wireshark que traduzca los primeros tres octetos de las direcciones MAC (el identificador de fabricante) a nombres (cuando pueda). Enable for Network Layer Este campo, cuando se selecciona, le indica a Wireshark que traduzca direcciones IP a nombres de dominio (cuando pueda). Nota: Si se selecciona esta opción y el servidor DNS no está disponible Wireshark será muy lento ya que espera a que venza el temporizador para respuestas del servidor DNS. Enable for Transport Layer Cuando se selecciona, le indica a Wireshark que traduzca las direcciones del nivel de transporte (números de puerto TCP/UDP) a nombres de servicios well-known (donde pueda).
Auto Scroll in	Este elemento, cuando se selecciona, le indica a Wireshark que

Live Capture	haga scrolling del panel de lista de paquetes cuando se capturen nuevos paquetes.
Zoom In	Este elemento, cuando se selecciona, le indica a Wireshark que haga scrolling del panel de lista de paquetes cuando se capturen nuevos paquetes.
Zoom Out	Disminuye la fuente en la que se visualizan los datos de los distintos paneles.
Normal Size	Reestablece la fuente en la que se visualizan los datos de los distintos paneles a su tamaño original establecido en las Preferencias de Wireshark.
Collapse All	Wireshark mantiene un lista de todos los subárboles de protocolo que se expanden, y la utiliza para asegurar que los subárboles correctos se expanden cuando se muestra un paquete. Este elemento contrae la vista de árbol de todos los paquetes en la lista de captura.
Expand All	Este elemento expande todos los subárboles de todos los paquetes de la captura.
Expand Tree	Este elemento expande el árbol actualmente seleccionado.
Coloring Rules	Este elemento muestra un cuadro de diálogo que permite colorear paquetes en el panel de lista de paquetes en función de las expresiones de filtro que se escojan. Puede ser muy útil para distinguir ciertos tipos de paquetes.
Show Packet in New Window	Este elemento muestra el paquete seleccionado en una ventana aparte..Esta ventana sólo muestra los paneles de vista de árbol y de vista de byte.
Reload	Este elemento permite recargar el fichero de captura actual. Este elemento ya no es necesario, y puede ser eliminado en futuras versiones de Wireshark.

El menú Go de Wireshark.

El menú Go (Ir) de Wireshark contiene los campos que se muestran en la Tabla 3-2.

Elemento	Acelerador	Descripción
Back	Alt-Left	Va a la trama anterior.
Forward	Alt-Right	Va a la siguiente trama.
Go to First Packet		Va al primer paquete de la lista.
Go to Last Packet		Va al último paquete de la lista.
Go to Packet…	Ctrl-G	Este elemento muestra un cuadro de diálogo que permite especificar una trama a la que ir por número de trama.

El menú Capture de Wireshark.

El menú Capture (Capturar) de Wireshark contiene los campos que se muestran en la tabla siguiente:

Elemento	Acelerador	Descripción
Start...	Ctrl-K	Este elemento muestra el cuadro de diálogo Preferencias de Captura (descrito más adelante en la sección Capturando paquetes con Wireshark) y permite empezar a capturar paquetes.
Stop	Ctrl-E	Este elemento detiene la captura actualmente en curso.
Interfaces		Muestra los interfaces de captura disponibles y su tráfico actual.
Capture Filters...		Este elemento muestra un cuadro de diálogo que permite crear y editar filtros de captura. Se puede dar nombre a los filtros, y se pueden guardar para uso futuro.

El menú Analyze de Wireshark.

El menú Analyze (Analizar) de Wireshark contiene los campos que se muestran en la Tabla.

Elemento	Acelerador	Descripción.
Display Filters...		Este elemento muestra un cuadro de diálogo que permite crear y editar filtros de visualización. Se puede dar nombre a los filtros, y se pueden guardar para uso futuro.
Apply as Filter		Este elemento permite seleccionar todos los paquetes que tienen un valor coincidente en el campo seleccionado en el panel de vista de árbol (panel central), componer la expresión de filtro para ellos y hacerla, añadirla o excluirla de la expresión de filtro actual. Esta opción aplica el filtro resultante inmediatamente.
Prepare a Filter		Este elemento es igual al anterior, sólo que no aplica
Enabled Protocols...	Mayús-Ctrl-R	Este elemento muestra un cuadro de diálogo que permite habilitar o deshabilitar la disección de protocolos individuales.
Decode As...		Permite forzar a Wireshark a decodificar o no los protocolos indicados en el nivel de enlace (ethertype), de red (protocolo IP) o de transporte (puerta origen, destino o ambas) como un protocolo particular.
User Specified Decodes...		Este elemento permite al usuario ver la lista de asociaciones de protocolos definidas por el usuario con el elemento anterior.
Follow TCP Stream		Este elemento abre una ventana aparte y muestra todos los segmentos TCP capturados que están en la misma conexión TCP que el paquete seleccionado. Los datos en el flujo TCP se ordena, y los segmentos

		duplicados se borran, y es entonces mostrado en ascii. Se puede cambiar el formato si se desea o mostrar sólo un sentido de la conversación, así como guardar, imprimir o filtrar el flujo TCP.

El menú Statistics de Wireshark.

El menú Statistics (Estadísticas) de Wireshark contiene los campos mostrados en la Tabla:

Elemento	**Descripción**
Summary	Este elemento muestra una ventana de estadísticas con información sobre los paquetes capturados.
Protocol Hierarchy Statistics	Este elemento muestra un árbol jerárquico de estadísticas de paquetes
Conversations	Este elemento muestra un resumen de todas las conversaciones existentes. Los prootcolos pueden ser Ethernet, FDDI, Fibre Channel, IPX, IPv4, TCP (IPv4 y IPv6), Token Ring o UDP (IPv4 y IPv6).
Endpoints	Este elemento muestra un resumen de todos los nodos (hosts) existentes. Los protocolos pueden ser Ethernet, FDDI, Fibre Channel, IPX, IPv4, TCP (IPv4 y IPv6), Token Ring o UDP (IPv4 y IPv6).
IO Graphs	Este elemento muestra una gráfica de las tramas capturadas en función del tiempo.
Conversation List	Este elemento muestra una lista de todas las conversaciones existentes y el tráfico en uno y otro sentido según un protocolo concreto. Los prootcolos pueden ser Ethernet, FDDI, Fibre Channel, IPX, IPv4, TCP (IPv4 y IPv6), Token Ring o UDP(IPv4 y IPv6).
Endpoint List	Este elemento muestra una lista de todos los nodos (hosts) existentes y el tráfico entrante y saliente según un protocolo

	concreto. Los protocolos pueden ser Ethernet, FDDI, Fibre Channel, IPX, IPv4, TCP (IPv4 y IPv6), Token Ring o UDP(IPv4 y IPv6).
Service	Este elemento muestra estadísticas de tiempos de respuesta ante distintos Response Time servicios. Los servicios pueden ser DCE-RPC, Fibre Channel, ITU-T H.225 RAS, LDAP, ONE-RPC o SMB.
ANSI	Este elemento lleva la cuenta de las llamadas a las distintas funciones de distintos protocolos ANSI. Incluye A-Interface BSMAP, A-Interface DTAP y la operación MAP.
BOOTP-DHCP	Este elemento lleva la cuenta de los distintos tipos de mensajes DHCP.
GSM	Este elemento lleva la cuenta de las llamadas a las distintas funciones del protocolo GSM. Incluye A-Interface BSSMAP, A-Interface DTAP (incluyendo el control de llamada; la gestión de la movilidad GPRS, de la sesión GPRS, de la movilidad y de los recursos de radio; el servicio de mensajes cortos; y los servicios suplementarios) y la operación MAP.
H.225	Lleva la cuenta de los distintos tipos o razones de mensajes ITU-T H.225.
H.223 Conversations	Lleva la cuenta y analiza las conversaciones ITU-T H.223.
HTTP	Este elemento lleva la cuenta de las solicitudes HTTP y las distintas respuestas obtenidas (informativas, éxito, etc.).
ISUP Message Types	Este elemento lleva la cuenta de los distintos tipos de mensajes ISUP.
MTP3	Este elemento analiza el tráfico MTP3.
ONC-RPC	Este elemento lleva la cuenta de las llamadas a los distintos programas ONE Programs RPC y sus tiempos de respuesta.
RTP	Este elemento muestra las estadísticas de los flujos RTP de la captura o analiza un flujo RTP concreto.
SIP	Este elemento lleva la cuenta de los distintos tipos de mensajes SIP.
TCP Stream	Este Elemento Analiza Gráficamente Flujos TCP. Permite

Graph	representar el RTT el throughput o la secuencia de tiempos
WAP-WSP	Este elemento lleva la cuenta de los distintos tipos de PDU y códigos de estado.

El menú Help de Wireshark.

El menú Help (Ayuda) de Wireshark contiene los campos que se muestran en la Tabla :

Elemento	**Descripción.**
Contents	Este elemento muestra un sistema de ayuda básico.
Supported Protocols…	Este elemento muestra una lista de los protocolos soportados por Wireshark y los campos disponibles para crear filtros de visualización.
Manual Pages	Este elemento muestra la ayuda HTML de Wireshark.
Ethereal online	Este elemento conecta con el web de Wireshark.
About Wireshark	Este elemento muestra una ventana con información simple sobre Wireshark

3.4 NTOP.

3.4.1 Descripción.

NTOP. Network Top nació con la idea de ser precisamente eso: algo parecido al comando Top (Utilización del CPU, estadísticas de procesos, utilización de memoria -top lo monitoriza todo) pero centrado en protocolos y tráfico de red. Es preciso ser un verdadero experto para sacarle todo el jugo a esta herramienta pero es bastante fácil sacarle partido aún quedándonos en la superficie de la cantidad de información que nos ofrece. Muchos cortafuegos basados en LINUX lo implementan como herramienta de información y estadísticas, así que no es difícil que nos encontremos con él en alguno de nuestros dispositivos aún sin habernos propuesto instalarlo. Existe una versión limitada para Windows (la versión completa es de pago para este sistema) pero puesto que se trata de un programa de fuente abierta si quieres hacerlo funcionar bajo Windows sólo tienes que bajarte el código y compilarlo. Ofrece gráficos por interfaz, por protocolo, por puerto, por ip/red, etc.

Es un software que permite visualizar la actividad de la red brindando datos de las máquinas que están utilizando la red en ese momento. Esta herramienta es de gran utilidad básicamente para tener información sobre números IP, tráfico que está generando cada máquina, porcentaje de saturación de la red, datos enviados y recibidos en tiempo real los usuarios y aplicaciones que están consumiendo recursos de red en un instante concreto, por lo tanto no debe faltar al administrador de red. Pero además proporciona una amplia gama de otros datos en forma de tablas y gráficos que facilitan su lectura. Los datos se actualizan automáticamente.

Es capaz de ayudarnos a la hora de detectar malas configuraciones de algún equipo (esto salta a la vista porque al lado del IP del host sale un banderín amarillo o rojo, dependiendo si es un error leve o grave), o a nivel de servicio. Posee un microservidor

Web que permite que cualquier usuario, que sepa la clave, pueda ver la salida NTOP de forma remota con cualquier navegador, y además es GNU. El software esta desarrollado para plataformas Unix y Windows.

En Modo Web, actúa como un servidor de Web, volcando en HTML el estado de la red. Viene con un recolector/emisor NetFlow/sFlow, una interfaz de cliente basada en HTTP para crear aplicaciones de monitoreo centradas en top, y RRD (Round Robin Databases) para almacenar persistentemente estadísticas de tráfico.

Los protocolos que es capaz de monitorizar son: TCP/UDP/ICMP, (R)ARP, IPX (Internetwork Packet Exchange; Intercambio de Paquetes Interred)., DLC, Decnet (Arquitectura de red propietaria de Digital (DEC) que es un sistema para conectar una red de ordenadores punto-a-punto, X.25 y Ethernet.), AppleTalk (Protocolo propietario que se utiliza para conectar ordenadores Macintosh de Apple en redes locales), Netbios, y ya dentro de TCP/UDP es capaz de agruparlos por FTP, HTTP, DNS, Telnet, SMTP/POP/IMAP, SNMP, NFS, X11.NTOP permite leer cualquier archivo del sistema.

La utilidad NTOP incluye un microservidor Web que, activado, permite que cualquier usuario con un navegador y con el conocimiento de la clave correcta pueda visualizar la salida de NTOP de forma remota y en tiempo real.

Por otra parte, si el fichero de control de acceso no existe, cualquier usuario con un navegador puede acceder a NTOP. Lo correcto sería que si el fichero de acceso no existe, o bien se deniegue el acceso a todo el mundo, o bien NTOP debe "quejarse" al ser invocado con la orden de activar el microservidor Web.

Algunos informes señalan que bajo determinadas circunstancias, NTOP puede permitir ejecutar código arbitrario con los privilegios de ROOT, ya que parecen haberse descubierto numerosos problemas de "buffer overflow" y NTOP debe ser ejecutado como "root" (o ser SETUID a "root") para que pueda analizar el tráfico de red.

Las versiones de NTOP vulnerables son las anteriores a la 1.3.1, al menos. La recomendación es eliminar el SETUID (bandera "+s") de NTOP, de forma que sólo sea ejecutable por "root", y no emplear nunca la opción de microservidor Web.

3.4.2 Instalación y Configuración.

Tenemos tres opciones para instalarlo:

(NOTA: se instala en el servidor principal):

1. Nada más sencillo que bajarse los fuentes y hacerle el :

- **./configure**
- **make**
- **make install**, pero conviene estar atentos por si no encuentra alguna que otra librería.

2. También se puede hacer siguiendo las instrucciones de su página principal y utilizando "cvs" (hay que instalar este paquete) para su descarga: **http://www.ntop.org/download.html**
3. Por último y ***más sencilla aun***:
   linux-snmp: apt-get install ntop.

Para hacerlo funcionar con todas sus prestaciones, conviene tener instalado:

- **GDChart**: Es un programa para poder hacer gráficos. Ya viene integrado en el fichero ntop-current.tgz, pero también hay que dejarlo instalado.
- **lsof:** Es un programa capaz de listar que ficheros están abiertos en el sistema.
- **nmap:** Es un programa capaz de escanear una red de ordenadores en busca de información.

- **Librerías OpenSSL**: Para poder optar a que el servidor Web acepte conexiones seguras (SSL).
- **Servidor MySQL:** Para permitir almacenar toda la información en una base de datos.

Para poder conectar NTOP con el MySQL, hay que ayudarse de un pequeño programita hecho en Perl que viene dentro del paquete y se llama "mySQLserver.pl". Evidentemente se ha de tener el módulo DBI de Perl (perl_DBI) para acceder al MySQL.

Y además hacer lo siguiente:

linux-snmp: cd /etc/sysconfig/
linux-snmp: pico ntop; Aquí cambiar el valor a las siguientes directivas:

NTOP_PORT=”192.168.1.0:3000”;

Los primeros cuatro números representan el número IP de la red que deseas monitorizar y el siguiente número después de : es el puerto por el que accederás a NTOP.

NTOPD_IFACE=”eth0” ;

Arrancar el programa

Ahora ya estamos en condiciones de verlo funcionar, o sea que vamos allá. Para arrancarlo por primera vez, se hace con cualquiera de las siguientes órdenes:

a) **linux-snmp: /etc/init.d/# ./ntop start**

b) **linux-snmp:/usr/bin/ntop –P /var/lib/ntop –i eth0 –u wwwrun –w 127.0.0.1:3000**

Acepta muchas condiciones, de las que se destacan estas:

-P /var/lib/ntop -w 3000 -W 3003 -i eth0 -b localhost:4000 -d -E -L -a /www/logs/ntop.log

-P /var/lib/ntop: Donde se dejan las tablas hash.

-w 3000: Abrimos el servidor en el puerto 3000.

-W 3003: Abrimos el servidor SSL en el puerto 3003.

-i eth0: Escuchamos todo el tráfico que pasa por la tarjeta de red eth0.

-b localhost:4000: Donde está el programa fuente para el servidor MySQL.

-d: Para que se convierta en demonio.

-E: Para que se ayude de herramientas externas (lsof, nmap, ...).

-L: Habilita la salida al syslog.

-a /www/logs/ntop.log: Es el fichero de acceso a la página Web del ntop.

3.4.3 Uso del Programa.

Ejecutar y mostrar la Información del Programa.

