ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

"Implementación de un Sistema de Gestión y Administración de Redes Basados en el Protocolo Simple de Monitoreo de Redes SNMP en la Red ESPOL- FIEC "

INFORME DE

PROYECTO DE GRADUACION

Previo a la obtención del Título de:

INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

Presentada por:

Gregory Giancarlo Valarezo Saldarriaga

Julio Cesar Simisterra Huila

GUAYAQUIL - ECUADOR

AÑO: 2011

AGRADECIMIENTO

Deseo expresar mi más sinceras muestras de agradecimiento a Dios por iluminarme todo los días de mi vida y encaminarme por el camino del bien. A mis padres, y mi hermano por creer y confiar en mí siempre brindándome su apoyo. Al MSc Cesar Yépez Flores por toda la ayuda y consejos brindados para este proyecto y durante el transcurso de mi carrera profesional.

Gregory Giancarlo Valarezo Saldarriaga.

Agradezco en primer lugar a Dios, luego al Msc. Cesar Yépez Flores, por haber hecho posible la realización de este proyecto. También a mis amigos y a todas aquellas personas que supieron estar en el momento adecuado dando palabras de aliento y brindaron siempre todo su apoyo para continuar. A todos ellos muchas gracias.

Julio César Simisterra Huila.

DEDICATORIA

Dedico este trabajo a Dios, a mi familia por creer y confiar siempre en mí y a mi novia Kerly por su amor, comprensión y apoyo.

Gregory Giancarlo Valarezo Saldarriaga

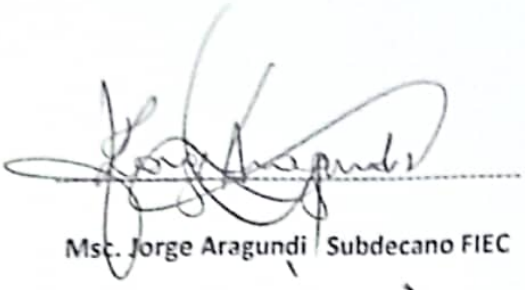
Esto quiero dedicárselo a mis padres Lenin y Marlene por todo lo que me han dado en esta vida, a mis hermanos, por acompañarme en todo momento brindarme siempre su apoyo, a mis tíos por estar siempre dispuestos a ayudarme.

A mi abuelita, qué siempre me guía y estoy seguro que en estos momentos al igual que mis padres está orgullosa de mi. Y por la ayuda en la realización de este proyecto y a mi novia Yuvis, por el apoyo durante todo este tiempo.

Julio César Simisterra Huila

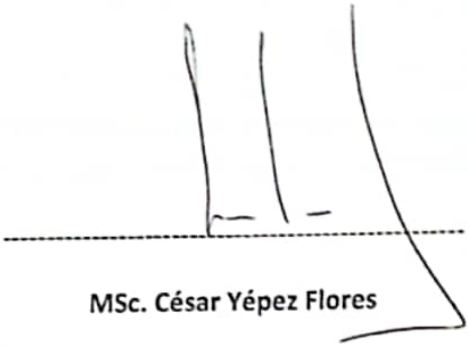
TRIBUNAL DE GRADUACIÓN.

TRIBUNAL DE GRADUACIÓN.



Msc. Jorge Aragundi Subdecano FIEC

PRESIDENTE



MSc. César Yépez Flores

DIRECTOR DEL PROYECTO

MSc.Sara Judith Rios Orellana

VOCAL PRINCIPAL

DECLARACIÓN

La responsabilidad del contenido de este Proyecto de Graduación, nos corresponde exclusivamente; y el patrimonio intelectual de la misma a la Escuela Superior Politécnica del Litoral

Gregory Giancarlo Valarezo Saldarriaga

Julio Cesar Simisterra Huila

INDICE GENERAL

INDICE DE FIGURAS

INDICE DE TABLAS

ANEXOS

BIBLIOGRAFIA

INTRODUCCION

Las necesidades de incrementar y mejorar el uso de las redes de información ha provocado que la administración y monitoreo de las mismas sea un factor preponderante en el campo de las telecomunicaciones, para que se pueda mantener un adecuado funcionamiento. Es aquí donde se hace necesaria una herramienta, que nos facilite el monitoreo y administración de tráfico de datos en redes LAN.

Para ello un administrador de Red o alguien a cargo de la supervisión de máquinas o servidores de una empresa, es muy importante saber el estado y tener el control de estas, ya que con esto se logra en parte una administración potencialmente satisfactoria, esta posibilidad la brinda las herramientas de Monitoreo y Gestión de la red, ya que por medio de estas podremos saber el estado de nuestras Maquinas, Roueters, Switchs y además podremos saber como andan nuestros servicios de red como son: dns, dhcp, web, Proxy, ftp, etc.

Saber que procesos corren en una maquina, que ancho de banda consume esta, cual es la carga promedio del sistema, el uso de la memoria actual, el trafico de red de cada una de las interfaces (si tuviese más de una), que tipo de software utiliza, y muchos otros aspectos que para un administrador de red son de vital importancia para detectar fallos y actuar con precisión.

GLOSARIO

1. SNMP (Protocolo simple de monitoreo de Redes): usado para administrar la configuración de dispositivos de red en una estación de trabajo.

2. MIB (Base de Información de Administración): es una colección de información que ordena en forma de árbol donde se registran las variables a monitorear.

3. OID (Objeto Identificador): idéntica de manera única cada objeto representado en la MIB.

4. IDS (Instrucción de Detección del Sistema): es una herramienta que permite monitorear el comportamiento y el uso que se le da a los recursos en una maquina.

5. ASN1 (Notación de Sintaxis Abstracta 1): Lenguaje utilizado para definir tipos de datos.

6. Agente SNMP: Programa que permite a un dispositivo responder a solicitudes SNMP.

7. Solicitud SNMP: solicitudes enviadas o recibidas por una entidad administradora estas pueden ser Get, Set Trap, etc.

8. Integridad de Datos: reflejan la realidad y que corresponda con lo que debe de ser y que no se haya modificado.

9. BER (Reglas Básicas de Codificación): es un conjunto de reglas para traducir valores ASN1 a un flujo de octetos para transmitir por la red.

10. Control de acceso y autorización: el proceso de determinar los recursos y servicios que puede usar una entidad.

11. NMS (Red de administración de la estación): estación de red encargada de gestionar varios dispositivos de red.

12. PDU (Protocolo de Unidad de datos): define la estructura de la información que va a ser enviada por la red.

CAPÍTULO 1

MONITOREO Y ADMINISTRACIÓN DE REDES

1.1 INTRODUCCIÓN A LA ADMINISTRACIÓN Y MONITOREO DE REDES

La administración de redes se ha convertido en un aspecto crítico, especialmente en redes de computadores de varios vendedores heterogéneos. El modelo Cliente – Servidor con una gran cantidad de estaciones de trabajo necesita de la administración de redes para manejar y controlar las redes y los componentes asociados al hardware y al software.