Una vez que ha arrancado, podemos ver qué está pasando en nuestra red visitando la página Web:

http://192.168.1.30:3000/

Se mostrará una página como la siguiente:

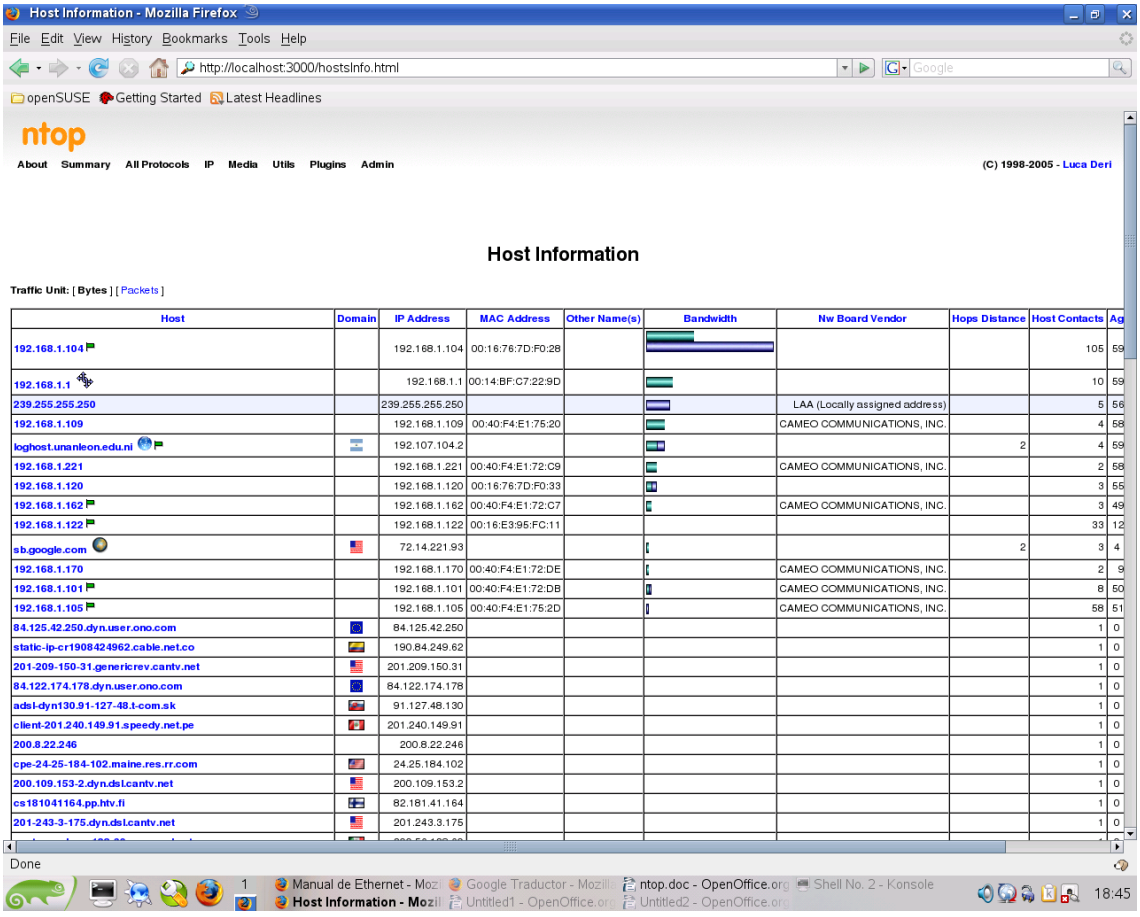


Fig. 38 Ventana Principal de NTop.

El menú de navegación principal se encuentra en el frame de arriba, y nos permite ver las siguientes opciones:

- **About**: Muestra una explicación del programa, como la versión, configuración, ayuda y reportes de problemas así como los créditos de las personas que lo han hecho.

- **Summary**: En este menú tenemos opciones como:

Traffic: Nos da un reporte del tráfico global generado incluyendo gráficos por paquetes, por tráfico, por Host Remoto, por carga de red y un historial del tráfico; gráficas de la distribución global de protocolo; gráficos de distribución global de protocolo TCP/UDP y gráficos por distribución de puertos. A continuación la pantalla que se nos presentara si elegimos esta opción:

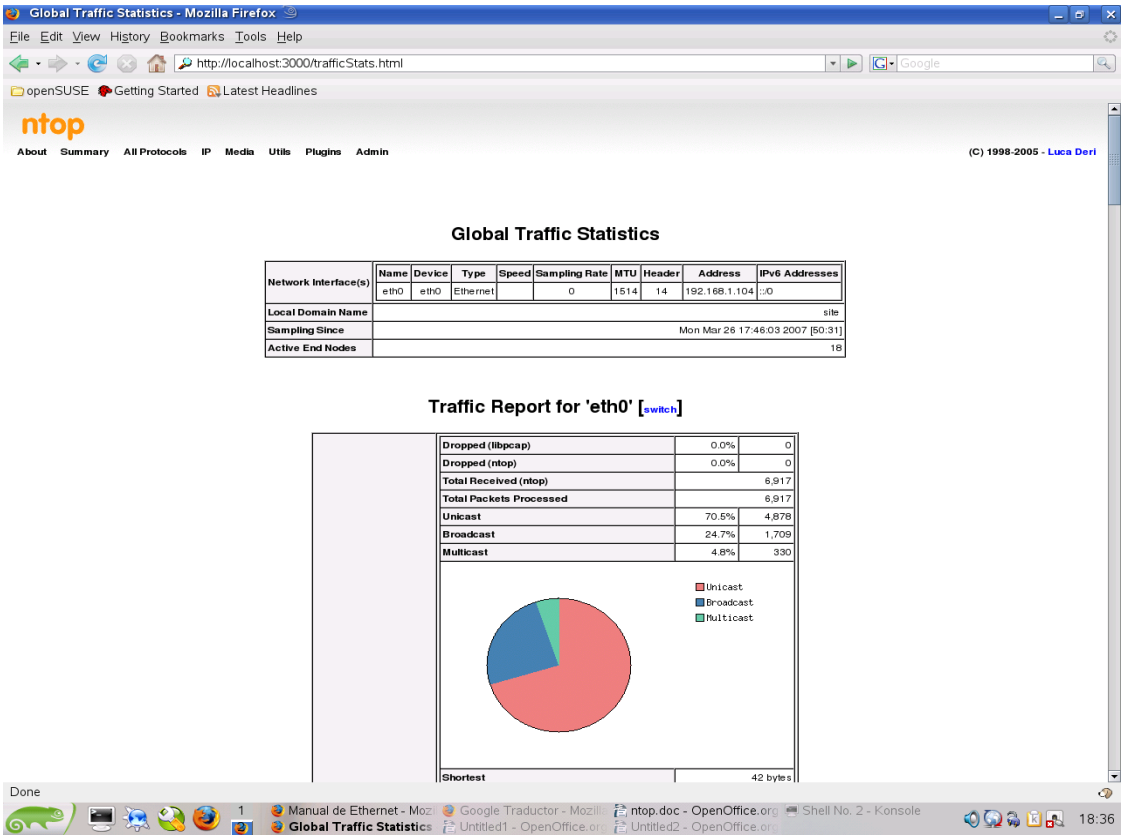


Fig. 39 Reporte de Trafico Global de NTop.

Hosts: Nos genera un reporte de los host de la red con su respectiva dirección IP y dirección MAC. Al dar un clic en cualquiera de las direcciones que aparecen en el primer campo de la tabla generada nos da información especifica del host así como gráficos del trafico generado por hora; estadísticas de paquetes enviados y recibidos por distribución de protocolo y distribución IP.

Network Load: Presenta estadísticas de la carga de la red por medio de gráficas de los 10 minutos pasados, una hora pasada, un día, y una semana.

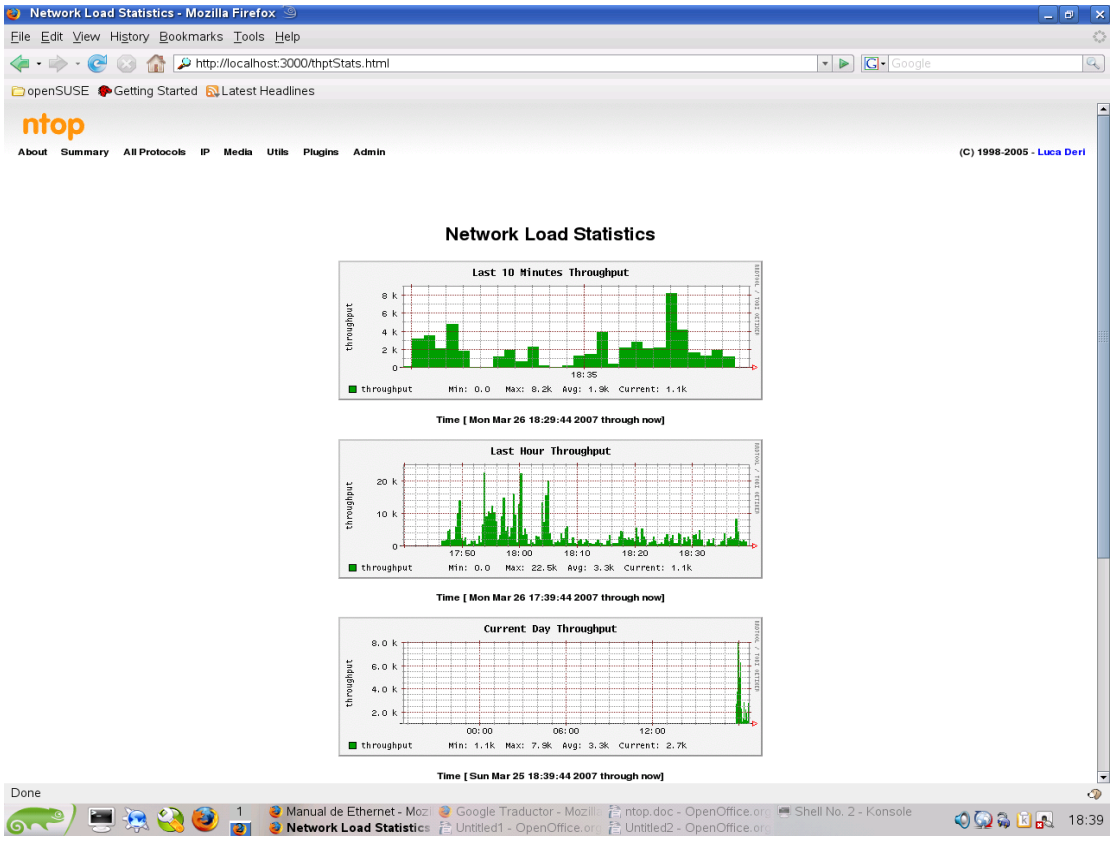


Fig. 40 Gráficas de Estadísticas de carga de red de NTop.

ASN Info: Muestra estadísticas del tráfico de sistemas autónomos.

Network Flows: genera información de los flujos de la red.

– **All Protocol**: Muestra información del trafico y la actividad generada en la red por protocolos.

Traffic: Lista los host con las estadísticas de los datos enviados mas los recibidos por cada protocolo y su porcentaje.

Throughput: Nos da el Rendimiento de Procesamiento de la red; lista los host con el dominio correspondiente, promedio de datos y paquetes; el average de los paquetes en paketes/segundos (Pkts/sec); el average de la cantidad de datos en bps (bits por segundos) ; así como los picos de datos en bps y los picos de paquetes en Pkts/sec.

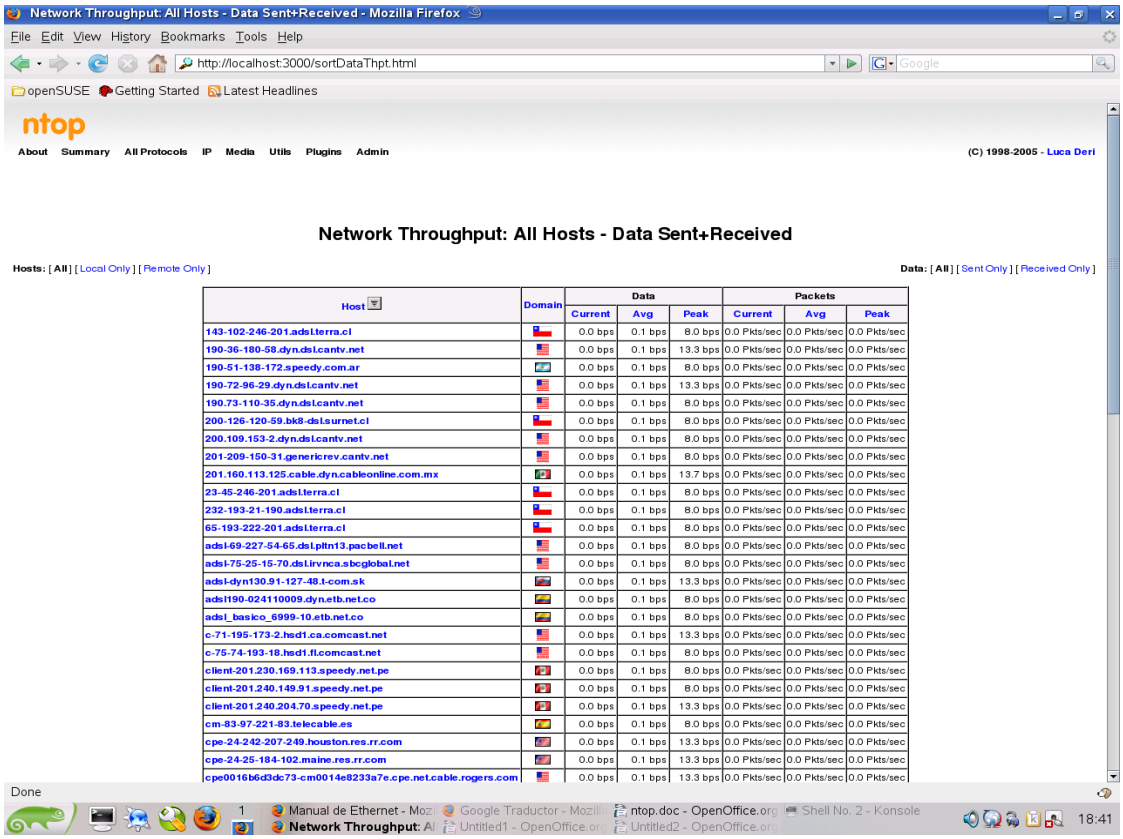


Fig. 41 Estadísticas del Rendimiento de Procesamiento de la Red.

Activity: Genera reporte de todos los host monitorizados con su dominio correspondiente y los porcentajes de los datos enviados mas los recibidos representados con colores que se corresponde con cada hora. Celeste : de 0% a 25% ; Verde: de 25% a 75% y Rojo de 75% a 100%.

- **IP** : Tiene las siguientes opciones:

♠**Summary**: Tiene las siguientes opciones:

Traffic: Presenta un resumen de todos los host con la cantidad en Megabytes y Kilobytes de los datos enviados más los recibidos y un porcentaje de los mismos así como la distribución de los mismos por protocolo.

Multicast: En éste menú encontramos estadísticas de los host que están generando tráfico multicast en una tabla con los números IP , su dominio , número de paquetes enviados, recibidos ; así como la cantidad de datos enviados y recibidos.

Internet Domain: Nos presenta estadísticas de todos los dominios en forma de tabla con los nombres de host, en dos distribuciones básicas, por TCP/IP (TCP y UDP) e ICMP (Ipv4 e IPv6) nos da las estadísticas de los paquetes enviados y recibidos por protocolo.

Distribution: Muestra un gráfico de pastel según la distribución de protocolos; gráficos de barra para el tráfico local por protocolo; gráfico de barras para el tráfico generado de host remotos al local por protocolo; gráfico de barra para el tráfico remoto, por protocolo; y un gráfico de barra para el tráfico generado del host local a los remotos.

Traffic Direction: Tiene las siguientes opciones:

Local to Local: Genera reporte con los números IP de la red local que han enviado peticiones a otros host de la red local. Nos da estadísticas de la cantidad y porcentaje de datos enviados y recibidos así como un total de tráfico.

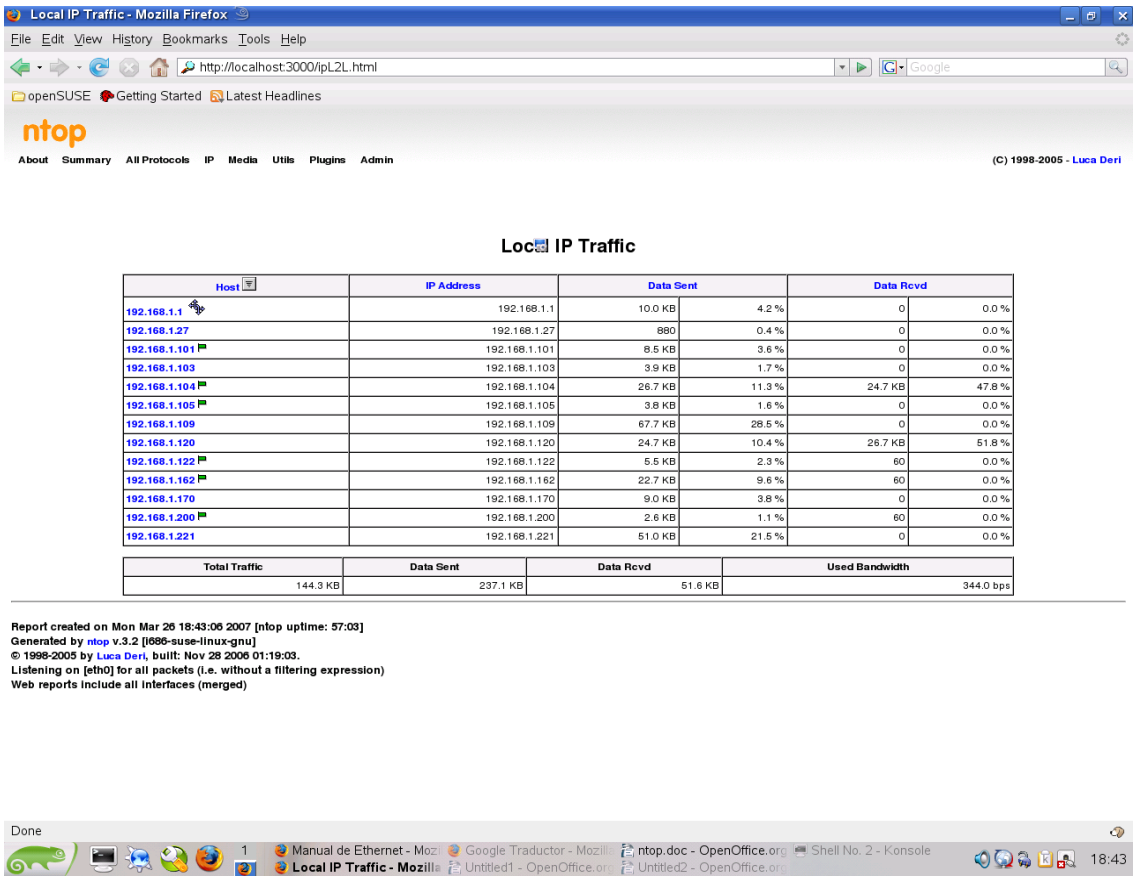


Fig. 42 Estadísticas de la cantidad y porcentaje de datos enviados y recibidos en la red local.