1.1.1 DEFINICIÓN Y OBJETIVOS DE LA ADMINISTRACIÓN DE REDES

1.1.1.1 ADMINISTRACIÓN

La administración de redes consiste en la organización, control, toma de precauciones y supervisión de la red, para mantener su funcionamiento eficiente, mediante el empleo de herramientas de red, aplicaciones y dispositivos.

A continuación se destaca un conjunto de actividades que a corto plazo permiten realizar un seguimiento de las tareas administrativas y elaborar informes periódicos para su posterior estudio:

- Detección y aislamiento de fallas.
- Evaluación del tráfico de datos.
- Mantenimiento de registro histórico de problemas.
- Mantenimiento de configuraciones.
- Contabilidad de red.
- Control de acceso.

1.1.1.2 OBJETIVOS DE LA ADMINISTRACIÓN DE RED

Los objetivos de la administración de red son los siguientes:

- Proporcionar herramientas automatizadas: manuales de administración de red para controlar posibles fallas o degradaciones en el desempeño de la misma.

- Disponer de estrategias de administración: optimizar la infraestructura existente, optimizar el rendimiento de aplicaciones y servicios. Además, prever los crecimientos en la red esperados debido al cambio constante en la tecnología.

- Alta disponibilidad de la red: proveyendo eficiencia operacional, reduciendo los downtime de la red y del sistema y proveyendo tiempos de respuesta aceptables. Los problemas de la red deben ser rápidamente detectados y corregidos.

- Reducción de costos operacionales de red: este es uno de los motivos primarios detrás de la administración de redes. Como las tecnologías cambian

rápidamente, es deseable la administración de sistemas heterogéneos y múltiples protocolos.

- Alta eficiencia: debemos incrementar la eficiencia en detrimento de otros objetivos de la administración pero dependerá de otros factores tales como utilización, costo operacional, costo de migración y flexibilidad.

- Facilidad de uso: las interfaces de usuario son críticas para el éxito de un producto. El uso de aplicaciones de administración de redes no debe incrementar la curva de aprendizaje.

1.1.1.3 FUNCIONES DE ADMINISTRACIÓN DE RED

Las funciones de administración de red se basan en dos procedimientos que ayudan a llevar a cabo numerosas tareas, estos procedimientos son los siguientes:

- **Monitoreo**

El monitoreo es un proceso eminentemente pasivo, el cual se encarga de observar el estado y comportamiento de la configuración de red y sus componentes.

También se encarga de agrupar todas las operaciones para la obtención de datos acerca del estado de los recursos de la red.

- **Control**

El control es un proceso que se lo considera activo, debido a que permite tomar información de monitoreo y actuar sobre el comportamiento de los componentes de la red administrada.

Abarca la configuración y seguridad de la red, como por ejemplo, alterar parámetros de los componentes de la red.

1.1.2 DEFINICIÓN Y OBJETIVOS DEL MONITOREO DE REDES.

1.1.2.1 MONITOREO

Monitoreo es la realización del estudio del estado de los recursos. Las funciones del monitoreo 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.

El monitoreo de una red abarca 4 fases:

- Definición de la información de administración que se monitorea.
- Acceso a la información.
- Diseño de políticas de administración.
- Procesamiento de la información.

Los tipos de monitoreo son:

- Local.
- Remoto.
- Automático.
- Manual.

Los elementos monitoreados pueden ser:

- En su totalidad.
- En Segmentos.

El monitoreo puede ser realizado en forma:

- Continua.
- Eventual.

1.1.2.2 OBJETIVOS DEL MONITOREO

Los objetivos del monitoreo son los siguientes:

- Identificar la información a monitorear.

- Diseñar mecanismos para obtener la información necesaria.
- Utilizar la información obtenida dentro de las distintas áreas funcionales de administración de red.

- Tomar nuevas medidas sobre aspectos de los protocolos, colisiones, fallas, paquetes, etc.

- Almacenar la información obtenida en Bases de Información de gestión para su posterior análisis.
- Del análisis, obtener conclusiones para resolver problemas concretos o bien para optimizar la utilización de la red.

Dentro del monitoreo de la actividad de la red, los eventos típicos que son monitoreados suelen ser:

- Ejecución de tareas como la realización de copias de seguridad o búsqueda de virus.

- Registro del estado de finalización de los procesos que se ejecutan en la red.

- 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, etc.

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

- Mensajes en la consola: método en el que se suele codificar en función de su importancia.

- Mensajes por correo electrónico: método mediante el cual se envía contenido el nivel de prioridad y el nombre del evento ocurrido.

1.2 ASPECTOS FUNCIONALES DE LA ADMINISTRACIÓN DE RED

La Organización Internacional de Estándares ISO (International Organization for Standardizations) ha definido la arquitectura de Administración OSI (Open System Interconnection), cuya función es permitir supervisar, controlar y mantener una red de datos.

Ésta arquitectura de administración, se encuentra dividida en cinco categorías de servicios de administración denominadas Áreas Funcionales Específicas de Administración, las cuales se muestran a continuación:

- Administración de prestaciones.
- Administración de fallas.
- Administración de contabilidad.
- Administración de configuraciones.
- Administración de seguridad.

Los aspectos o categorías funcionales de la administración de red brindan servicio a las actividades de Monitoreo y Control de red. Y se las puede ubicar de la siguiente manera:

a) Monitoreo de la red: obtiene información de los elementos:

- Administración de prestaciones.
- Administración de fallas.
- Administración de contabilidad.
- Administración de configuraciones.

b) Control de la red: actúa sobre los elementos:

- Administración de configuraciones.
- Administración de seguridad.

1.2.1 ADMINISTRACIÓN DE PRESTACIONES O RENDIMIENTO

Es medir la calidad de funcionamiento, proveer información disponible del desempeño de la red (hardware y software), asegurar que la capacidad y prestaciones de la red correspondan con las necesidades de los usuarios, analizar y controlar parámetros como: utilización, rendimiento, tráfico, cuellos de botella, tiempo de respuesta, tasa de error, throughput, etc.; esto de los distintos componentes de red como switches, ruteadores, hosts, etc., para poder ajustar los parámetros de la red, mantener el funcionamiento de la red interna en un nivel aceptable, poder efectuar análisis precisos y mantener un historial con datos estadísticos y de configuración, predecir puntos conflictivos antes de que éstos causen problemas a los usuarios.

El conocimiento de esta información nos permite en el futuro tomar acciones correctivas como balanceo o redistribución de tráfico, establecer y reportar tendencias para ser utilizadas en la toma de decisiones y planificación del crecimiento.

Se debe definir claramente los parámetros de funcionamiento o desempeño alrededor de los cuáles, se van a organizar las tareas de Administración de prestaciones como las siguientes:

- Obtención de la información de funcionamiento de la red a través del monitoreo sobre los recursos disponibles.

- Análisis de la información recolectada para determinar los niveles normales de utilización de la red.

- Comparación entre los valores obtenidos y los normales, para generar acciones de inicio de alarmas que pueden generar la toma de medidas preventivas o correctivas.

1.2.2 ADMINISTRACIÓN DE FALLAS

Los objetivos de esta área son: detección, aislamiento, corrección, registro y notificación de los problemas existentes en la red, sondeo periódico en busca de mensajes de error y establecimiento de alarmas.

Las consecuencias de estas fallas pueden causar tiempo fuera de servicio o la degradación inaceptable de la red, por lo que es deseable su pronta *detección y corrección*.

La diferencia entre falla y error está en que un error es un evento aislado como la pérdida de un paquete o que éste no llegue correctamente, pero una falla es un funcionamiento anormal que requiere una intervención para ser corregido. La falla se manifiesta por un funcionamiento incorrecto o por exceso de errores.

Las acciones o procedimientos para esta corrección son:

- Determinar exactamente dónde está la falla.

- Aislar el resto de la red, para que pueda seguir operando sin interferencia.

- Reconfigurar la red para minimizar el impacto de operar sin el componente averiado.

- Reparar o reemplazar el componente averiado para devolver la red al estado inicial.

- Si es posible, ejecutar un proceso de corrección automática y lograr el funcionamiento óptimo de la red.

- Registrar la detección y la resolución de fallas.

- Hacer seguimiento de la reparación de fallas.

1.2.3 ADMINISTRACIÓN DE LA CONFIGURACIÓN

Es el proceso de preparación de los dispositivos, puesto que la configuración de éstos, determina el comportamiento de los datos en la red.

Las funciones de ésta administración son: inicialización, desconexión o desactivación ordenada de la red o de parte de ella, mantenimiento y adición de componentes, reconfiguraciones, definición o cambio de parámetros de configuración, denominación de los elementos de la red, conocimiento de que dispositivos hay en la red, hardware y configuraciones de software de dichos dispositivos.

Las tareas que se presentan en la administración de configuración son:

- Es deseable que el arranque y parada de componentes específicos, se puedan realizar de forma remota.

- Definir información de configuración de recursos.

- Mantener ésta información, por si se sufre un ataque, poder realizar una

comprobación de la información de configuración para asegurar que permanece en un estado correcto.

- Modificación de propiedades de recursos e información al usuario de estos cambios.

- Control de versiones de software.

- Actualización de software.

- Establecer qué usuarios pueden utilizar qué recursos.

- Inicialización y finalización de servicios de red.

Las herramientas típicas para ésta administración son: monitorear la red para ver qué elementos hay activos y con qué características obtener la información, para saber de qué modo están conectados entre sí los diferentes elementos, ésta información se mantiene para ayudar a otras funciones de administración.

1.2.4 ADMINISTRACIÓN DE LA SEGURIDAD

Su objetivo es controlar el acceso a los recursos de la red, y protegerla de modo que no pueda ser dañada (intencional o involuntariamente), y que la información que es vulnerable pueda ser utilizada con una autorización apropiada. Comprende el conjunto de facilidades mediante las cuales, el administrador de la red modifica la funcionalidad que proporciona la red frente a intentos de acceso no autorizados.
En la Administración de Seguridad se pueden tener dos tipos de ataques:

- Ataques Activos.
- Ataques Pasivos.

1.2.4.1 ATAQUES ACTIVOS

En este tipo de ataques existe evidencia del hecho por mal funcionamiento de componentes o servicios, o por sustitución de usuarios en ejecución de tareas orientados a tratar de conseguir información privilegiada o interrumpir un servicio crítico para la organización, puede ser desde el interior o del exterior.

Ejemplos de estos ataques son: modificación del contenido de los datos que circulan por la red, alteración del orden de llegada de los datos, supresión de mensajes con un destino particular, saturación de la red con datos inútiles para degradar la calidad de servicio, engaño de la identidad de un host o usuario para acceder a datos confidenciales, desconfiguraciones para sabotaje de servicios.

1.2.4.2 ATAQUES PASIVOS

Ataques difíciles de detectar, ya que no se produce evidencia física del ataque pues no hay alteración de datos ni mal funcionamiento o comportamiento fuera de lo habitual de la red, escucha o “intercepción del tráfico de la red y los servicios involucrados”, estudio de parámetros de configuración de manera ilegal por parte del intruso, robo de información sensible para las organizaciones.

Para cualquiera de los tipos de ataques se puede prevenir o solucionar a través de las siguientes actividades:

- Fortalecer políticas de administración y asignación de claves.
- Historiales de seguridad, para posterior análisis.
- Uso de cortafuegos para monitorear y controlar los puntos de acceso internos y externos a la red.
- Encriptar o cifrar la información enviada por la red.

- Localizar la información importante.

- Registrar los usuarios que consultan dicha información y durante qué períodos de tiempo, así como los intentos fallidos de acceso.
- Señales de alarma.

- Establecimiento de mecanismos y políticas de prevención.

- Sistemas de detección de intrusos.

- Sensibilización de seguridad en el usuario.

- Mantenimiento del sistema operativo y sus aplicaciones relacionadas.

- Utilizar herramientas de monitoreo en los diferentes niveles.

- Configurar de manera segura los elementos y servicios de red.

1.3 MODELOS DE ADMINISTRACIÓN DE RED

Es la estandarización del empleo de una variedad de herramientas de red, aplicaciones y dispositivos para la administración de red. Para permitir que los componentes (de distintos fabricantes o proveedores) que conforman una red, y los sistemas operativos de los hosts puedan ínteroperar con el Sistema de Administración de Red.

Para la Administración de Red existen tres modelos fundamentales:

- Administración de Red OSI.
- Administración Internet.
- Arquitectura TMN (Telecommunications Management Network).

Administración de Red OSI. Definido por ISO, con el objetivo de lograr la administración de los recursos según el modelo de referencia OSI.

Administración Internet. Definido por la Fuerza de Tareas de Ingeniería de Internet IETF (Internet Enginnering Task Force) y la IAB (Internet Activities Board), para administrar según la arquitectura de red TCP/IP (Protocolo de Control de Transporte/ Protocolo de Internet, Transport Control Protocol / Internet Protocol).

Arquitectura TMN (Red de Administración de Telecomunicaciones). Definida por la ITU-T(Unión Internacional de Telecomunicaciones). Más que un modelo de red, define una estructura de red basada en los modelos anteriores.

Los modelos OSI e Internet se refieren a redes de hosts, mientras que el modelo TMN es de utilidad para los grandes operadores de redes de telecomunicaciones.