Local to Remote: Nos da un reporte en forma de tabla con los números IP de los host de la red local así como por números IP de los host remotos a los que han accedido estos, allí podremos ver la cantidad y porcentaje de datos enviados como los recibidos.

Remote to Local: Si elegimos esta opción podremos ver un reporte del tráfico generado de los host remotos al localhost, como cantidad y porcentaje de datos enviados y recibidos.

♠**Local**: Presenta las siguientes opciones de menú:

Ports used: Nos presenta una tabla con los puertos y servicios utilizados, con la respectiva indicación del numero IP del cliente y de servidor.

Active TCP/UDP Session: Nos presenta el nombre de los servidores y sus números IP en los que tenemos una sesión activa (accediendo a través de internet) con TCP/UDP.

Host Fingerprint: Genera resumen de los Sistemas Operativos que tienen instalados los host de la red local que se están monitorizando. también podremos saber cuántos tienen Windows y que versión ; así como cuántos tienen Linux.

Local Hosts Characterization: Genera una tabla completa y detallada con todas las características de la maquina servidor.

Network Traffic Map: como lo indica textualmente obtendrá un mapa del tráfico de la red local monitorizada.

- **Media**: En esta opción nos dará un detalle en cuanto a la actividad, los host, el tráfico, las sesiones abiertas de los ordenadores que utilizan las protocolos SCSI (Small Computer System Interface) y Fibre Channel (Canal de Fibra).

- **Utils**: El ntop puede descargar datos del tráfico en los varios formatos (Ej. texto, Perl, php) de modo que los programas externos puedan utilizar estos datos para la transformación posterior, tal como almacenaje en una base de datos.

- **Plugins**: En esta opción podemos ver las estadísticas de los últimos host monitorizados (Host Last Seen) en éste tenemos la opción de desactivar y ver las descripciones de el último host monitorizado; así como desactivar , ver las descripciones del flujo de la red, de la Base de Datos Round Robin Database.

Admin: Sirve para poder cambiar la interfaz de red, crear filtros, y un mantenimiento de usuarios.

3.5 BigSister.

3.5.1 Descripción.

BigSister es un sistema de monitorización que supervisa la red y los host unidos a ésta. Examina los agentes para verificar que servicios están corriendo. BigSister alerta al administrador en caso de problemas.

El nombre BigSister está relacionado con BigBrother, software de monitorización de red de Quest Software, Bigsister es compatible con BigBrother, pero es lanzado bajo licencia GNU-GLP, en contraste a los productos Quest't .La diferencia entre ellos es con respecto a la flexibilidad y la licencia.

BigSister esta compuesto de tres partes.

bigSister: Paquete base que necesita ser instalado si se va a instalar ya sea un Servidor o Agente BigSister.

bigsister-servidor: Paquete adicional para la configuración (bb-display.cfg), al ser instalado la computadora se comportará como un servidor. Este archivo de configuración debe estar presente solamente en el servidor, si no se producirán errores.
bigsister-agente: Paquete adicional de configuración (uxmon-net), al instalarlo la máquina se comportará como un agente. Si se desea que el servidor supervise sus propios recursos, debe instalar también este paquete en el servidor.

Una red supervisada con BigSister, se compone de un Servidor BigSister y varios nodos de agentes en el que corre un demonio llamado uxmon. El servidor es responsable del almacenamiento, procesamiento y visualización de los datos recogidos o recibidos de los sistemas monitorizados. Un BigSister Servidor actually corre BigSister Server **(bbd)**, BigSister Monitor **(bsmon)** y Apache **httpd**. **Bdb** es responsable de la comunicación con

los agentes, incluso verifica si los agentes permiten la realización de ciertas operaciones. **Bbd** entonces pasa toda la información que consigue de los agentes al **bsmon**. Bsmon procesa ésta información para ser mostrada.

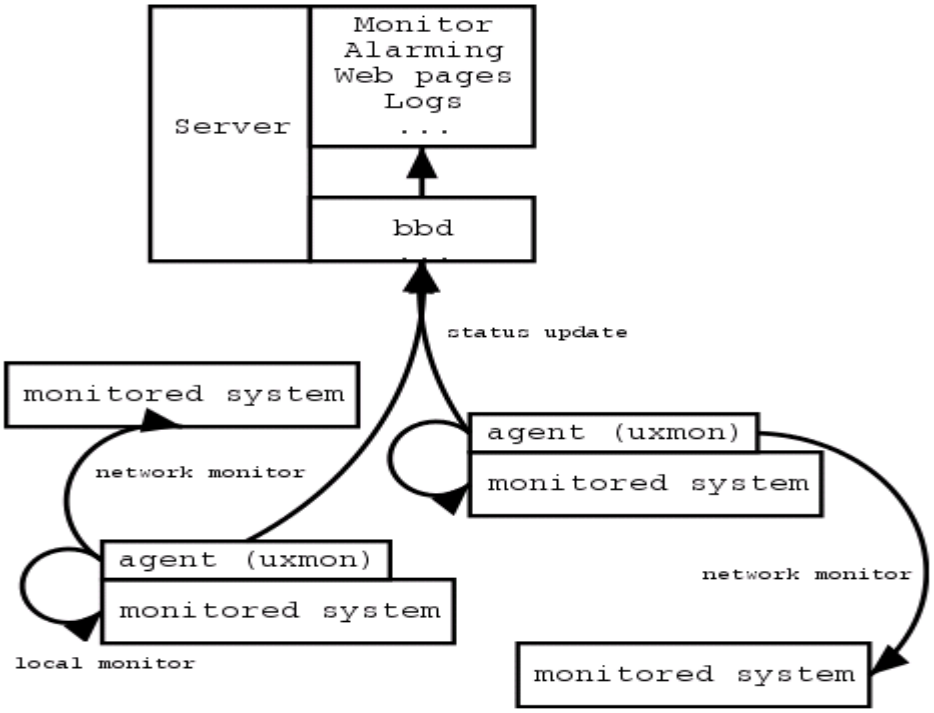


Fig. 43 Componentes de Bigsister.

Cada sistema o servicio es supervisado por un agente de BigSister (uxmon-net). Algunos de los healthcheks (chequeos) son realizados en el sistema en el que está instalado el agente, mientras que otros se hacen vía red. Los sistemas con los agentes instalados recogen los datos específicos de su propio estado, y lo envían al servidor de BigSister para su transformación. Por defecto ésto es realizado por medio de una conexión **tcp** al puerto **1984** en la máquina del servidor. Aquí es donde el **bbd** espera a escuchar datos de sus agentes y los remite al **bsmon**. Hay que tener presente que se tiene que abrir el

puerto 1984 en el cortafuego, en el caso de que los agentes y el servidor estén separados por un cortafuego.

BigSister utiliza tres técnicas de monitorización de sistema:

- El agente está instalado en el host monitor (monitored) que utiliza el puerto TCP para transmitir mensajes del estado al servidor.
- Cada agente instalado puede supervisa remotamente los servicios de red de otras plataformas (por ejemplo SMTP, el NTP, HTTP…) y divulgar el estado al servidor de BigSister.
- BigSister también soporta el estándar SNMP (Simple Network Management Protocol).

3.5.2 Instalación y configuración de un servidor BigSister.

Los paquetes necesarios para la instalación de BigSister son:

- bigsister-1.02-4.noarch.rpm
- bigsister-agent-1.02-4.noarch.rpm
- bigsister-server-1.02-4.noarch.rpm
- rrdtool-perl-1.0.41-2.i586.rpm

Crear un grupo llamado bigsis, y un usuario llamado bigsister:

#groupadd bigsis

#useradd bigsister -d /var/lib/bigsister -g bigsis -m

Instalar los archivos rpm:

#rpm -ivh rrdtool-perl-1.0.41-2.i586.rpm

#rpm -ivh bigsister-1.02-4.noarch.rpm

#rpm -ivh bigsister-server-1.02-4.noarch.rpm

#rpm -ivh bigsister-agent-1.02-4.noarch.rpm

Agregar al archivo de configuración de apache2 (httpd.conf), las siguientes líneas

ScriptAlias /bigsis/cgi /usr/share/bigsister/cgi

```
Alias /bigsis /var/lib/bigsister/www
<Directory /var/lib/bigsister/www>
Order allow,deny
Allow from all
Options +FollowSymLinks
</Directory>

<Directory /usr/share/bigsister/cgi>
Order allow,deny
Allow from all
Options -FollowSymLinks +ExecCGI

<IfModule mod_perl.c>
   <FilesMatch "\.mpl$">
   SetHandler perl-script
   PerlHandler Apache::Registry
   PerlSendHeader On
   PerlSetEnv PERL5LIB /usr/share/bigsister/bin
   </FilesMatch>
</IfModule>

</Directory>
```

Configuración del agente

Aquí solo se instalarán los paquetes:

rrdtool-perl-1.0.41-2.i586.rpm

bigsister-1.02-4.noarch.rpm

bigsister-agent-1.02-4.noarch.rpm

Ahora se procederá a configurar y a activar los agentes. Primero se debe configurar todos los nodos del agente para señalar a tu servidor BigSister, editando el archivo uxmon-net.

Localizar la línea **localhost bsdisplay**, sustituir el localhost por el nombre (o el IP address) del sistema que recibe tu servidor de BigSister y comenzar el agente. También se cambiará el nombre de la comunidad utilizada.

Ejemplo del archivo de configuración uxmon-net

##

KEYWORD Default settings Apply To TEST

DEFAULT community=**prueba** frequency=5 perf=5 ALL

DEFAULT version=1 proto=udp rpc

DEFAULT proto=udp ping

Information about defined systems to monitor using DESCR command.

KEYWORD SYSTEM FEATURES Apply To HOST

DESCR features=unix,linux localhost

DESCR features=unix,sysv,solaris someotherhost

Run the following tests.

Note: host1(host2) is reported under host2

Note: host can be an IP address

Report Host Health Test List

localhost load memory network cpuload

localhost disk

localhost syslog

localhost proc=**snmpd** procs proc=sshd procs

localhost users

EDIT THIS, replace localhost by the name or IP address of your Big Sister server

BigSis Server bsdisplay /options

192.168.1.125 bsdisplay

include file for specific hosts, do not name it uxmon-net.* as a new
process is started for every file matching that pattern
include include_checks.$HOST
##
Para arrancar el agente BigSister se deberá ejecutar el siguiente comando

#/etc/init.d/bigsister start

El servidor deberá inmediatamente empezar a escuchar por el puerto (TCP 1984), y creará primero una página Web en el directorio /var/lib/bigsister/www.

Configuración del servidor BigSister

Aquí se instalan todos los paquetes.

Se deberá modificar el archivo uxmon-net y agregar las siguientes líneas:

```
##########################################################################
# KEYWORD     Default settings                  Apply To TEST
DEFAULT           community=prueba frequency=5 perf=5     ALL
DEFAULT           version=1 proto=udp                rpc
DEFAULT           proto=udp                          ping
# KEYWORD     SYSTEM FEATURES                 Apply To HOST
DESCR             features=unix,linux                localhost
DESCR             features=unix,linux                192.168.1.120
#DESCR            features=unix,sysv,solaris         someotherhost

localhost      load memory network cpuload
localhost      disk
localhost      syslog
```

localhost proc=**snmpd** procs proc=sshd procs
localhost users

192.168.1.120 load memory network cpuload
192.168.1.120 disk
192.168.1.120 syslog
192.168.1.120 proc=snmpd procs proc=sshd procs
192.168.1.120 users
##

Iniciamos el servicio de bigsister
#/etc/init.d/bigsister start

Hay que tener en cuenta que también tiene que estar corriendo Apache y snmp
#/etc/init.d/apache2 start
#/etc/init.d/snmpd start

Por último acceder a la siguiente dirección:
http://localhost/bigsis/

Mostrará la siguiente página:

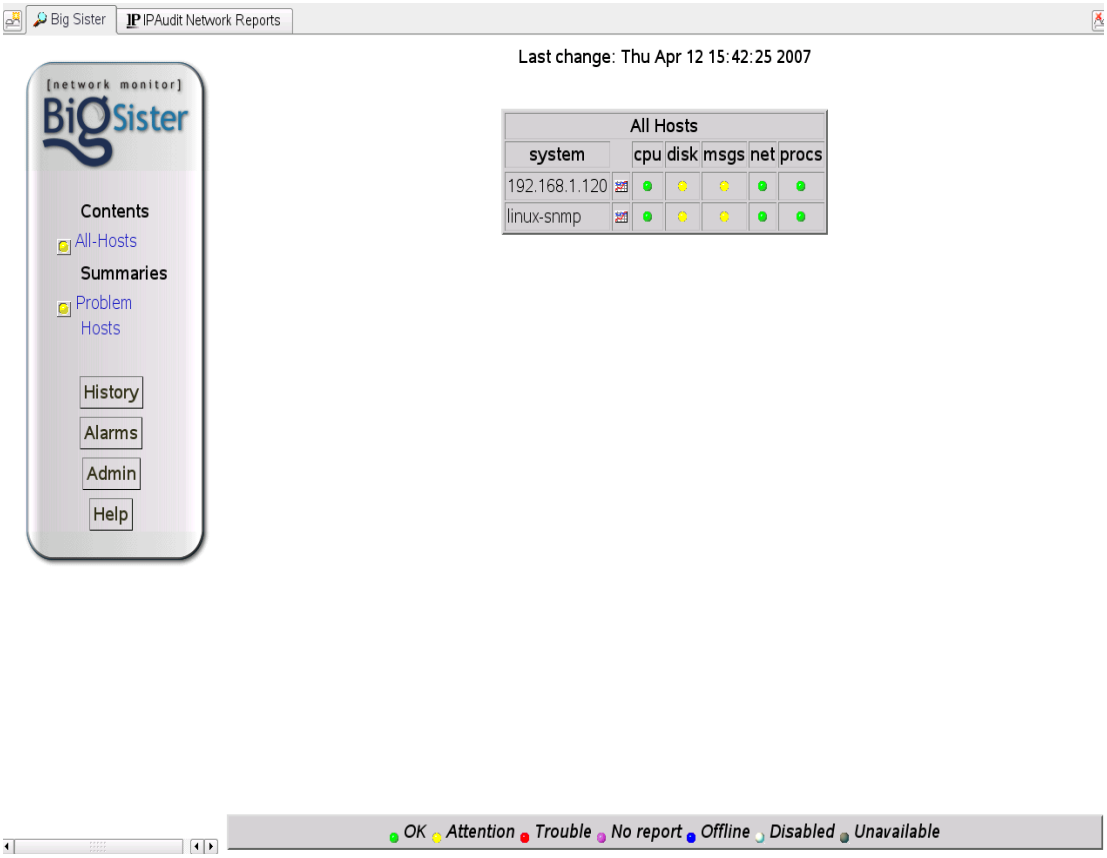


Fig. 44 Ventana Principal de BigSister.

3.6 IPAUDIT.

3.6.1 Introducción a IPAudit.

Las distintas herramientas de monitorización y control del tráfico de red utilizan un modo de trabajo de las tarjetas de red denominado "**modo promiscuo**". El modo de funcionamiento normal de una tarjeta de red consiste en que al recibir un paquete de información, si la dirección de destino no es la que hay configurada en la tarjeta de red lo ignora. En el modo promiscuo toda la información que llega a la tarjeta de red es accesible.

IPAudit es una herramienta práctica que permitirá que analices todos los paquetes que entran y salen de tu red. Escucha un dispositivo de la red en modo promiscuo, apenas como un sensor de las identificaciones, y proporciona los detalles en los host, los puertos, y los protocolos. Puede ser utilizado para supervisar la anchura de banda, pares de la conexión, para detectar compromisos, para descubrir botnets (normalmente es un gusano que corre en un servidor infectado con la capacidad de infectar a otros servidores), y para ver quienes exploran tu red.

Hemos escogido el uso de la versión IPaudit-Web debido principalmente a que la versión "normal" de IPaudit es en modo texto (línea de comandos) lo que dificulta la interpretación de los datos. Por otro lado, la versión Web nos permite consultar de una forma visual y sencilla los diferentes datos y gráficas generadas por el tráfico de red. La versión Web utiliza procesos CRON que se lanzan cada media hora y que durante treinta minutos se encargan de registrar todo el tráfico generado en la red. De ésta forma tenemos que cada 30 minutos se envía una señal de finalización controlada al proceso anterior (que escribe en el fichero de log correspondiente los datos del tráfico pendientes) y se inicia un nuevo proceso de captura de datos.