1.3.1 MODELO DE ADMINISTRACIÓN INTERNET

El Modelo de Administración Internet depende de la existencia en cada dispositivo de "Agentes SNMP Protocolo Simple de Administración de Red (Simple Network Management Protocol)", que principalmente se encargan de la recolección de la información sobre dicho dispositivo, ésta información se puede dividir en tres tipos:

- Información de estado
- Advertencias, y
- Alarmas.

La información que los agentes recogen, la envían a una aplicación central que controla el sistema. Esta se compone de una base de datos jerárquica o MIB (Management Information Base), por un lado, y una consola de administración, por el otro.

Para la gestión en Internet, el protocolo SNMP trabaja con otros componentes que cooperan con éste. Así, en el nivel superior, en la gestión se tiene:

- Estructura de Información de Gestión (SMI, Structure of Management Information)

- Base de Información de Gestión (MIB, Management Information Base).

SNMP utiliza los servicios ofrecidos por estos dos componentes para realizar su trabajo. Los componentes del modelo de Administración de Internet se muestran en la figura 1.1.

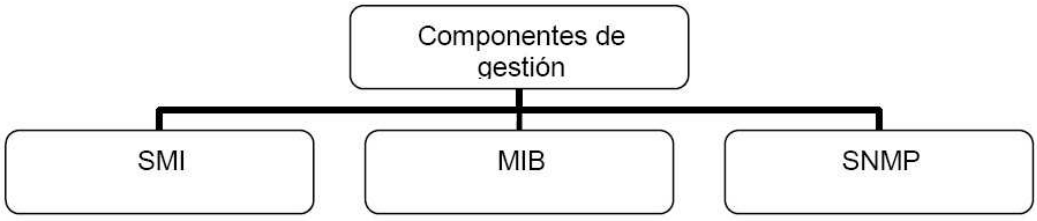


Figura 1.3.1.

1.3.2 ESTRUCTURA E IDENTIFICACIÓN DE LA INFORMACIÓN DE GESTIÓN SMI (STRUCTURE AND IDENTIFICATION OF MANAGEMENT INFORMATION)

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 ASN.1 (Abstract Syntax Notation 1, estándar ISO 8824), que es un lenguaje de descripción de datos.

1.3.3 OPERACIONES DE LA ADMINISTRACIÓN DE RED.

Las operaciones principales de un sistema de administración de red son las siguientes:

Administración de fallas.

La administración de fallas maneja las condiciones de error en todos los componentes de la red, en las siguientes fases:

a. Detección de fallas.
b. Diagnóstico del problema.
c. Darle la vuelta al problema y recuperación.
d. Resolución.
e. Seguimiento y control.

Control de fallas.

Esta operación tiene que ver con la configuración de la red (incluye dar de alta, baja y reconfigurar la red) y con el monitoreo continuo de todos sus elementos.

Seguridad.

La estructura administrativa de la red debe proveer mecanismos de seguridad apropiados para lo siguiente:

- Identificación y autentificación del usuario, una clave de acceso y un password.
- Autorización de acceso a los recursos, es decir, solo personal autorizado.
- Confidencialidad. Para asegurar la confidencialidad en el medio de comunicación y en los medios de almacenamiento, se utilizan medios de criptografía, tanto simétrica como asimétrica.

Un administrador de redes en general, se encarga principalmente de asegurar la correcta operación de la red, tomando acciones remotas o localmente. Se encarga de administrar cualquier equipo de telecomunicaciones de voz, datos y video, así como de administración remota de fallas, configuración rendimiento, seguridad e inventarios.

1.3.4 FUNCIONES DE ADMINISTRACIÓN DEFINIDAS POR OSI.

OSI define las cinco funciones de administración básicas siguientes:

- Configuración
- Fallas
- Contabilidad
- Comportamiento
- Seguridad.

La configuración comprende las funciones de monitoreo y mantenimiento del estado de la red.

La función de fallas incluye la detección, el aislamiento y la corrección de fallas en la red.

La función de contabilidad permite el establecimiento de cargos a usuarios por uso de los recursos de la red.

La función de comportamiento mantiene el comportamiento de la red en niveles aceptables para un buen funcionamiento de la red.

La función de seguridad provee mecanismos para autorización, control de acceso, confidencialidad y manejo de claves.

El modelo OSI incluye cinco componentes claves en la administración de red:

CMIS: Common Management Information Services. Éste es el servicio para la colección y transmisión de información de administración de red a las entidades de red que lo soliciten.

CMIP: Common Management Information Protocol. Es el protocolo de OSI que soporta a CMIS, y proporciona el servicio de petición/respuesta que hace posible el intercambio de información de administración de red entre aplicaciones.

SMIS: Specific Management Information Services. Define los servicios específicos de administración de red que se va a instalar, como configuración, fallas, contabilidad, comportamiento y seguridad.

MIB: Management Information Base. Define un modelo conceptual de la información requerida para tomar decisiones de administración de red. La información en el MIB incluye: número de paquetes transmitidos, número de conexiones intentadas, datos de contabilidad, etc...

1.3.5 ESQUEMA DE ADMINISTRACIÓN.

Como se observa, el agente y la MIB residen dentro del aparato que es monitoreado y controlado. La estación administradora contiene software que opera los protocolos usados para intercambiar datos con los agentes, y software de aplicación de administración de red que provee la interfaz de usuario para a fin de habilitar a un operador para saber el estado de la red , analizar los datos recopilados e invocar funciones de administración.

Figura 1.3.5(a)

El administrador de red controla un elemento de red pidiendo al agente del elemento que actualice los parámetros de configuración y que le de un informe sobre el estado de la MIB. El agente intercambia mensajes con el administrador de la red con el protocolo SNMP. Cualquier elemento que participe en la red puede ser administrado, incluidos host, ruteadores, concentradores, puentes, multiplexores, módems, switches de datos, etc... Cuando el aparato controlado no soporta SNMP, se usa un agente Proxy. El agente Proxy actúa como un intermediario entre la aplicación de administración de red y el aparato que no soporta SNMP.

Administración de un aparato que no soporta SMMP:

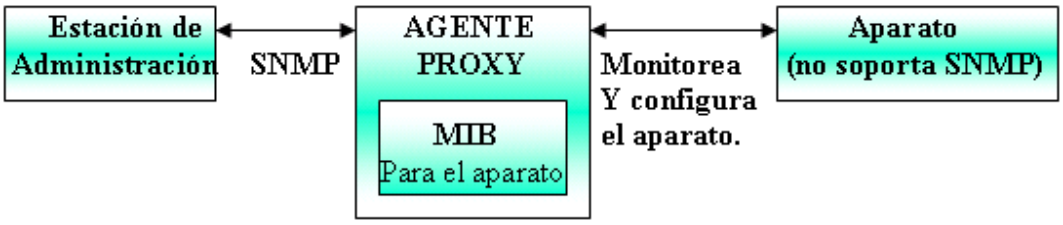


Figura 1.3.5(b)

CAPÍTULO 2

PROTOCOLOS Y COMANDOS DE RED

2.1 PROTOCOLO TCP/IP

Una red TCP/IP transfiere datos mediante el ensamblaje de bloques de datos en paquetes, cada paquete comienza con una cabecera que contiene información de control; tal como la dirección del destino, seguido de los datos. Cuando se envía un archivo por la red TCP/IP, su contenido se envía utilizando una serie de paquetes diferentes. El Internet protocol (IP), un protocolo de la capa de red, permite a las aplicaciones ejecutarse transparentemente sobre redes interconectadas. Cuando se utiliza IP, no es necesario conocer que hardware se utiliza, por tanto ésta corre en una red de área local.

2.1.1 DEFINICION TCP/IP

El nombre TCP / IP Proviene de dos protocolos importantes de la familia, el Transmission Contorl Protocol (TCP) y el Internet Protocol (IP). Todos juntos llegan a ser más de 100 protocolos diferentes definidos en este conjunto.

El TCP / IP es la base del Internet que sirve para enlazar computadoras que utilizan diferentes sistemas operativos, incluyendo PC, minicomputadoras y computadoras centrales sobre redes de área local y área extensa. TCP / IP fue desarrollado y demostrado por primera vez en 1972 por el departamento de defensa de los Estados Unidos, ejecutándolo en el ARPANET una red de área extensa del departamento de defensa.

2.1.2 FUNCIONALIDAD DE LAS CAPAS

Una vez que se toma la decisión de subdividir los problemas de comunicación en cuatro subproblemas y organizar el software de protocolo en módulos, de manera que cada uno maneja un problema, surge la pregunta. "¿Qué tipo de funciones deben instalarse en cada modulo?". La pregunta no es fácil de responder por varias razones. En primer lugar, un grupo de objetivos y condiciones determinan un problema de comunicación en particular, es posible elegir una organización que optimice un software de protocolos para ese problema. Segundo, incluso cuando se consideran los servicios generales al nivel de red, como un transporte confiable es posible seleccionar entre distintas maneras de resolver el problema. Tercero, el diseño de una arquitectura de red y la organización del software de protocolo esta interrelacionado; no se puede diseñar a uno sin considera al otro.

2.1.3 MODELO DE ESTRATIFICACION POR CAPAS DE TCP/IP DE INTERNET

El modelo de estratificación por capas no se origina de un comité de estándares, sino que proviene de las investigaciones que se realizan respecto al conjunto de protocolos de TCP/IP. Con un poco de esfuerzo, el modelo ISO puede ampliarse y describir el esquema de estratificación por capas del TCP/IP, pero los presupuestos subyacentes son lo suficientemente distintos para distinguirlos como dos diferentes.

En términos generales, el software TCP/IP está organizado en cuatro capas conceptuales que se construyen sobre una quinta capa de hardware. En la siguiente figura se muestra las capas conceptuales así como la forma en que los datos pasan entre ellas.

APLICACION
TRANSPORTE
INTERNET

CAPAS CONCEPTUALES PASO DE OBJETOS ENTR E CAPAS

Figura 2.1.3

- Capa de aplicación: Es el nivel mas alto, los usuarios llaman a una aplicación que acceda servicios disponibles a través de la red de redes TCP/IP. Una aplicación interactúa con uno de los protocolos de nivel de transporte para enviar o recibir datos. Cada programa de aplicación selecciona el tipo de transporte necesario, el cual puede ser una secuencia de mensajes individuales o un flujo continuo de octetos. El programa de aplicación pasa los datos en la forma requerida hacia el nivel de transporte para su entrega.

- Capa de transporte: La principal tarea de la capa de transporte es proporcionar la comunicación entre un programa de aplicación y otro. Este tipo de comunicación se conoce frecuentemente como comunicación punto a punto. La capa de transporte regula el flujo de información. Puede también proporcionar un transporte confiable, asegurando que los datos lleguen sin errores y en secuencia. Para hacer esto, el software de protocolo de transporte tiene el lado de recepción enviando acuses de recibo de retorno y la parte de envío retransmitiendo los paquetes perdidos. El software de transporte divide el flujo de datos que se está enviando en pequeños fragmentos (por lo general conocidos como paquetes) y pasa cada paquete, con una dirección de destino, hacia la siguiente capa de transmisión. Aun cuando en el esquema anterior se utiliza un solo bloque para representar la capa de aplicación, una computadora de propósito general puede tener varios programas de aplicación accesando a la red de redes al mismo tiempo. La capa de transporte debe aceptar datos desde varios programas de usuario y enviarlos a la capa del siguiente nivel. Para hacer esto, se añade información adicional a cada paquete, incluyendo

códigos que identifican qué programa de aplicación envía y qué programa debe recibir, así como una suma de verificación que se utiliza para verificar que el paquete ha llegado intacto y utiliza el código de destino para identificar el programa de aplicación en el que se debe entregar.

- Capa Internet: La capa Internet maneja la comunicación de una máquina a otra. Ésta acepta una solicitud para enviar un paquete desde la capa de transporte, junto con una identificación de la máquina, hacia la que se debe enviar el paquete. La capa Internet también maneja la entrada de datagramas, verifica su validez y utiliza un algoritmo de ruteo para decidir si el datagrama debe procesarse de manera local o debe ser transmitido. Para el caso de los datagramas direccionados hacia la máquina local, el software de la capa de red de redes borra el encabezado del datagrama y selecciona, de entre varios protocolos de transporte, un protocolo con el que manejará el paquete. Por último, la capa Internet envía los mensajes ICMP de error y control necesarios y maneja todos los mensajes ICMP entrantes.

- Capa de interfaz de red: El software TCP/IP de nivel inferior consta de una capa de interfaz de red responsable de aceptar los datagramas IP y transmitirlos hacia una red específica. Una interfaz de red puede consistir en un dispositivo controlador (por ejemplo, cuando la red es una red de área local a la que las máquinas están conectadas directamente) o un complejo subsistema que utiliza un protocolo de enlace de datos propios (por ejemplo, cuando la red consiste de conmutadores de paquetes que se comunican con anfitriones utilizando HDLC).

- Capa de Hadware: La capa de hardware es un elemento del sistema operativo que funciona como una interfaz entre el software y el hardware del sistema, proveyendo una plataforma de hardware consistente sobre la cual correr las aplicaciones. Cuando se emplea una HAL, las aplicaciones no acceden directamente al hardware sino que lo hacen a la capa abstracta provista por la HAL. Del mismo modo que las API, las HAL permiten que las aplicaciones sean independientes del hardware porque abstraen información acerca de tales sistemas, como lo son las cachés, los buses de E/S y las interrupciones, y usan estos datos para darle al software una forma de interactuar con los requerimientos específicos del hardware sobre el que deba correr.