3.6.2 Instalación y configuración.

IPAudit es una aplicación basada en Perl, escrito por Juan Rifkin en la universidad de Connecticut. Puede ser descargado de Sourceforge **(http://ipaudit.sourceforge.net)** y se licencia bajo el GNU GPL(Licencia publica general de software libre).

IPAudit es una línea de comando, herramienta que utiliza la biblioteca del libpcap para escuchar el tráfico y para generar datos. El paquete del IPAudit-Web incluye el IPAudit binario además del interfaz Web que crea los informes basados en los datos obtenidos. Se recomienda usar el paquete del Web, pues te da un interfaz gráfico completo con mapas del tráfico y una característica de búsqueda.

- Necesitarás tener a disposición de un sistema de Linux o del Unix con la biblioteca del **libpcap** instalada para su acceso a los dispositivos de red.

- Además necesitarás el compilador del lenguaje **Perl** debido a que IPAudit-Web ejecuta varios scripts escrito en este lenguaje.

- Es necesario un servidor WWW (**Apache)** que este bien configurado para que permita al usuario la visualización de paginas HTML (**http://servidor/~ipaudit**) y la ejecución de scripts (**http://servidor/~ipaudit/cgi-bin**) desde su directorio de trabajo (/home/ipaudit/public_html usualmente).

- Necesita la utilidad **GNUplot** para la generación de los gráficos.

- Y un módulo del Perl llamado **"Time:: ParseDate"** que se encuentra en el paquete "libtime-modules-perl_2003.1126.orig.tar.gz".

Referir a la documentación de tu distribución de Linux para más información sobre cómo instalar estos paquetes (aquí está un ejemplo: En Debian Linux, ejecute el comando "apt-

get install libtime-modules-Perl" para instalar el modulo Time:: ParseDate). Una vez que hayas instalado estos paquetes, estás listo para comenzar a instalar IPAudit:

Paso 1

Desde la consola, entrar como root al sistema

$**su root**

Password:

Crea un usuario llamado "ipaudit"; y un directorio casero (típicamente /home/ipaudit).

#groupadd ipaudit

#useradd –g ipaudit –d /home/ipaudit –m ipaudit

Ahora cambiarse al usuario (creado recientemente) del "ipaudit".

#su ipaudit

Paso 2

Descargar y desempaquetar el IPAudit-Web:

$tar zxvf ipaudit-web-1.OBETA9.tar.gz

Paso 3

Cambiar al directorio de compilación:

$cd ipaudit-web-1.OBETA9/compile

Paso 4

Ejecutar el script de configuración y el comando make:

$ ./configure

$ make

Paso 5

Se convierte a root y ejecute el comando make-install:

$ su root

Password:

 # make install

 # make install-cron

#exit (salir del root y entrar como usuario ipaudit otra vez)

$

Paso 6

Ahora necesitarás corregir /home/ipaudit/ipaudit-web.conf

#

LOCALRANGE=127.0.0

#

#

INTERFACE=eth0

#

Cambiar la variable de LOCALRANGE a tu subnet local en el interior de tu red. También estar seguro de fijar la variable INTERFACE al interfaz que utilizas para capturar el tráfico deseado en tu red.

Paso 7

Agregar las líneas siguientes a tu archivo de Apache httpd.conf si no existen ya:

<Directory/home/*/public_html>

AllowOverride All

Options MultiViews Indexes Includes FollowSymLinks

Order allow,deny

Allow from all

</Directory>

<Directory /home/*/public_html/cgi-bin>

Options +ExecCGI -Includes -Indexes

SetHandler cgi-script

</Directory>

Observar que tu servidor de Apache puede contener ya la configuración similar a lo dicho anteriormente para el directorio de "/home/*/public_html". Si no planeas utilizar el módulo de Userdir para cualquier cosa con excepción de IPAudit, se sugiere que comentes la configuración original y la sustituyas por la configuración de arriba.

Tu servidor de Apache necesitará apoyar **SUEXEC** (httpd-suexec-2.0.53-3.3.i386.rpm), **Mod_Perl** (apache2-mod_perl), y **Mod_Userdir**. Una vez que hayas modificado la configuración de Apache hacer un **restart** a tu servidor de Apache (**/etc/init.d/apache2 restart**). Para más detalles en la instalación del IPAudit-Web, referir al archivo de la INSTALACIÓN situado en el directorio de la instalación de ese paquete. Además contiene más información sobre el módulo requerido de Perl Time:: ParseDate, SUEXEC, y contraseña que protege tu instalación de IPADUIT-Web.

Paso 8

Comprobar tu instalación

Abrir un Web browser e ir:

http://localhost/~ipaudit

Si tu instalación era acertada debes ahora ver una pantalla como la que está demostrada abajo.

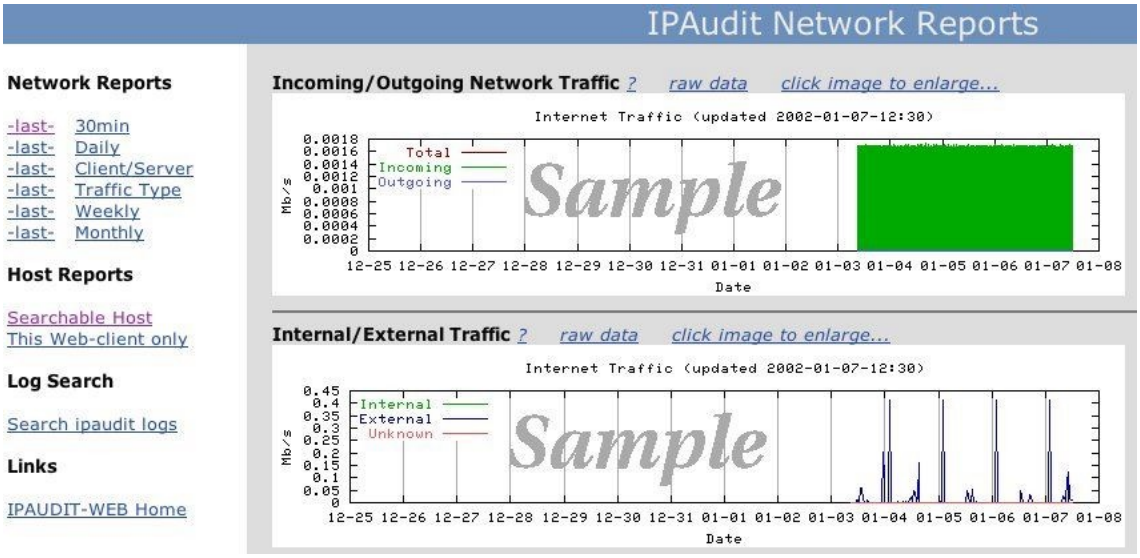


Fig. 45 Interfaz de Funcionamiento del IPAudit-Web por primera vez.

Debes asegurarte que el tiempo del servidor en el que funciona IPAudit sea correcto y que se mantenga actualizado usando **NTP** (Network Time Protocol: es un protocolo de comunicaciones que permite sincronizar el reloj de un ordenador que esté conectado a la red con un servidor central de tiempo. Con ello lograremos una exactitud del orden de milisegundos en una red local). Sin tiempo exacto, IPAudit conseguirá confusiones, si el tiempo en los paquetes diferencia grandemente del tiempo del sistema.

Después de la primera marca de media hora, IPAudit comenzará a representar todo tu tráfico gráficamente y a generar algunos informes.

Los gráficos conseguirán ser más interesantes según pase el tiempo y mientras IPAudit vea más tráfico. Un “punto” en el gráfico denota típicamente una indicación de un problema, tal como un host que envía un ataque del DOS (negación del servicio).

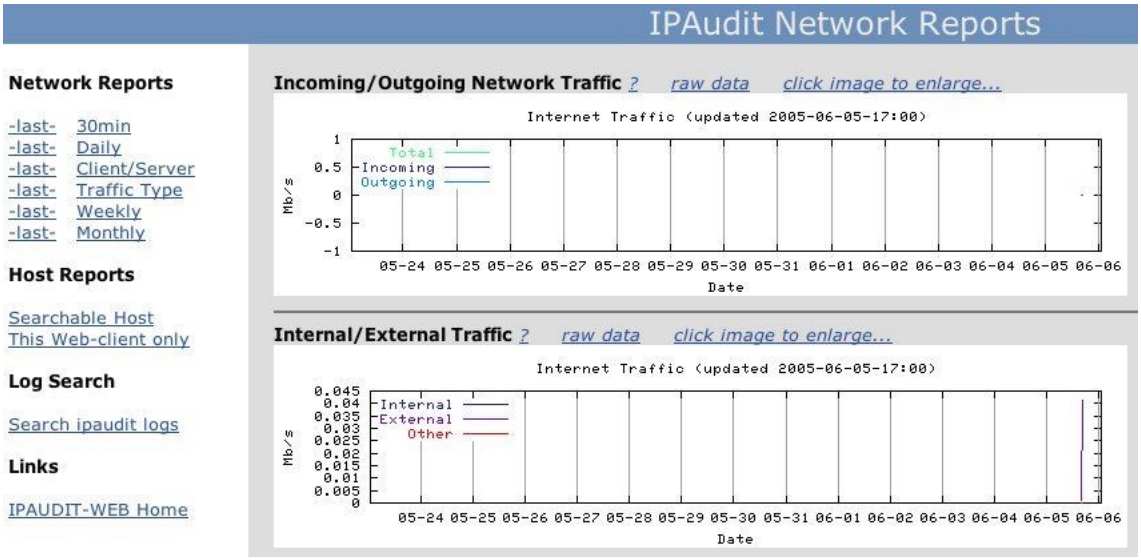


Fig. 46 El primer gráfico aparece después de 30 minutos.

3.6.3 Uso de IpAudit..

Los informes de la red de IPAudit son útiles por muchas razones expuestas al inicio del informe. El treinta-minuto y los informes diarios son exactamente igual, excepto por supuesto para el marco de tiempo (timeframe). Tecleando sobre el enlace de la etiqueta "-last- "al lado del enlace "30min" verás el informe para los 30 minutos pasados.

En la parte de arriba de la pantalla se pueden observar las estadísticas generales de la red, lo cual es bueno si estás intentando guardar presentaciones de la utilización total del ancho de banda.

General Stats		Incoming/Outgoing Traffic (bytes)		Internal/External Traffic (bytes)		Local Hosts		Remote Hosts	
Connections	316	**Incoming**	231,126	**Internal**	112,001	**Probed**	3	**Probed**	20
Packets	2,457	**Outgoing**	153,343	**External**	8,291	**Responding**	9	**Responding**	38
Bytes	504,761	**Total**	384,469	**Other**	0	**Total**	9	**Total**	41

Ésta es seguida por el reporte de los **host locales más ocupados**, lo cual es una buena manera de vigilar quién está transfiriendo la mayoría de los datos dentro y fuera de tu red (organización).

Busiest Local Hosts				
IP	Host Name	Incoming (bytes)	Outgoing (bytes)	Total (bytes)
192.168.001.237	-	75,353,878	67,232,083	142,585,961
192.168.001.045	bud.mycompany.com	9,093,694	1,102,415	10,196,109
192.168.001.211	-	7,269,574	1,275,000	8,544,574
192.168.001.022	hanzo.mycompany.com	2,835,412	84,678	2,920,090
192.168.001.223	-	711,662	73,614	785,276
192.168.001.015	ns.mycompany.com	424,349	201,070	625,419
192.168.001.010	ns2.mycompany.com	168,262	93,613	261,875
192.168.001.232	-	176,063	46,217	222,280
192.168.001.238	-	100,123	42,951	143,074
Elapsed time is 8 seconds.				

Fig. 48 Hosts más ocupados en la red local.

Los servidores, tales como servidores de correo SMTP, estarán típicamente de primero en esta lista (además de los hosts P2P (peer-to-peer): se refiere a una red que no tiene clientes y servidores fijos, sino una serie de nodos que se comportan a la vez como clientes y como servidores de los demás nodos de la red, claro si permites ese uso). En un cierto plazo de tiempo desarrollarás una línea de fondo de los hosts más ocupados. Cuando compruebes diario los reportes, puede suceder que notes a un nuevo host que ocupara las celdas superiores de los host más ocupado, ésta sería una muy buena causa para que investigues.

El informe de los **hosts remotos mas ocupados** dice a y desde quien en el Internet esta transfiriendo la mayoría de los datos.

Busiest Remote Hosts				
IP	Host Name	Incoming (bytes)	Outgoing (bytes)	Total (bytes)
207.025.253.040	dispsd-40-www3.boulder.ibm.com	14,019,752	289,638	14,309,390
069.170.243.096	69-170-243-96.atlsfl.adelphia.net	5,566,213	6,770,587	12,336,800
068.015.034.115	entropy.tmok.com	8,100,317	329,706	8,430,023
024.080.174.189	S0106001217040deb.vc.shawcable.net	1,861,523	5,086,032	6,947,555
204.152.191.007	mirrors1.kernel.org	6,587,080	170,039	6,757,119
062.058.237.015	dslam15-237-58-62.adsl.versatel.nl	6,372,971	193,395	6,566,366
024.068.097.037	S0106000625633845.gv.shawcable.net	6,144,556	208,611	6,353,167
070.028.234.144	CPE000ea640812c-CM0012c9db37da.cpe.net.cable.rogers.com	3,434,800	2,782,488	6,217,288

Fig. 49 Hosts remotos más ocupados.

Éstos típicamente tienden a ser direcciones IP de la actualización de Windows, y otros sitios Web populares como Google o Yahoo. Si uno de los sitios enumerados da resoluciones algo desconocedor como www.evil.com, debe ser causa para alarmar.

El informe siguiente es, los **hosts entrantes posibles de la exploración**, que muestra las direcciones IP de los hosts que están conectados, o han intentado conectar, con la dirección IP más utilizada sobre la subred local. Éste informe es útil para considerar quién está explorando tu red, y hacia qué puertos están explorando.

Possible Incoming Scan Hosts		
IP	Host Name	Local Hosts Contacted
218.066.104.131	-	77,605
061.129.051.046	-	73,451
061.134.049.034	-	60,119
211.169.240.201	-	40,999
142.179.217.049	s142-179-217-49.ab.hsia.telus.net	40,786
210.222.009.101	-	40,726
203.146.102.212	-	40,713

Fig. 50 Direcciones remotas IP que exploran tu red.

Es bueno comprobar esta tabla diaria al supervisar una red. Es útil tomar los puertos más comunes que están siendo explorados en la red para ser investigados.

Los sitios Web siguientes son útiles al determinarse qué aplicaciones corresponden a los puertos que los host entrantes están explorando:

- SANS (**http://isc.sans.org**)
- La base de datos portuaria oficial de las asignaciones (**http://www.iana.org**)
- Google (**http://www.google.com**)

La actividad de la exploración de Puerto es algunas veces debido a una herramienta nueva para la exploración de red que está siendo lanzada (como scannssh), o un nuevo virus o un gusano que está circulando. Teniendo esta información, es bueno advertir al usuario sobre la amenaza.

La exploración de los posibles hosts salientes se enumera después. Mientras que la exploración de los posibles hosts entrantes se puede utilizar para las medidas proactivas, el informe siguiente de los hosts salientes es más útil para las medidas reactivas.

Possible Outgoing Scan Hosts		
IP	Host Name	Remote Hosts Contacted
192.168.001.045	bud.mycompany.com	6,634
192.168.001.015	ns.mycompany.com	19
192.168.001.010	ns2.mycompany.com	14
192.168.001.237	-	12
192.168.001.223	-	2
192.168.001.211	-	2
Elapsed time is 8 seconds.		

Fig. 51 La exploración de los hosts salientes es útil para descubrir las máquinas Troyanas.

Si encuentro los hosts que están en el interior de la red explorando hacia fuera, es generalmente una indicación que una máquina se ha comprometido con un gusano o un virus, y en algunos casos un atacante real ha tomado el control del host y lo está utilizando para explorar otras máquinas. Cuando comienzas a comprobar los informes sobre una base regular podrás desarrollar una línea de fondo y sabrás qué es lo normal en tu red en lo que respecta al número de hosts contactados en un día dado.

La tabla de los **pares de host más ocupados** es el informe final. Enumera qué hosts hicieron las más grandes transferencias entre ellos. Es una buena idea hacer una observación a esta lista y asegurarse que las transferencias entre lo hosts sean normales. El comportamiento normal sería alguien que descarga una imagen de la ISO de Linux, mientras que comportamiento menos normal podría ser alguien que descargaba medios

Busiest Host Pairs

Local IP	Local Host Name	Remote IP	Remote Host Name	Incoming (bytes)	Outgoing (bytes)	Total (bytes)
192.168.001.001	-	239.255.255.250	-	0	2,913,834	2,913,834
192.168.001.120	-	192.168.151.190	-	320,862	0	320,862
192.168.001.021	-	192.168.151.190	-	191,520	0	191,520
192.168.001.001	-	255.255.255.255	-	0	1,710	1,710

Elapsed time is 0 seconds.

pirateados de un host ya comprometido.

Fig. 52 Pares de Host más ocupados.

Yendo de nuevo a la página principal de IPAudit, notarás más informes que puedas explorar.

Informe cada 30 minutos: En este tipo de informe se muestra el total de bytes enviados y recibidos en nuestra red durante la media hora analizada. Asimismo se muestran las veinte direcciones IP locales y remotas con más tráfico generado, lo que nos permite examinar más a fondo las comunicaciones que nos interesen.

Informes diarios (daily): Estos informes presentan el resumen de la acumulación de todos los informes de media hora realizados durante el día. Visualiza el total de bytes enviados y recibidos en nuestra red así como las veinte direcciones IP locales y remotas que más tráfico han generado o recibido.

El informe del **Client/Server** puede ser útil para supervisar quién está utilizando los siguientes servicios de tu red:

- Servidores **HTTP.**
- Servidores de **Correo**.
- Servidores **SSH.**
- Servidores **Telnet.**
- Servidores **HTTPS.**

Compruebo típicamente estos informes sobre una base semanal para obtener una idea de quién está utilizando los servicios del servidor en la red. Una bandera roja sería una estación de trabajo del usuario que termina en la lista de los primeros diez servidores SMTP. Esto podría indicar que el host ha sido infectado y está usándose para distribuir el SPAM(es el hecho de enviar correos electrónicos (habitualmente de tipo comercial) no solicitados y en cantidades masivas). El listado de los servidores HTTP es útil para ver no sólo quién está utilizando el servidor Web legítimo en su red, también puede ser una indicación de cualquier persona que hace un túnel hacia otros protocolos con el HTTP y que está utilizando por encima de los puertos 80 o 443 del TCP. Puesto que IPAudit solamente observa el IP y la información de la capa de transporte, no distinguirá entre el tráfico real del HTTP y el tráfico generado por túneles (que realmente puede ser bueno en éste caso).

El informe tipo de tráfico (type traffic): En éste informe se realiza una agregación de todo el tráfico generado diariamente y se clasifica según el protocolo (TCP, UDP, ICMP, NetBios, Telnet, FTP…) al que pertenezca.

Informes semanales (weekly): Estos informes reflejan la suma total del tráfico generado por la red durante la semana y presentan las 25 direcciones IP que más tráfico han generado o recibido.

Informes mensuales (montly): Este informe es exactamente igual que el informe semanal, pero mostrando el tráfico de las 25 direcciones IP con más tráfico de red durante el mes.

Los informes **traffic type**, **weekly** y **monthly** todos contienen información sumaria sobre tu red. Deben ser comprobados semanalmente para conseguir una descripción de qué protocolos de red están en uso, y qué hosts transmiten y reciben la mayoría de los datos. Los informes del host contienen mucha de la misma información que los informes semanales y mensuales, excepto sobre una par de host base.

La característica de **búsqueda de un registro** es una manera excelente de encontrar ciertos tipos de tráfico usando múltiples criterios.

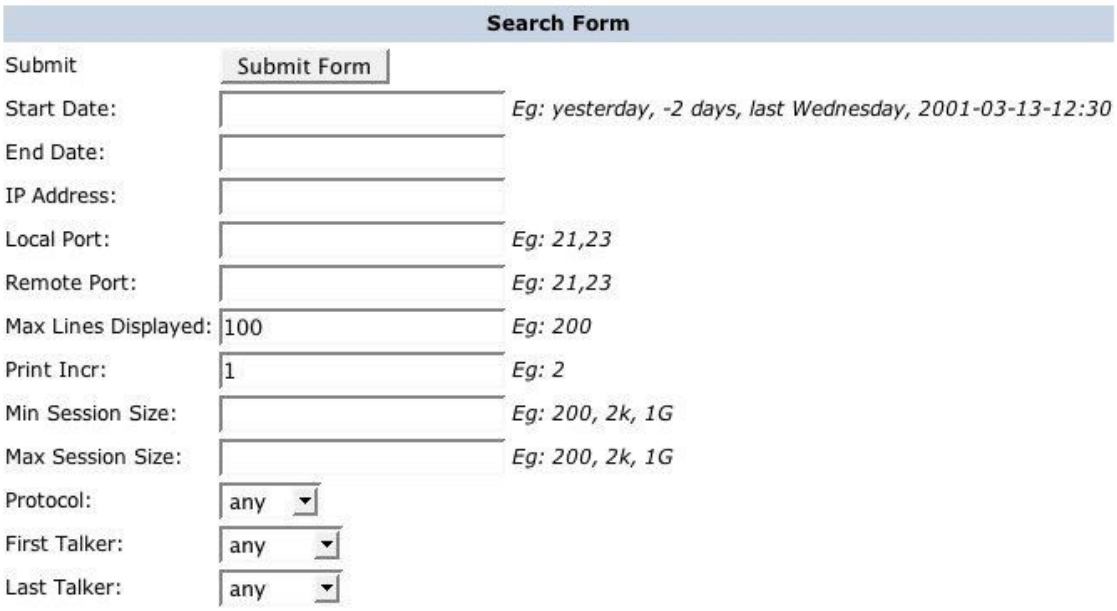


Fig. 53 Buscar los registros de IPAudit.

IPaudit-Web te permite realizar la búsqueda en los ficheros de log por una gran cantidad de campos (tipo de protocolo, dirección IP, puerto de origen, puerto de destino…) lo que permite filtrar el tráfico registrado para analizar únicamente las comunicaciones deseadas.

Puedes ajustar tu pregunta a un período del momento específico (en los campos Start Date-End Date). El IP address puede ser un host de la red local o de la red externa-Internet. El puerto local está concerniente al espacio de dirección local que especificaste en el archivo de la configuración del IPAudit-Web, al igual que el puerto remoto. Los dos campos siguientes, Max Lines Displayed y el Print Incr dice a IPAudit cómo imprimir hacia fuera la pregunta. Es mejor comenzar con un número bajo para la línea exhibida la primera vez que realizas una pregunta, apenas en caso de que hay millares de resultados que podrían tomar una cierta hora. El tamaño de la sesión es un campo particularmente útil cuando se intenta determinar el tipo de tráfico. Algunas veces deseas distinguir entre las transferencias de datos reales y una exploración de puertos. Jugando alrededor con los valores en estos campos puedes realizar justamente eso (por ejemplo, suponer que

deseas saber quién conectó realmente con el servidor de MySQL, no quién lo exploró). La opción protocol permite que elijas entre el TCP, el UDP, y el ICMP. IPAudit intenta no perder de vista el estado indicando quién fue el primero en transmitir cuando se estableció la conexión (en los campos First Talker y Last Talker).

Total, IPAudit tiene muchas características útiles y muchas maneras en las cuales observar su tráfico de red.

Utilizando IPAudit, podemos examinar el tráfico del host víctima. Primero iría al buscador del IPAudit con las características del host, incorporo el marco de tiempo que deseo ver (Star-Date y End-Date), luego la dirección IP (IP Address). Esto produce un informe según:

Local IP	Remote IP	Proto-col	Local Port	Remote Port	Incoming Bytes	Outgoing Bytes	Incoming Packets	Outgoing Packets	First Packet Time	Last Packet Time	First Talker	Last Talker
192.168.1.223	10.1.29.217	tcp	39402	445	0	54	0	1	14:30:01.8879	14:30:01.8879	L	-
192.168.1.223	10.1.28.212	tcp	39402	445	0	54	0	1	14:30:01.8880	14:30:01.8880	L	-
192.168.1.223	10.1.26.192	tcp	39402	445	0	54	0	1	14:30:01.8881	14:30:01.8881	L	-
192.168.1.223	10.1.30.99	tcp	39402	445	0	54	0	1	14:30:01.8881	14:30:01.8881	L	-
192.168.1.223	10.1.22.134	tcp	39402	445	0	54	0	1	14:30:01.8882	14:30:01.8882	L	-
192.168.1.223	10.1.25.139	tcp	39402	445	0	54	0	1	14:30:01.8883	14:30:01.8883	L	-
192.168.1.223	10.1.30.234	tcp	39402	445	0	54	0	1	14:30:01.8883	14:30:01.8883	L	-
192.168.1.223	10.1.17.149	tcp	39402	445	0	54	0	1	14:30:01.8884	14:30:01.8884	L	-
192.168.1.223	10.1.20.46	tcp	39402	445	0	54	0	1	14:30:01.8885	14:30:01.8885	L	-
192.168.1.223	10.1.22.17	tcp	39402	445	0	54	0	1	14:30:01.8885	14:30:01.8885	L	-

Fig. 54 Buscar los resultados para un ciertos timeframe e IP address.

Los datos antes dichos indicaron que el host está escaneando puertos por el puerto 445. Primero, vemos que el mismo puerto origen está utilizado para múltiples conexiones con diversos hosts destino. En comunicaciones normales del TCP, un diferente puerto origen sería utilizado para conectar con hosts diferentes. En segundo lugar, vemos muchos esfuerzos para el puerto 445 de una red de clase B, con pocos datos que son transferidos. También, si miramos la columna etiquetada “First Talker” indica que el host en la red local inició la conexión. La columna del “Last Talker” esta en blanco, diciéndonos que 192.168.1.223 envió los paquetes, pero que no recibió ninguna respuesta. Éstas son todas las señales reveladoras de la exploración de puertos.

Descripción de los gráficos generados por IpAudit.

La Fig. 55 muestra el tráfico IP que se ve fluir por tu conexión(es) de red que esté viajando entre las computadoras locales y remotas (el tráfico observado viajando de local a local o de remoto a remoto se demuestra en el gráfico interno y externo del tráfico). Los paquetes que se envían de máquinas remotas a las locales se trazan como entrante - los paquetes que viajan en la otra dirección son salientes. Las computadoras locales son las que su dirección IP pertenecen al LOCALRANGE según lo especificado en el archivo de la configuración de ipaudit-web.conf.

El tráfico se muestrea sobre 30 intervalos minuciosos.

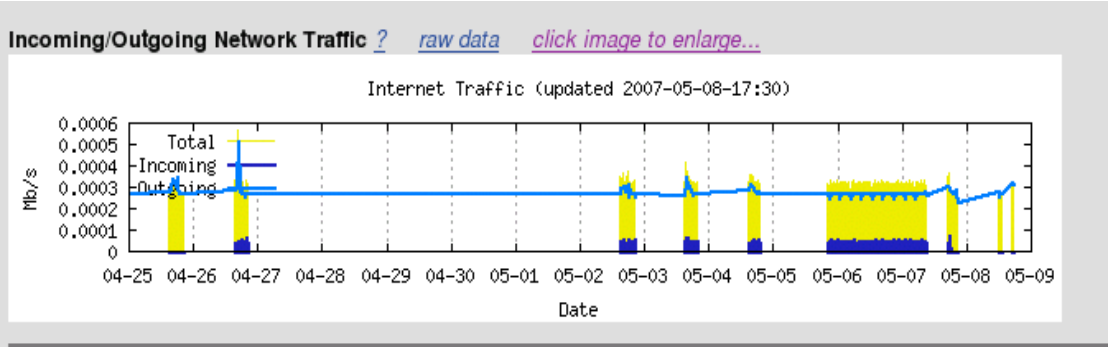


Fig. 55 Gráfico de Tráfico generado por IpAudit.

La Fig. 56 muestra el tráfico IP que se ve fluir por tu conexión de red. Las direcciones locales IP (según lo determinado por la variable de LOCALRANGE en el archivo de ipaudit-web.conf) se consideran internas (local). Todos los otros se consideran externos (remoto). La línea marcada externa muestra el ancho de banda utilizado por los paquetes con origen externo y direcciones de destino. Tales datos se pudieron causar por tráfico del multicast. Si un host local está enviando datos con spoofed la dirección origen externa que puede demostrarse como trafico externo. La línea marcada interna muestra paquetes de rendimiento de procesamiento con ambas direcciones IP internas a tu red.

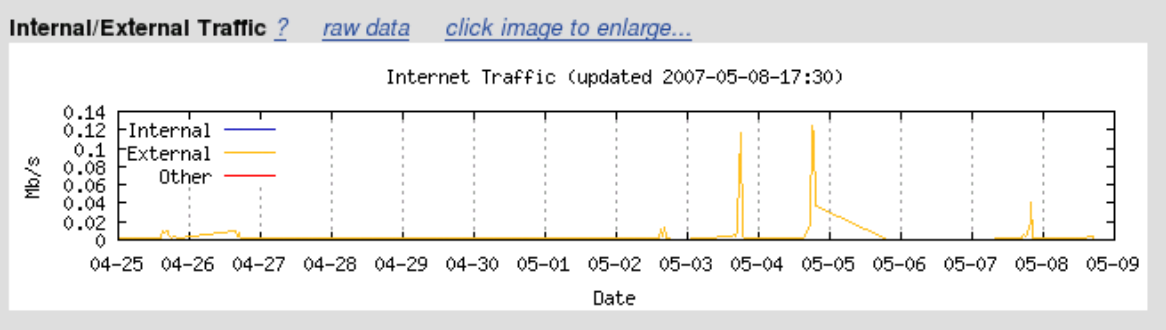


Fig. 56 Trafico Internet.

La Fig. 57 muestra el número de host locales (o las direcciones) en tres grupos - los cuáles han recibido solamente los paquetes, ésos cuáles han enviado solamente los paquetes, y ésos cuáles han enviado y recibidos los paquetes. Host locales (direcciones) que han recibido solamente los paquetes de casi ciertamente el resultado de la exploracion de la red remota. Una gran cantidad de host locales que han enviado solamente los paquetes son inusuales. Puede ser que sea causado un host local que atacaba un host remoto mientras que aleatoriamente spoofing su dirección origen. Las computadoras locales son las que su direccion IP pertenece al LOCALRANGE según lo especificado en el archivo de configuración ipaudit-web.conf. La mayoría de las direcciones envían y reciben los paquetes, éstos son hots implicados en la comunicación normal de la red.

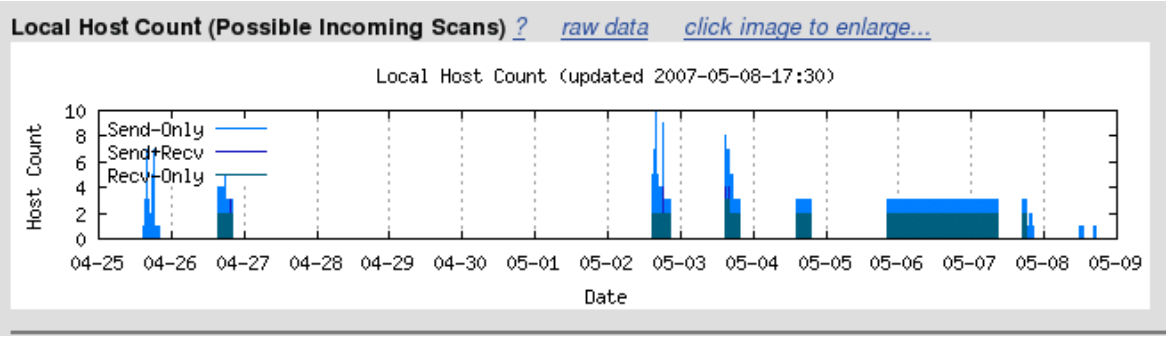


Fig. 57 Número de Host Locales.

La Fig. 58 muestra el número de host remotos (o direcciones) en tres grupos - los que han recibido solamente los paquetes, ésos que han enviado solamente los paquetes, y ésos que han enviado y recibidos los paquetes. Host locales (direcciones) que han recibido solamente los paquetes de casi ciertamente el resultado de la exploracion de la red remota. Una gran cantidad de host locales que han enviado solamente los paquetes son inusuales. Puede ser que sea causado un host local que atacaba un host remoto mientras que aleatoriamente spoofing su dirección origen. Las computadoras locales son las que su direccion IP pertenece al LOCALRANGE según lo especificado en el archivo de configuración ipaudit-web.conf. La mayoría de las direcciones envían y reciben los paquetes, éstos son hots implicados en la comunicación normal de la red.

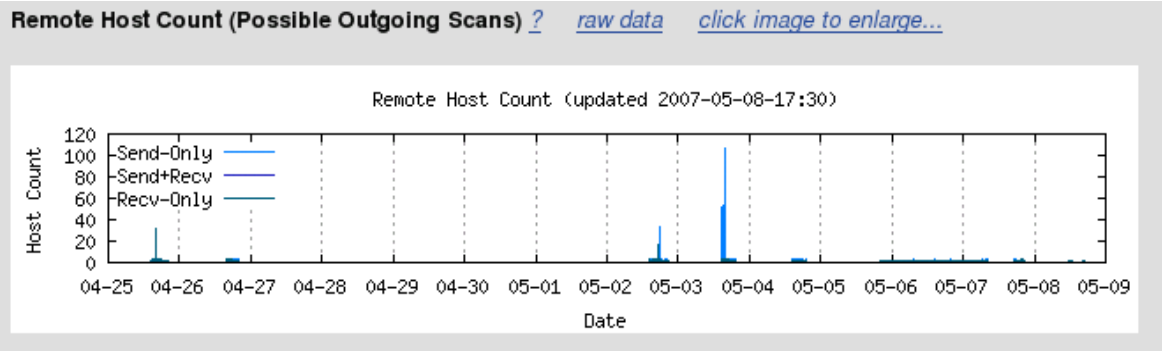


Fig. 58 Número de Host remotos.

La Fig. 59 muestra el tráfico para cada uno de los cinco hosts locales más ocupados en cada media hora.