2.1.4 ESTRATIFICACION POR CAPAS EN PRESENCIA DE UNA SUBESTRUCTURA DE RED

Desde la perspectiva del IP, el conjunto de conexiones punto a punto entre ruteadores puede funcionar como un conjunto de redes físicas independientes o funcionar colectivamente como una sola red física.

En el primer caso, cada enlace físico es tratado exactamente como cualquier otra red en una red de redes. A esta se le asigna un número único de red (por lo general de clase C) y los dos anfitriones que comparten el enlace tiene cada uno una dirección única IP asignada para su conexión.

Los ruteadores se añaden a una tabla de ruteo IP como lo harían para cualquier otra red. Un nuevo modulo de software se añade en la capa de interfaz de red para controlar el nuevo enlace de hardware, pero no se realizan cambios sustanciales en el esquema de estratificación por capas. La principal desventaja del enfoque de redes independientes es la proliferación de números de redes (uno por cada conexión entre dos maquinas), lo que ocasiona que las tablas de ruteo sean tan grandes como sea necesario. Tanto la línea serial IP (Serial Line IP o SLIP) como el protocolo punto a punto (Point to Point Protocol o PPP) tratan a cada enlace serial como una red separada.

El segundo método para ajustar las conexiones punto a punto evita asignar múltiples direcciones IP al cableado físico. En lugar de ello, se tratan a todas las conexiones colectivamente como una sola red independiente IP con su propio formato de trama, esquema de direccionamiento de hardware y protocolos de enlace de datos. Los ruteadores que emplean el segundo método necesitan solo un numero de red IP para todas las conexiones punto a punto.

Usar el enfoque de una sola red significa extender el esquema de estratificación por capas de protocolos para añadir una nueva capa de ruteo dentro de la red, entre la capa de interfaz de red y los dispositivos de hardware. Para las máquinas con una sola conexión punto a punto, una capa adicional parece innecesaria.

Las diferencias entre una tabla de ruteo de red de redes y una tabla de ruteo dentro de la red son que esta ultima, es mucho más pequeña. Contiene solamente información de ruteo para los anfitriones conectados directamente a la red punto a punto. La razón es simple: la capa Internet realiza la transformación de una dirección de destino arbitraria hacia una ruta de dirección especifica antes de pasar el datagrama hacia una interfaz de red. De esta manera, la capa dentro de la red solo debe distinguir entre máquinas en una sola red unto a punto.

2.2 PROTOCOLO SIMPLE DE RED

SNMP significa Protocolo simple de administración de red . Es un protocolo que les permite a los administradores de red administrar dispositivos de red y diagnosticar problemas en la red.

2.2.1. PROTOCOLO SIMPLE DE GESTIÓN DE RED SNMP (SIMPLE NETWORK MANAGEMENT PROTOCOL) SNMP

El SNMP es un protocolo de capa de aplicación que facilita la administración y el manejo de la información entre dispositivos de red. Es una parte del conjunto del Protocolo de Control de Transmisión/Protocolo de Internet (TCP/IP Transmission Control Protocol/Internet Protocol)

SNMP, en sus distintas versiones, es un conjunto de aplicaciones de gestión de red que emplea los servicios ofrecidos por TCP/IP y que ha llegado a convertirse en un estándar. Surge a raíz del interés por encontrar un protocolo de gestión que fuese válido para la red Internet, dada la necesidad del mismo a causa de la gran dimensión que estaba tomando. Para el protocolo SNMP la red constituye un conjunto de elementos básicos: Administradores o Gestores (Network Management Stations) ubicados en los equipos de gestión de red y Agentes (elementos pasivos ubicados en los host, routers, multiplexores, módems, etc. a ser gestionados), siendo los segundos los que envían información a los primeros, relativa a los elementos gestionados, bien al ser interrogados o de manera secuencial.

2.2.1.2.1 PRINCIPIO OPERATIVO DEL SNMP

El sistema de administración de red se basa en dos elementos principales: un supervisor y agentes. El supervisor es el terminal que le permite al administrador de red realizar solicitudes de administración. Los agentes son entidades que se encuentran al nivel de cada interfaz. Ellos conectan a la red los dispositivos administrados y permiten recopilar información sobre los diferentes objetos.

La arquitectura de administración de la red propuesta por el protocolo SNMP se basa en tres elementos principales:

- Los dispositivos administrados son los elementos de red (puentes, concentradores, routers o servidores) que contienen "objetos administrados" que pueden ser información de hardware, elementos de configuración o información estadística.

- Los agentes, es decir, una aplicación de administración de red que se encuentra en un periférico y que es responsable de la transmisión de datos de administración local desde el periférico en formato SNMP.

- El sistema de administración de red (NMS), esto es, un terminal a través del cual los administradores pueden llevar a cabo tareas de administración

Obtención de la Información:

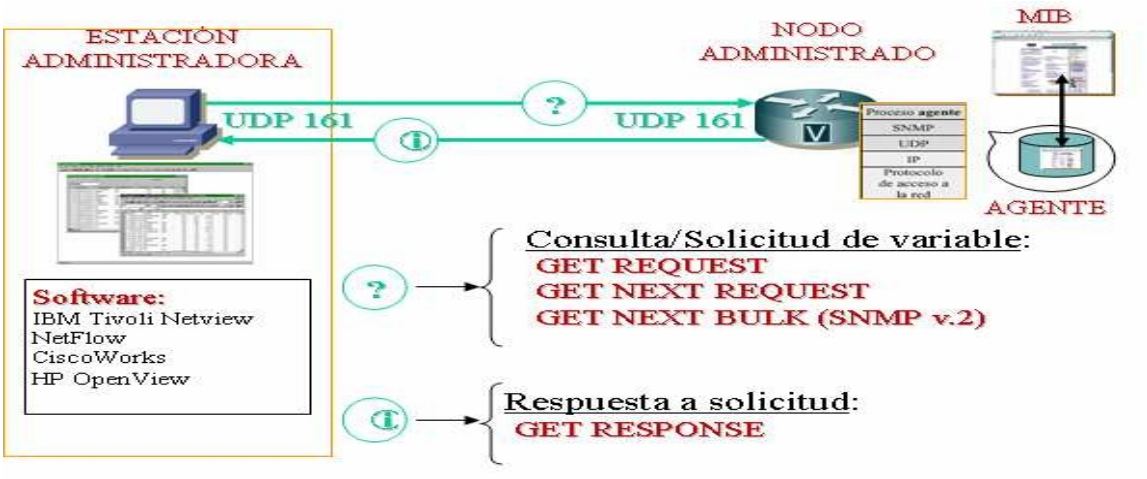


Figura 2.2.1.2.1(a)