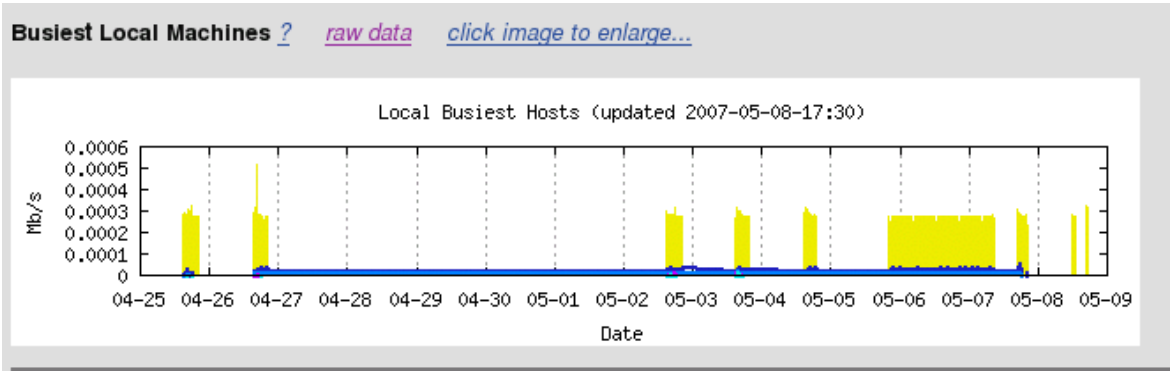


Fig. 59 Cinco Host locales más ocupados.

La Fig. 60 muestra el tráfico para cada uno de los cinco hosts remotos más ocupados en cada media hora.

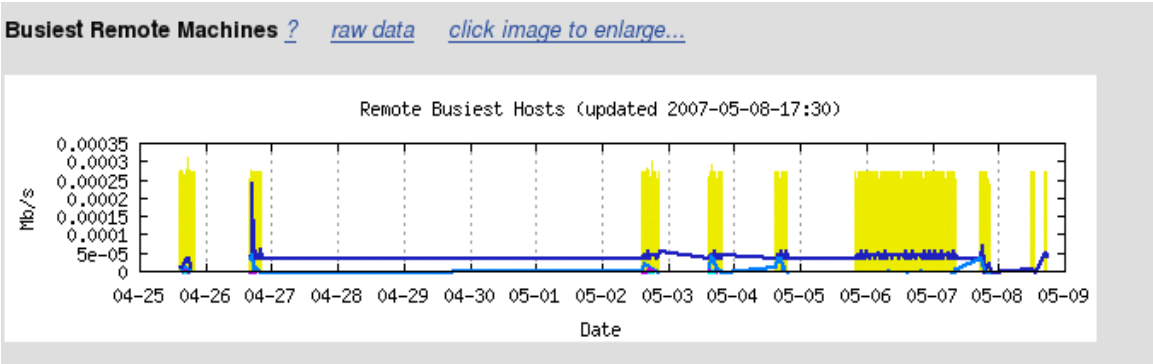


Fig. 60 Cinco Host Remotos mas usados.

IPAudit es una gran adición para la monitorización de tu red. Proporciona los informes que te dan una descripción de tu red, te informan sobre acontecimientos de la seguridad, y divulgan sobre anomalías. Cuando está utilizado conjuntamente con la detección de intruso, un incidente de la seguridad puede ser trazado hacia fuera con mucho detalles. Lo mejor de todo, IPAudit es una herramienta de software libre que es fácil de instalar y de mantener.

Nota: Los nombres de los paquetes mencionados en este informe son los que hemos utilizado en nuestro trabajo investigativo.

3.7 Comparación de las Herramientas de Gestión de Redes.

Todas las herramientas configuradas para el desarrollo de nuestro trabajo monográfico, son de mucha importancia para el Administrador de Red, a pesar que no todas nos proporcionan la misma información y eso es lo que hace a una mejor que la otra.

Hay que tomar en cuenta que algunas de las herramientas nos proporcionan información solo de manera gráfica y ésto hace un poco complicado su entendimiento, en cambio hay

otras que nos proporcionan información tanto gráfica como tabular esto facilita una mejor interpretación para el Administrador de la Red.

Se podría considerar que la mejor Herramienta de Gestión (de las que configuramos) es NTop, ya que proporciona amplia información de lo que ocurre en la red, y tomando en cuenta no solo un protocolo sino todos los protocolos de red posibles, de forma gráfica y tabular, es una herramienta muy fácil de entender; al igual que IpAudit y Wireshark que también proporcionan mucha información.

Cabe mencionar que NTop, IpAudit y Wireshark no hacen uso del protocolo de Gestión de red SNMP, en cambio las herramientas que hacen uso del mismo (SNMP) son: MRTG, Cacti y BigSister. Estas nos generan sólo información gráfica y eso las hace un poco complicadas a la hora de realizar los análisis de resultados. A continuación presentamos una tabla de comparación de las Herramientas que configuramos.

Tabla de comparación de las herramientas GNU, para gestión de redes.

Herramientas GNU	MRTG	Cacti	IpAudit	Ntop	BigSister	Wireshark
Uso de SNMP	si	si	no	no	si	no
Monitor web	si	si	si	si	si	no
Depende de Apache ?	si	si	si	no	si	no
Uso de RRdTool	no	si	si	si	no	no
Tipo de información generada	gráfica	gráfica	gráfica y tabular	gráfica y tabular	gráfica y tabular	gráfica y tabular
Buen rendimiento en redes con DHCP	no	no	si	si	no	si

VI. CONCLUSIONES.

Simple Network Management Protocol (SNMP) y cualquier Herramienta de Gestión de Redes, junto al ingenio de un buen administrador de red para crear script, constituyen un sistema muy potente para monitorizar toda la red, que además de utilizarlos para saber en tiempo real que ocurre y poder detectar cualquier anomalía, también nos será útil para saber el uso que se hace de los recursos informáticos.

En el presente trabajo dejamos bien documentada la instalación y configuración de seis Herramientas de Gestión, aunque no solo éstas existen; queda a elección del lector cuál de éstas, es la que necesita para gestionar la red que administra. Para hacer una buena elección debe fijarse en la tabla presentada en el apartado 3.7 Comparación de las Herramientas de Gestión.

Cabe mencionar que no todas las herramientas configuradas para el cumplimiento de nuestros objetivos, hacen uso del protocolo SNMP, pero aun así son muy buenas; claro está que tienen sus desventajas.

Hay herramientas que nos proporcionan solo información gráfica, y otras que nos proporcionan información tanto gráfica como tabular, por lo que se recomienda tener al menos dos herramientas configuradas en nuestro servidor.

VII. RECOMENDACIONES.

- Hacer una investigación e implementación de protocolos de gestión de redes más avanzados como RMON.

- A los Administradores de la red de la UNAN-León, se les recomienda el uso de al menos dos herramientas de gestión a la vez, para obtener resultados precisos sobre el trafico de entrada y salida de la red.

- Implementar el programa decodificador escrito en C, de manera que se validen las entradas y ademas pueda decodificar mensajes SNMP en formato largo.

- Que se realice una modificación y adaptación de una de las Herramientas de Gestión de redes, según las necesidades de la red de la UNAN-León, de manera que ésta envíe notificaciones mediante correo, cuando aparecen problemas en la red y cuando se resuelven.

VIII. Glosario.

Agente: En el modelo cliente/servidor, es la parte del sistema que facilita el intercambio de la información entre el cliente y el servidor.

Apache: El servidor HTTP más ampliamente disponible en Internet. Soporta los lenguajes PERL y PHP.

API: Una API (del ingles Application Programming Interface - Interfaz de Programación de Aplicaciones) es un conjunto de especificaciones de comunicación entre componentes software. Representa un método para conseguir abstracción en la programación, generalmente (aunque no necesariamente) entre los niveles o capas inferiores y los superiores del software.

Asíncrona: Tipo de comunicación donde cada byte se transmite del emisor al receptor de modo independiente. Es la que utilizan los módems, por ejemplo.

CCITT: Son las siglas de Comité Consultivo Internacional Telegráfico y Telefónico - Consultative Committee for International Telegraphy and Telephony - Comité Consultatif International Télégraphique et Téléphonique, antiguo nombre del comité de normalización de las telecomunicaciones dentro de la UIT ahora conocido como UIT-T.

Consola: Interfase de comandos de un sistema operativo que permite el envío de ordenes a la computadora a través del teclado.

CRON: Es un admistrador regular de procesos en segundo plano ("demonio") que ejecuta programas a intervalos regulares (por ejemplo, cada minuto, día, semana o mes). Los procesos que deben ejecutarse y la hora en la que deben hacerlo se especifican en el fichero crontab.

DARPA: Defense Advanced Research Projects Agency o Agencia de Investigación en Proyectos de Defensa Avanzada, responsable del desarrollo de ARPANET, basamento inicial del cual surgió

Internet.

Datagrama: Fragmento de paquete que es enviado con la suficiente información como para que la red pueda simplemente encaminar el fragmento hacia el ordenador receptor, de manera independiente a los fragmentos restantes. Esto puede provocar una recomposición desordenada o incompleta del paquete en el ordenador destino.

Encriptar: Alterar información digital inteligible (un archivo o correo electrónico, por ejemplo) utilizando códigos secretos para que la información sea ininteligible para partes no autorizadas. Al encriptar mensajes, únicamente usted y la persona con la que se comunica pueden leer el mensaje. El mensaje viaja a través de la Internet de manera ininteligible, o la información se almacena en su computadora personal del mismo modo, para evitar que sea leída por terceros no autorizados. Conversión de los datos propios a un código privado e ininteligible para los demás. Básicamente, un password es una encriptación de una identidad. La encriptación transforma la información en un texto incomprensible, por medio de un algoritmo (o secuencia de operaciones) que responde a una clave de encriptación denominada "llave".

Estación: Corresponde a un elemento computacional de trabajo, por ejemplo un PC, un Mac, una impresora, etc.

Gestor: Programa que se encarga de una tarea específica, como el gestor de ficheros, el de impresión, o el de memoria, dentro del sistema operativo de un ordenador.

GNU/Linux: Es la denominación defendida por Richard Stallman y otros para el sistema operativo que utiliza el kernel Linux en conjunto con las aplicaciones de sistema creadas por el proyecto GNU. Comúnmente este sistema operativo es denominado simplemente Linux.

GNUplot: Programa de computadora para la representación de funciones y de datos.

Hardware: Se denomina hardware o soporte físico al conjunto de elementos materiales que componen un ordenador. En dicho conjunto se incluyen los dispositivos electrónicos y electromecánicos, circuitos, cables, tarjetas, armarios o cajas, periféricos de todo tipo y otros elementos físicos.

Hosts: Computadora que realiza algunas tareas en beneficio de otras aplicaciones llamadas clientes. Algunos servicios habituales son los servicios de archivos, que permiten a los usuarios almacenar y acceder a los archivos de un ordenador y los servicios de aplicaciones, que realizan tareas en beneficio directo del usuario final.

IAB : Consejo regulador el cual toma decisiones sobre los estándares que regirán a Internet. Determina las necesidades técnicas a medio y largo plazo; así como la toma las decisiones sobre la orientación tecnológica de la Internet. Aprueba las recomendaciones y estándares del Internet a través de una serie de documentos denominados.

IESG: Grupo de Dirección de Ingeniería de Internet. (Internet Engineering Steering Group). Grupo voluntario que se encarga de considerar los estándares propuestos por el Internet Engineering Task Force (IETF) que posteriormente ser n establecidos por el IAB.

IETF: El IETF (Internet Engineering Task Force, en castellano Grupo de Trabajo en Ingeniería de Internet) es una organización internacional abierta de normalización, que tiene como objetivos el contribuir a la ingeniería de Internet, actuando en diversas áreas, tales como transporte, encaminamiento, seguridad. Fue creada en EE.UU. en 1986.

Interface: Una interfaz es la parte de un programa informático que permite a éste comunicarse con el usuario o con otras aplicaciones permitiendo el flujo de información. Zona de contacto o conexión entre dos componentes de "hardware"; entre dos aplicaciones; o entre un usuario y una aplicación. Apariencia externa de una aplicación informática.

Interred: Se refiere a la capa de red o nivel de red del modelo OSI.

IRC: Significa Internet Relay Chat, que es un protocolo de comunicación en tiempo real basada en texto, la cual permite debates en grupo y/o privado, el cual se desarrolla en canales de chat que generalmente comienzan con el caracter # o &, este último solo es utilizado en canales locales del servidor. Es un sistema de charlas muy popular actualmente y ampliamente utilizado por personas de todo el mundo.

ISDN: Integrated Services Digital Network) Red Digital de Servicios Integrados. En español se abrevia RDSI. En el servicio de ISDN las líneas telefónicas transportan señales digitales en lugar de señales analógicas, lo que aumenta considerablemente la velocidad de transferencia de datos a la computadora. Si se cuenta con el equipo y el software necesarios, y si la central telefónica local ofrece ISDN y el proveedor de servicios lo soporta, el ISDN es posible utilizarlo.

ISO: International Organization for Standardization. Fundada en 1946, es una federación internacional que unifica normas en unos cien países. Una de ellas es la norma OSI, modelo de referencia universal para protocolos de comunicación.

LDAP: LDAP (Lighweight Directory Access Protocol - Protocolo ligero de acceso a directorios.) en sí es un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red. LDAP puede considerarse una base de datos (aunque su sistema de almacenamiento puede ser otro diferente) al que pueden realizarse consultas

LIBPNG: Originalmente llamada pnglib. Es una biblioteca independiente de la plataforma que contiene las funciones de C para manejar imágenes del png. Es desarrollado por Guy Eric Schalnat, Andreas Dilger, Glenn Randers-Pehrson y otros. La versión actual es 1.2.8 y fue lanzada el 2 de diciembre de 2004.

MAC: En redes de computadoras Media Access Control address cuyo acrónimo es MAC es un identificador físico un número, único en el mundo, de 48 bits- almacenado en fábrica dentro de una tarjeta de red o una interface usada para asignar globalmente direcciones únicas en algunos modelos OSI (capa 2) y en la capa física del conjunto de protocolos de internet.

Netbios: Protocolo de red originalmente creado para redes locales de computadoras IBM PC. NetBIOS fue la API del producto llamado "PC Network", desarrollado por Sytec, empresa contratada por IBM. "PC Network" soportaba menos de 80 nodos y era bastante simple, pero en aquella época era más apropiado para los ordenadores personales que su pariente más viejo y complejo para mainframes de IBM, el SNA.

OSI: El modelo OSI (Open Systems Interconection) es la propuesta que hizo la Organización

Internacional para la Estandarización (ISO) para estandarizar la interconexión de sistemas abiertos. Un sistema abierto se refiere a que es independiente de una arquitectura específica. Se compone el modelo, por tanto, de un conjunto de estándares ISO relativos a las comunicaciones de datos.

OS X: Sistema Operativo desarollado por Apple para sustituir al tradicional sistema MacOs en los ordenadores MacIntosh. OSX es un sistema Unix basado en la variantes Free BSD y NextStep, de Next.

Perl: Lenguaje de programación utilizado en el WWW a través de un CGI, principalmente para realizar consultas a bases de datos como Oracle, SQL-Server, SyBase, etc, o a herramientas locales como WAIS. Perl es un lenguaje para manipular textos, archivos y procesos, proporciona una forma fácil y legible para realizar trabajos que normalmente se realizarían en Co en un shell. Perl nació y se ha difundido bajo el sistema operativo UNIX, aunque existe para otras plataformas.

peer to peer: Sistema de red en el que los archivos se reparten en diferentes computadoras, los usuarios accesan a éste de uno a otro en vez de por un servidor central.

PHP: (acrónimo recursivo de "PHP: Hypertext Preprocessor", originado inicialmente del nombre PHP Tools, o Personal Home Page Tools) es un lenguaje de programación interpretado. Aunque fue concebido en el tercer trimestre de 1994 por Rasmus Lerdorf no fue hasta el día 8 de Junio de 1995 que fue lanzada la versión 1.0. Se utiliza entre otras cosas para la programación de páginas web activas, y se destaca por su capacidad de mezclarse con el código HTML.

Placeholder: En matemáticas, y en otras disciplinas que implican lenguajes formales, incluyendo lógica matemática e informática, una variable libre es una notación para un lugar o los lugares en una expresión, en a la cual una cierta substitución definida puede ocurrir, o con respecto a cuál puede ocurrir una cierta operación (adición o cuantificación, dar dos ejemplos).

PNG: Es la Extensión que corresponde a un tipo de fichero gráfico de mapa de bits (Portable Network Graphics).

protocolo: Se le llama protocolo de red o protocolo de comunicación al conjunto de reglas que controlan la secuencia de mensajes que ocurren durante una comunicación entre entidades que

forman una red. En este contexto, las entidades de las cuales se habla son programas de computadora o automatismos de otro tipo, tales y como dispositivos electrónicos capaces de interactuar en una red.

Proxy: Servidor situado entre la máquina del usuario e Internet. Puede actuar como una barrera que protege y como un área "cache" para acelerar la visualización de una página Web.

Routers: Un router (enrutador o encaminador) es un dispositivo hardware o software de interconexión de redes de ordenadores/computadoras que opera en la capa 3 (nivel de red) del modelo OSI. Este dispositivo interconecta segmentos de red o redes enteras. Hacen pasar paquetes de datos entre redes tomando como base la información de la capa de red. Típicamente una máquina, aunque también puede ser un software, que actúa como puerta para permitir el acceso a los recursos de una red, independientemente de los protocolos o sistemas operativos de los usuarios.

RRdTool: Viene de Round Robin Databases, bases de datos circulares, es un sistema que permite almacenar y representar datos en intervalos temporales (ancho de banda, temperatura, ...). Guarda los datos en una base de datos que no crece en el tiempo y permite crear bonitas gráficas para representar los datos.