Modificación de Información:

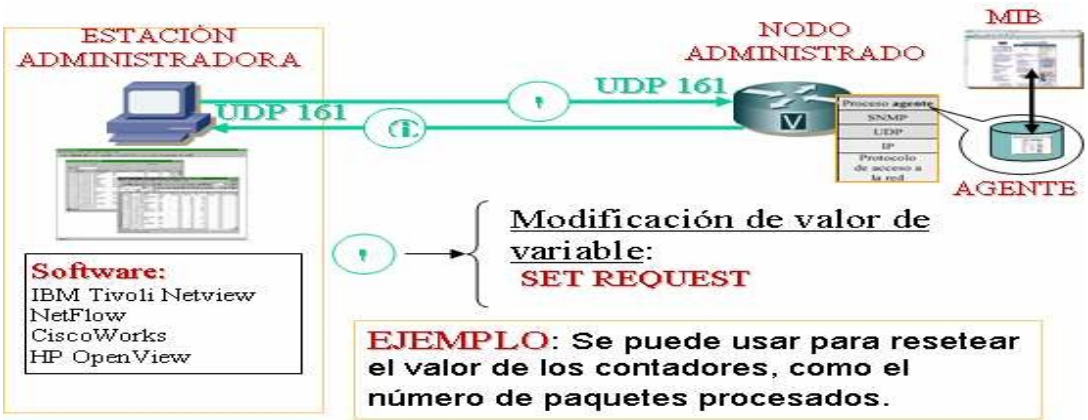


Figura 2.2.1.2.1(b)

2.2.1.2.2 GESTORES Y AGENTES

La gestión de red se lleva a cabo mediante una aplicación software residente en el computador designado como Gestor de la red que, mediante una interface de operador, permite la gestión, y otras residentes en cada uno de los elementos que conforman la estructura de la red, es decir nodos y medios de transmisión. El software de gestión responde a los comandos del operador de red, enviando información a los elementos de la red y/o recibiendo información de ellos.

En la gestión de red se identifican cinco áreas funcionales que son:

Gestión de fallos, para facilitar la detección, aislamiento y corrección de las incidencias que se produzcan en la red, controlando cualquier funcionamiento que se salga de los márgenes de tolerancia fijados por el administrador de ella. Lo normal es que al producirse un fallo se genere una alarma que indique la causa y el lugar del mismo, alertando al personal encargado de la gestión, que actuará en consecuencia.

Gestión de la configuración, para realizar las labores rutinarias de cambios en los parámetros de funcionamiento de los elementos que configuran la red, mantener el inventario de todos los elementos que conforman la red, realizar altas y bajas de usuarios y asegurar que el tráfico se mantiene conforme a lo planificado. Así, en caso de caída de algún enlace, se puede establecer un camino alternativo hasta que se restablezcan las condiciones iníciales.

Gestión del rendimiento, que incluye todas las funciones necesarias para evaluar el comportamiento de los objetos gestionados y de la red en su conjunto, incluidos los medios de transmisión. En base al resultado se determina la carga real de tráfico (throughput) , la disponibilidad y el tiempo de respuesta, y se puede prever la congestión de determinados nodos o rutas, adelantándose a llegue a suceder y que la demanda de los usuarios se vea insatisfecha.

Gestión de la tarificación, que engloba las funciones relativas a la administración de los recursos de la red y el cargo que por su uso hay que hacer a los usuarios. Permite distribuir los costos, generando las facturas para los distintos departamentos de la empresa.

Gestión de la seguridad, uno de los aspectos más críticos en la gestión de una red corporativa, esencial para mantener la integridad y confidencialidad de los datos, protegiendo frente a la intrusión por terceros. Con la adopción de Internet como medio de comunicación global y la implantación de su tecnología en las empresas para la creación de Intranets, el aislamiento entre el entorno corporativo y el mundo exterior se ha de conseguir a base de establecer cortafuegos y claves de acceso, que han de estar integrados en el sistema de gestión de red.

AGENTE : El agente almacena información de sistema, de interfaces, etc, en una base de información de gestión, mientras que el gestor tiene acceso a los valores de esta base.
Los agentes también pueden enviar al gestor mensajes de advertencia (denominados *traps*), si el programa que se ejecuta en el agente detecta algún error en el entorno.

2.2.1.2.3 ARQUITECTURA DE SNMP

Se muestra a continuación los componentes del protocolo SNMP:

- Una estación de gestión.
- Un agente de gestión (incluidos agentes proxy).
- Una base de información de gestión (MIB).
- Protocolo de gestión de red.

Los elementos de la estación de gestión son los siguientes:

- Aplicaciones (para análisis de datos, etc.)

- Interfaz de usuario.
- Capacidad de convertir las solicitudes del usuario a peticiones de Monitoreo.
- Control a los elementos remotos y base de datos con información de las MIBs de los elementos de la red gestionados.

En la figura se indica una arquitectura común de Administración de Red.

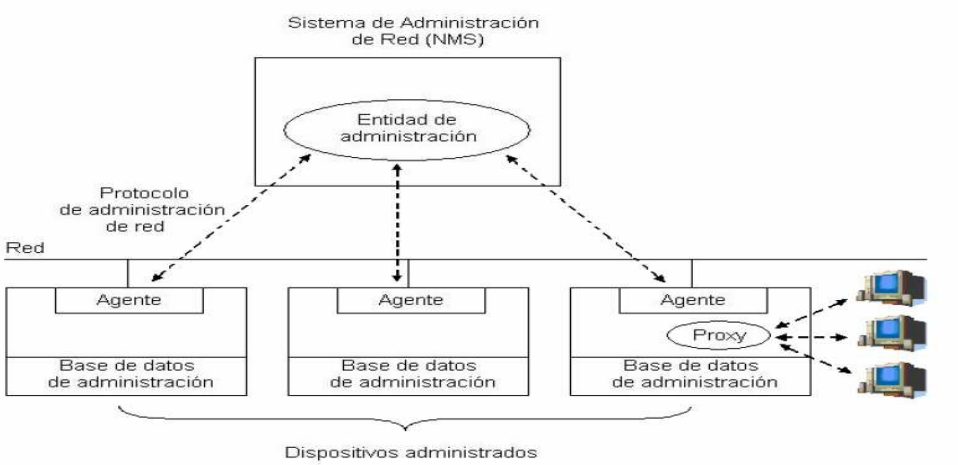


Figura 2.2.1.2.3

2.2.1.2.4 TIPOS DE MENSAJES SNMP

En la figura se muestran los diferentes tipos de mensajes que intercambian el gestor y el agente SNMP.