RTP: Real Time Protocol. Protocolo de Tiempo Real. Protocolo utilizado para la transmisión de información en tiempo real como por ejemplo audio y video en una video-conferencia.
RTT: Una medida de la corriente retrasa en una red

Script: En la programación de computadoras es un programa o una secuencia de instrucciones que es interpretado y llevado a cabo por otro programa en lugar de ser procesado por el procesador de la computadora.

Scrollig: Para ver líneas consecutivas de datos sobre la pantalla de visualización. El término voluta significa que una vez que la pantalla sea llena, cada uno de giro nuevo aparece en el borde de la pantalla y el resto de las líneas se mueven sobre una posición. Por ejemplo, cuando enrollas abajo, cada uno de giro nuevo aparece en el fondo de la pantalla y todas las otras líneas levantan una fila,

de modo que desaparezca la línea superior.

SIP: Establece las sesiones para las características tales como audio/videoconferencia, juego interactivo, y expedición de llamada que se desplegarán sobre las redes del IP que permiten así a abastecedores de servicio integrar servicios básicos de la telefonía del IP con Web, E-mail, y servicios de la charla.

Sniffer: Husmeador. Monitor de red. Es como si se pinchase un teléfono o la red telefónica. Son programas que monitorizan la información que circula por la red con el objeto de capturar información. Las placas de red tienen un sistema de verificación de direcciones mediante el cual saben si la información que pasa por ella está dirigida o no a su sistema. Si no es así, la rechaza.

SNMP: Simple Network Management Protocol o Protocolo Simple de Gestión de Redes, es aquel que permite la gestión remota de dispositivos de red, tales como switches, routers y servidores.

SMTP: Simple Mail Transfer Protocol (SMTP), o protocolo simple de transferencia de correo electrónico. Protocolo de red basado en texto utilizado para el intercambio de mensajes de correo electrónico entre computadoras y/o distintos dispositivos (PDA's, Celulares, etc).

Software : Software -también conocido como programática y aplicación informática- es la parte lógica del ordenador, esto es, el conjunto de programas que puede ejecutar el hardware para la realización de las tareas de computación a las que se destina. Es el conjunto de instrucciones que permite la utilización del equipo.

Spoof: Cuando se refiere a un "spoof" se quiere decir que la fuente dirección IP de un correo es falso, el que lo ha mandado no quiere que le conozcan.

Suite: Conjunto de programas diseñados para trabajar juntos. En español se suele llamar "Paquete Integrado". Es frecuente que incluyan un procesador de texto, una hoja de cálculo, un organizador personal, y pueden tener otros módulos, como gestores de bases de datos, programas de gráficos o presentaciones,

Switches: Un interruptor es un dispositivo de la red que selecciona una trayectoria o un circuito para enviar una unidad de datos a su destinación siguiente. Un interruptor puede también incluir la función de la rebajadora, de un dispositivo o del programa que pueden determinar la ruta y específicamente

qué punto adyacente de la red los datos se deben enviar. Un interruptor es generalmente un mecanismo más simple y más rápido que una rebajadora, que requiere conocimiento sobre la red y cómo determinar la ruta.

Tag: Líneas de código de programación que se pegan en la pagina web para controlar impresiones, clicks y visitantes únicos entre otras variables.

TCP/IP: Conjunto básico de protocolos de comunicación de redes, popularizado por Internet, que permiten la transmisión de información en redes de computadoras. El nombre TCP/IP proviene de dos protocolos importantes de la familia, el Transmission Control Protocol (TCP) y el Internet Protocol (IP).

Trap: Mecanismo de interrupción del programa que pone al día automáticamente el estado de la red a la dirección de la red alejada recibe. El agente del SNMP en el interruptor apoya estas trampas del SNMP.

UIT: Organismo Internacional que agrupa la práctica totalidad de las administraciones y operadores públicos de red. Algunas de sus comisiones, como la CCITT, establecen normas que se convierten en estándares internacionales para los servicios de telecomunicación.

UNIX: Es un sistema operativo portable, multitarea y multiusuario; desarrollado en principio por un grupo de empleados de los laboratorios Bell de AT&T, entre los que figuran Ken Thompson, Dennis Ritchie y Douglas McIlroy.

WWW: La World Wide Web (del inglés, Telaraña Mundial), es un sistema de hipertexto que funciona sobre Internet. Para ver la información se utiliza una aplicación llamada navegador web para extraer elementos de información (llamados "documentos" o "páginas web") de los servidores web (o "sitios") y mostrarlos en la pantalla del usuario.

IX. Bibliografía.

Páginas Web visitadas:

http://www.ibermatica.com/ibermatica/outsourcing2/outsourcinginfraestructuras/gestionredessistemas

http://es.wikipedia.org/wiki/Simple_Network_Management_Protocol

http://www2.rad.com/catalog/spanish/images/radch7s.pdf

http://www.coit.es/publicac/publbit/bit102/quees.htm

http://www-gris.det.uvigo.es/~mramos/gprsi/programa.html

http://lavisit.upf.edu/tutorial.jsp?content=/tutorial/snmp/tutorial_1.htm

http://www.ntop.org/nProbe.html

http://www-es.netapp.com/go/techontap/0107tot/ntop.html

http://www.ntop.org.

http://sourceforge.net/projects/ntop/

http://ipaudit.sourceforge.net/ipaudit-web/

http:www.securityfocus.com/infocus/1842

http://www.seguritos.org/phpnuke/modules.php?name=Downloads&d_op=viewdownload&cid=4&orderby=titleD

http://wireshark.softonic.com/ie/21702

http://software.elpais.com/rss/939/last_news_by_date_actualized.xml

http://forums.cacti.net/about11702.html

http://www.microsoft.com/technet/prodtechnol/windowsserver2003/es/library/ServerHelp/4afcfe9e-b2e5-4cf5-b235-9bc66a1aaf36.mspx?mfr=true

http://www.inf-cr.uclm.es/www/jprozas/GdR/T7_Tecnolog%EDa_de_Gesti%F3n_de_Red.pdf

Lista de RFC:

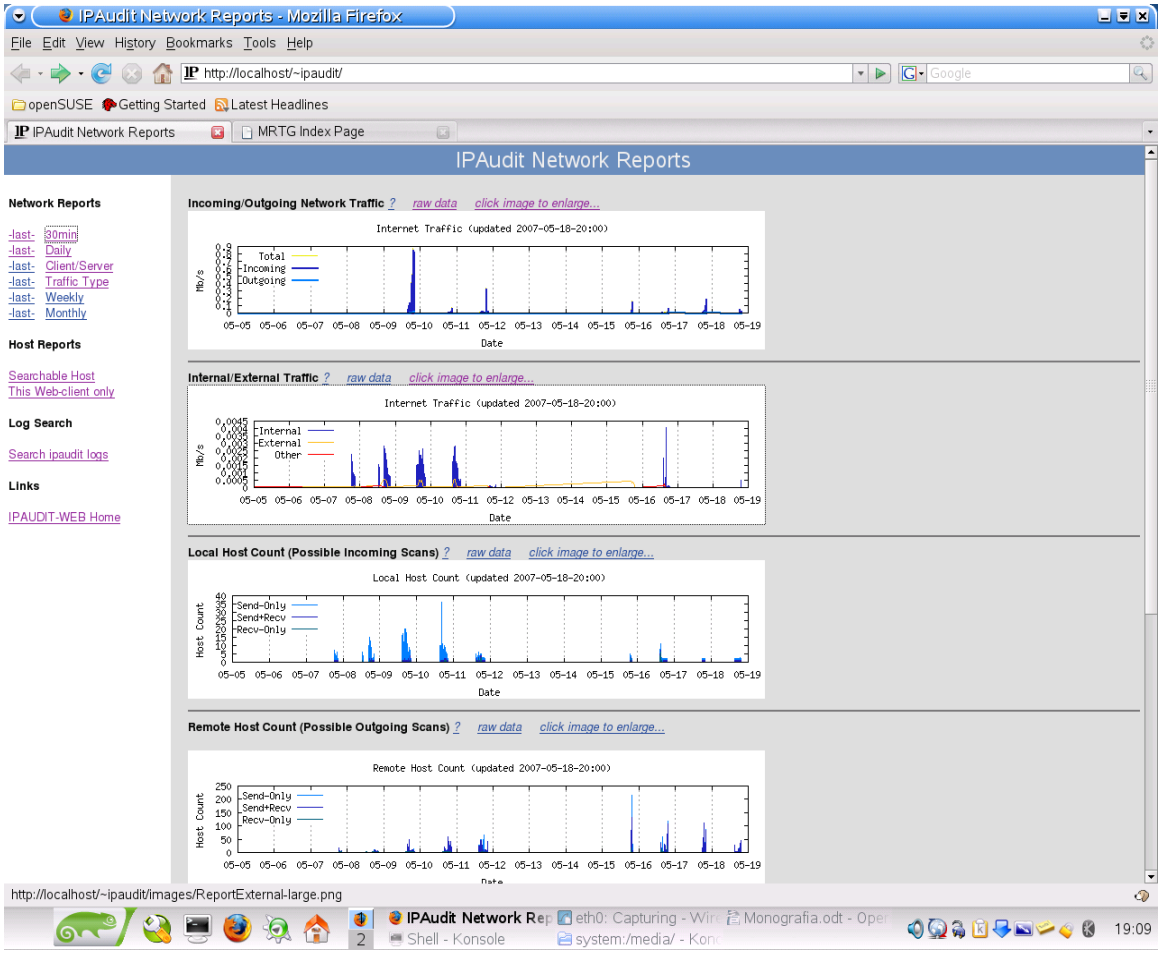
RFC 1212.- Definición de MIB

RFC 1157 - SNMP

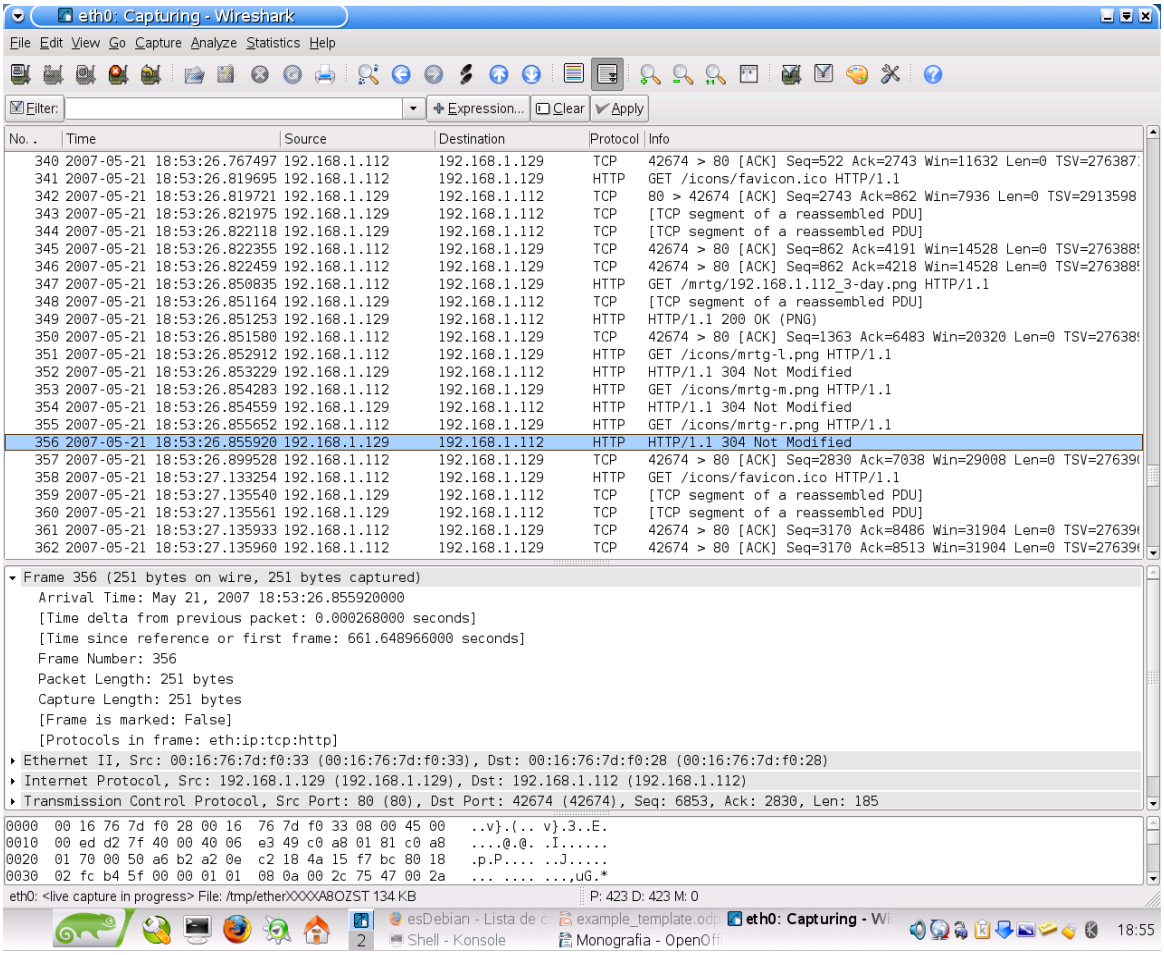
RFC 1155 - SMI

RFC 1156 y 1213 - MIB

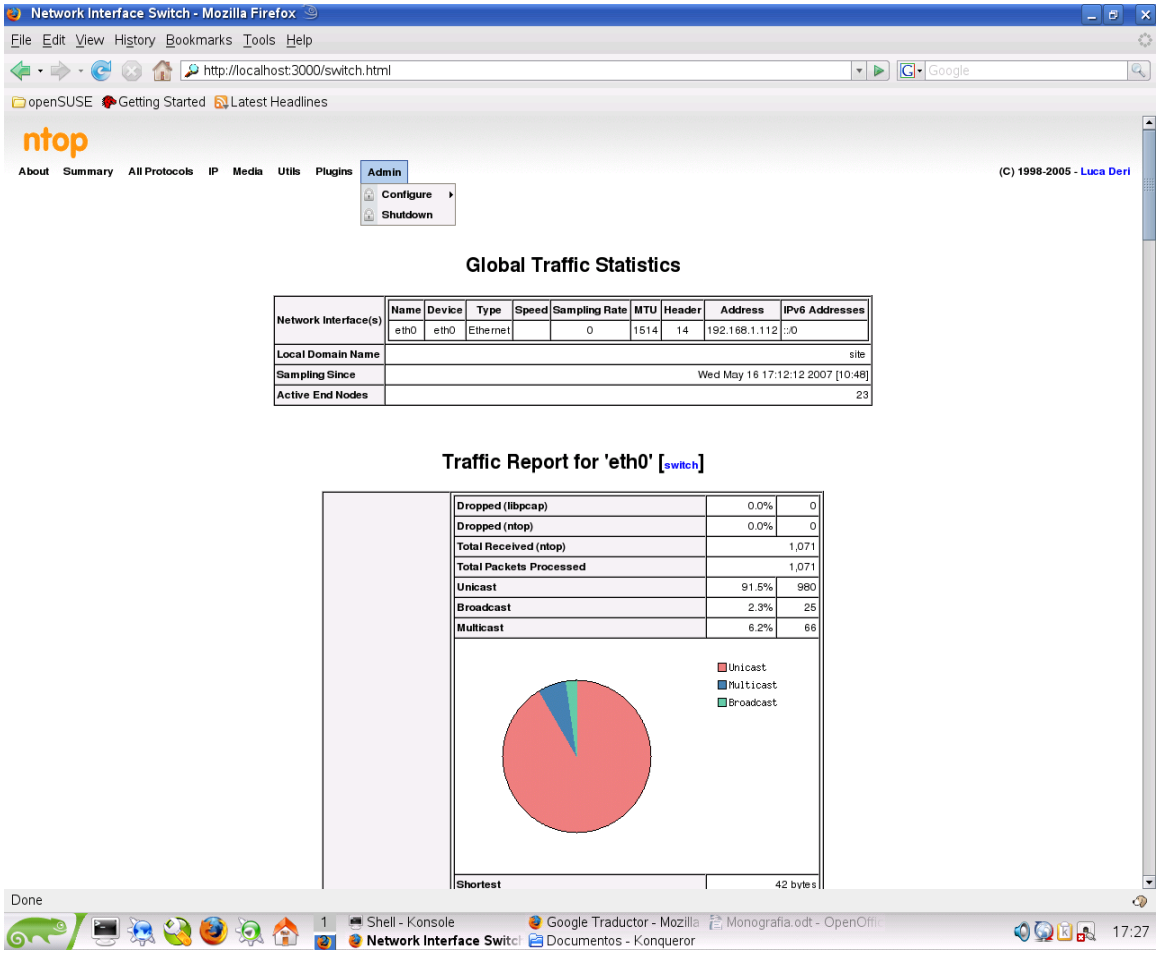
ANEXOS



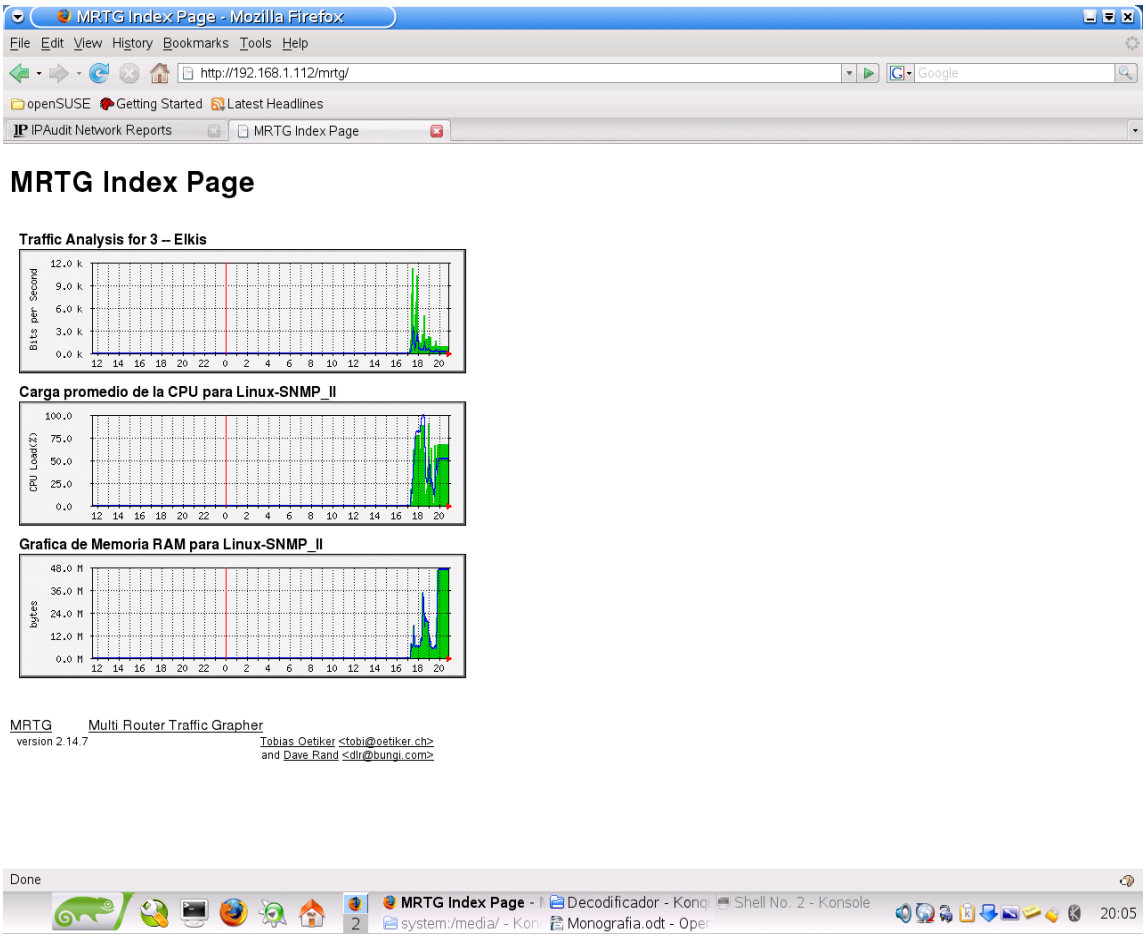
Página principal de IpAudit.



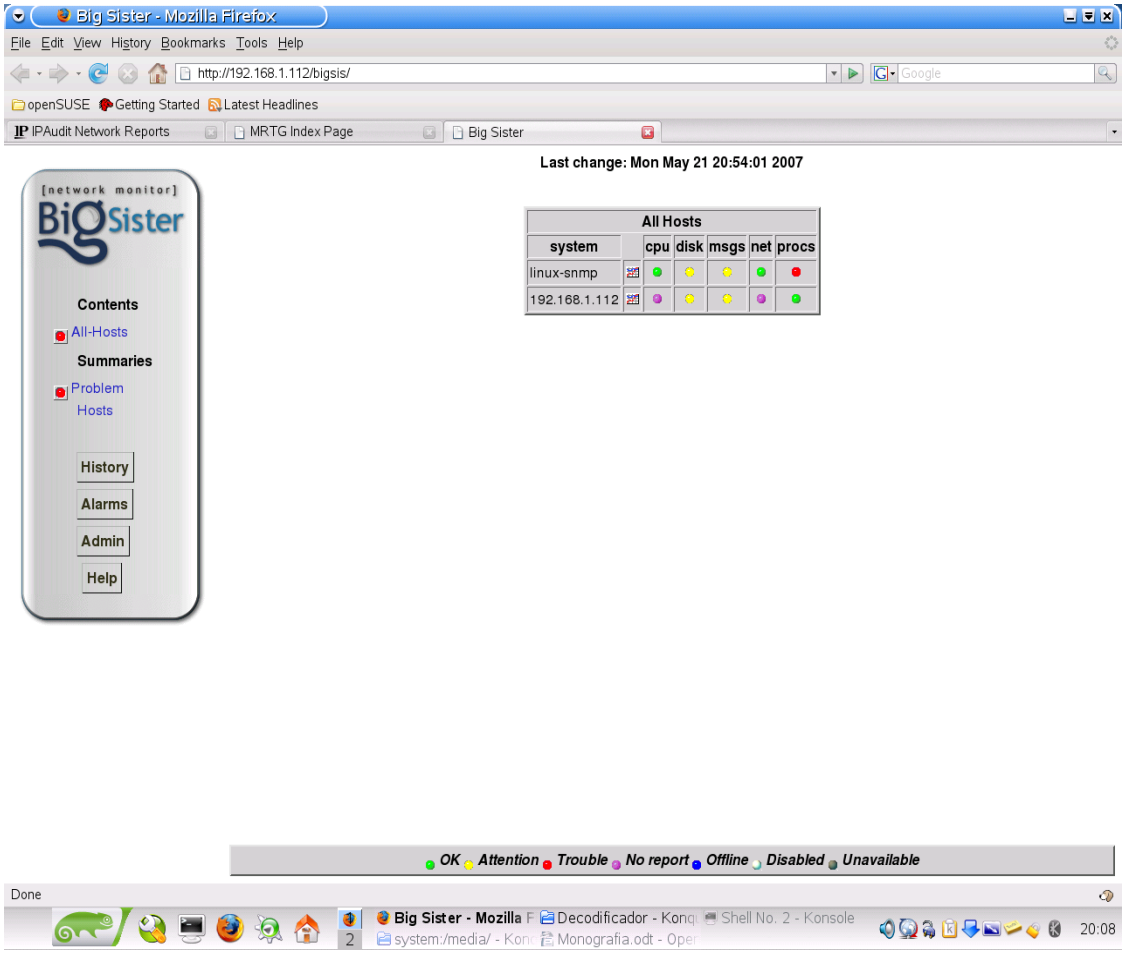
Wireshark después de una Captura.



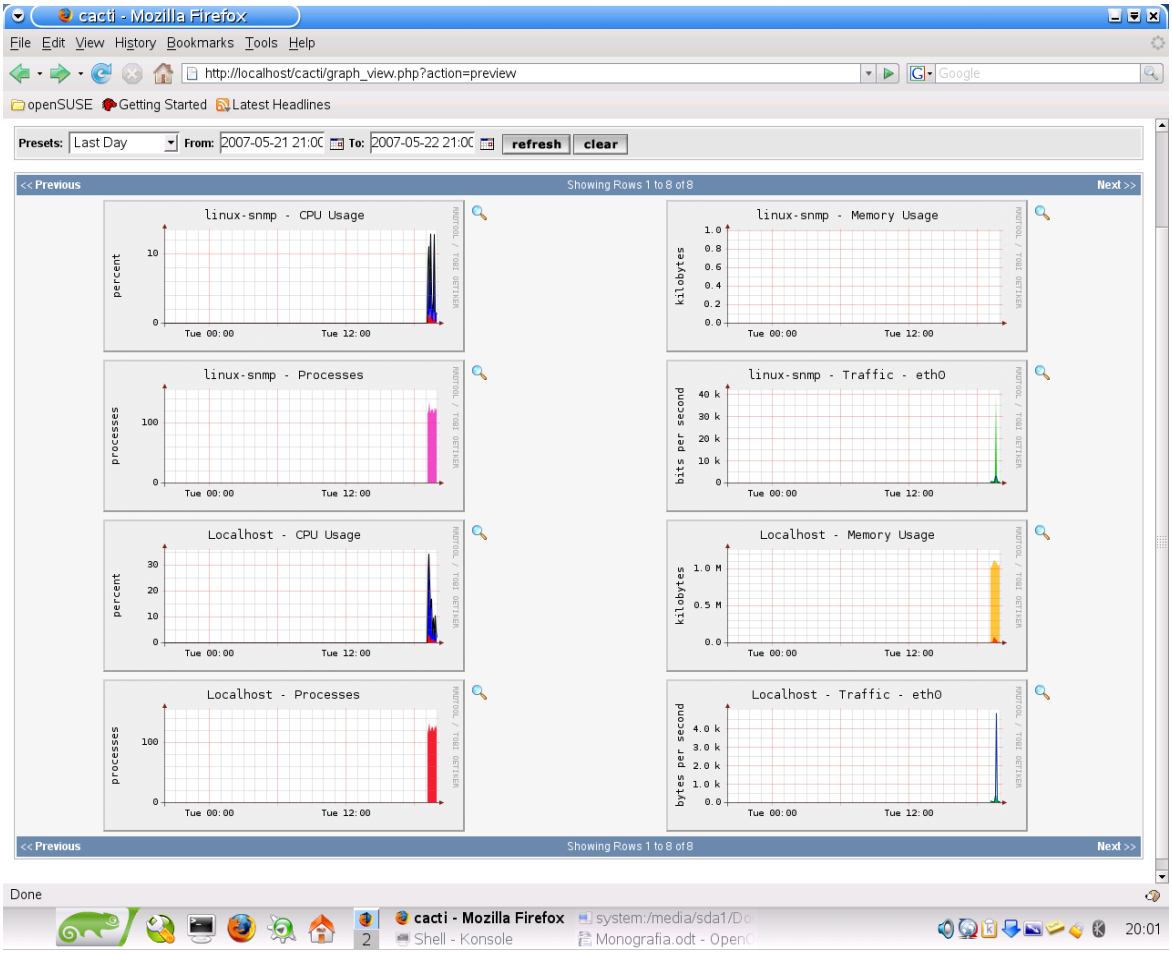
Página principal de NTOP.



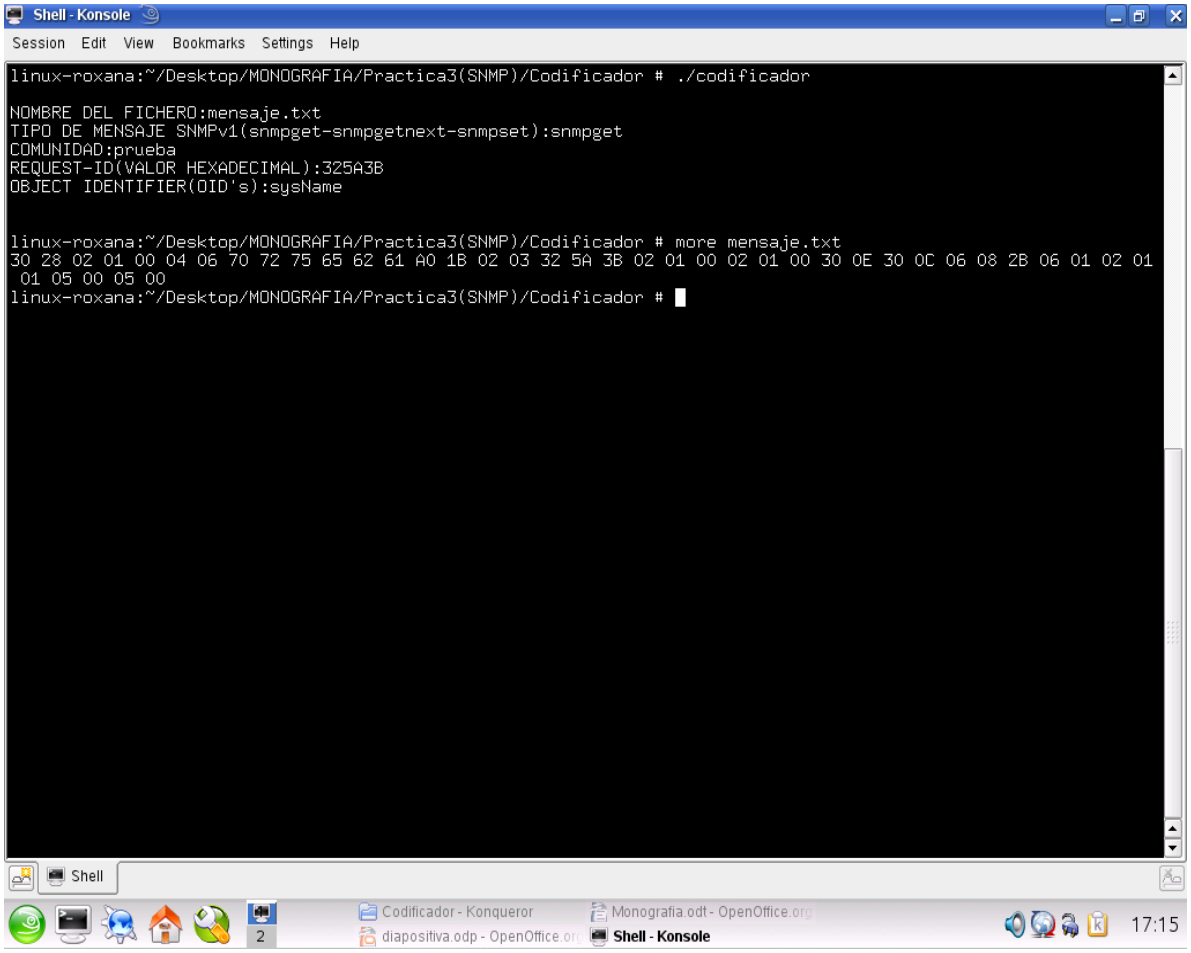
Ventana Principal de MRTG.



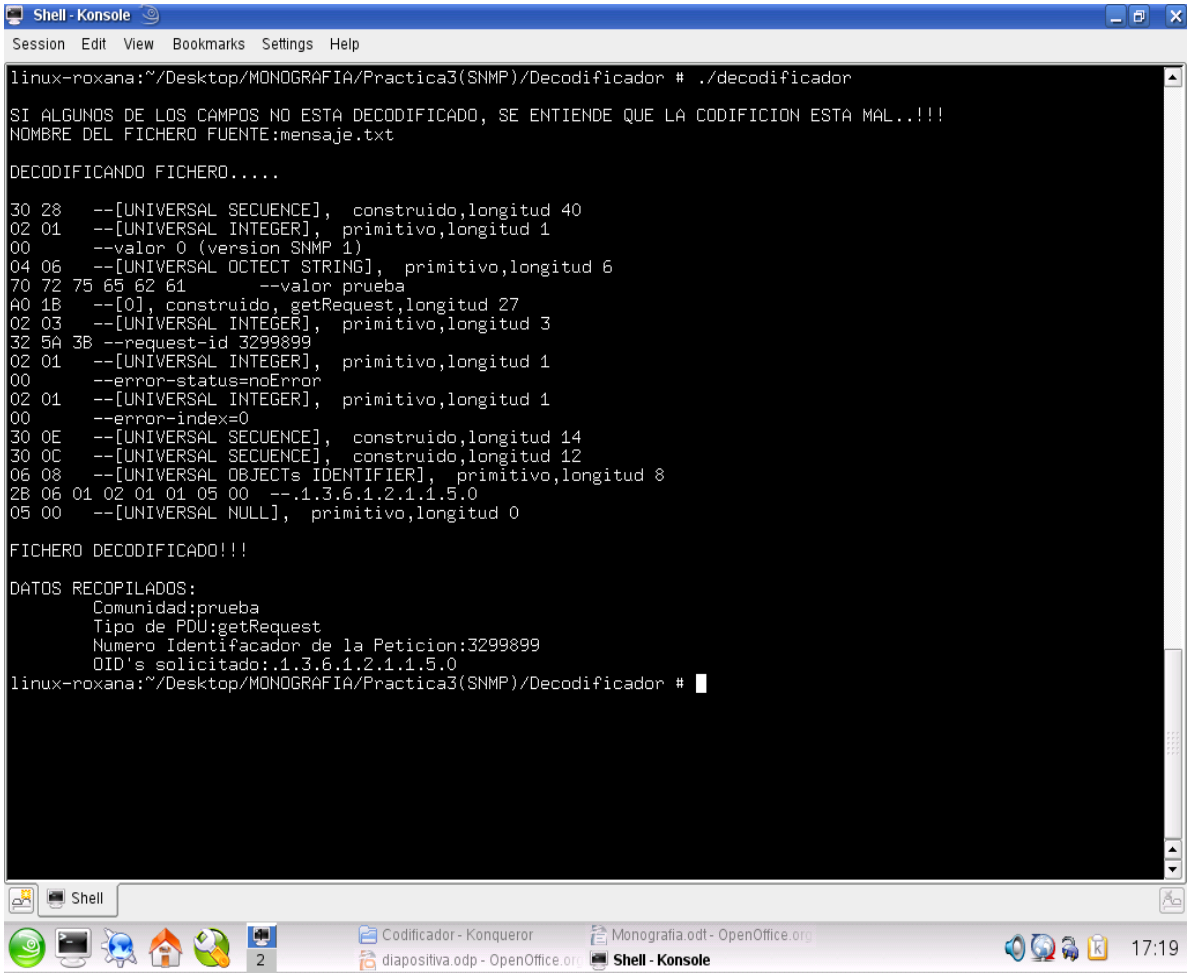
Página principal de BigSister.



CACTI en ejecución.



Ejecución del programa Codificador.



Ejecución del Programa Decodificador.