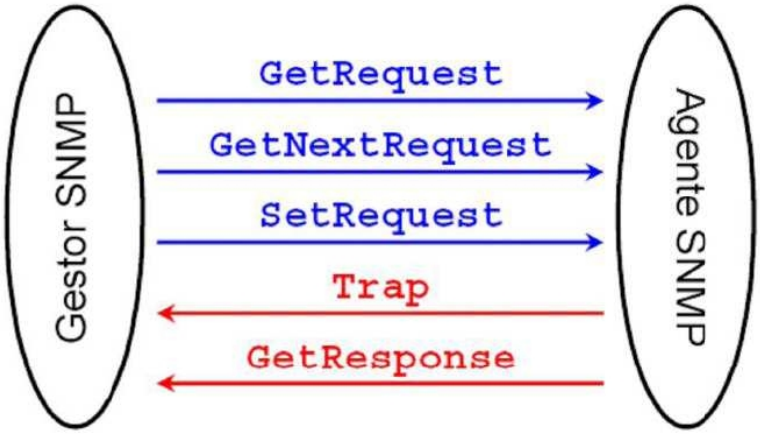


Figura 2.2.1.2.4

Los diferentes tipos de mensajes se describen a continuación:

- **GetRequest:** el gestor pide al agente el valor de un dato.
- **GetNextRequest** es similar al *GetRequest*, permitiendo extraer datos de una tabla.
- **SetRequest:** el gestor pide al agente que modifique los valores de las variables que especifique. El agente modificará todos o ninguno de los valores.
- **GetResponse:** Respuesta del agente a las peticiones *GetRequest, GetNextRequest* y *SetRequest*.
- **Trap**: Mensaje generado por el agente en respuesta a un evento que afecte a la MIB o a los recursos gestionados. El gestor no confirma la recepción de un *trap* al agente.

2.2.1.2.5 COMUNIDADES SNMP

A continuación se describen las diferentes características que deben cumplir los agentes y comunidades SNMP:

- Cada agente es responsable de su MIB local, controlando sus estaciones gestoras.

- Control de acceso a la MIB de un agente: concepto de comunidad.
- Comunidad es la relación que se tienen entre un agente SNMP y un conjunto de estaciones de gestión SNMP, definen unas características de autentificación y control de acceso.
- El agente establece una comunidad para cada combinación deseada de autentificación y control de acceso, y a cada comunidad se le da un nombre de comunidad (community name) que es su nombre único dentro del agente. Este nombre las estaciones de gestión pertenecientes a una comunidad la emplean en todas las operaciones *GetRequest*, *GetNextRequest* y *SetRequest*.

- El agente puede establecer cualquier número de comunidades. Y a su vez una estación de gestión puede pertenecer a varias comunidades.

- Una estación de gestión debe almacenar los nombres de comunidad asociados a cada agente.

- Mediante el uso de comunidades, un agente puede limitar el acceso a su MIB en dos formas:
  - Vistas de la MIB: subconjunto de los objetos de la MIB.
  - Modos de Acceso: solo lectura o solo escritura.

- El agente sólo atiende la petición si el nombre de la comunidad es correcto para el tipo de acceso solicitado.

2.2.3 TIPOS DE VERSIONES DE SNMP

El Protocolo SNMP existen 3 versiones que han ido evolucionando con el pasar del tiempo en lo que respecta a seguridad. Estas se detallan a continuación:

2.2.3.1 PROTOCOLO SIMPLE DE GESTIÓN DE RED VERSIÓN 1 (SNMPv.1)

En el año de 1990 surge un nuevo estándar llamado: SNMP (Simple Network Managenement Protocol, Protocolo Simple de Administración de Redes), definido en el RFC 1157. Este protocolo muestra una manera de administrar y supervisar las redes de computo para identificar y resolver problemas, así como para planear su crecimiento. Se encuentra implementado en la capa de aplicación y pertenece al grupo de protocolos de TCP/IP.

La seguridad se basa en comunidades (que usan passwords comunes sobre texto plano) que permiten usar dispositivos si se conoce el password. Se puede explotar por la fuerza bruta

Formato de mensajes SNMPv1

Para realizar las operaciones básicas de gestión de red, SNMP usa un conjunto de mensajes que son intercambiados entre el gestor y el agente. Es importante mencionar que estas operaciones se ejecutan de manera atómica, es decir se ejecuta todo o nada. Estos mensajes SNMP se estructuran con el formato que se muestra en la Figura 2.2.3.1(a)

Figura 2.2.3.1(a)

Traps: Las traps son notificaciones que emite el agente SNMP ante el acontecimiento de un evento particular, aún cuando el gestor no lo haya solicitado. Las traps manejan un formato de PDU diferente

Figura 2.2.3.1(b)

Enterprise: Identifica al tipo de objeto que emite la notificación.

Dirección del agente: Dirección de red del agente que emite la notificación.

Trap Genérico: Indica el tipo de trap generado, pudiendo ser una de las opciones que se detalla en la Tabla

Tipo	Descripción
(0) coldStart	Reinicio del agente con posibles cambios de configuración.
(1) warmStart	Reinicio del agente sin cambios de configuración.
(2) linkDown	Enlace caído, no disponible.
(3) linkUp	Restablecimiento de un enlace.
(4) authenticationFailure	Autenticación agente-gestor fallida.
(5) egpNeighborLoss	Usado cuando un router que maneja EGP pierde conexión con otro router EGP vecino.
(6) enterpriseSpecific	Trap que no coincide con las opciones anteriores.

Tabla 2.2.3.1(c)

Trap específico: Usado para emitir traps privados, y en ocasiones para precisar información de las traps genéricas.

Marca de tiempo: Tiempo transcurrido desde el instante en que inicia el agente hasta que emite la trap.

Información adicional: Acerca de la trap suscitada, por ejemplo cuando un enlace falla se emite la trap linkDown, junto con el nombre y número de la interfaz en la que ocurrió la falla como información adicional.

2.2.3.2 PROTOCOLO SIMPLE DE GESTIÓN DE RED VERSIÓN 2 (SNMPv.2)

Como resultado de una serie de propuestas para mejorar las características de SNMP, en 1996 se publica un nuevo estándar, el protocolo SNMPv.2. Los cambios se traducen fundamentalmente en una mejora de las prestaciones de intercambio de información de gestión, como la implementación de seguridad que fue una de las principales limitaciones de SNMPv.1.

Reduce la carga de tráfico adicional para la monitorización (con uso de GetBulk e Informs) y soluciona los problemas de monitorización remota o distribuida (con las sondas RMON). SNMPv2 puede leer SNMPv1.

Formato de mensajes SNMPv2

De la misma manera que la versión 1 se encuentra distribuida de la siguiente manera en la parte de versión va la versión implementada y cambia el PDU

Figura 2.2.3.2(a)

Versión: Un número que indica la versión del protocolo SNMP, 0 para SNMPv1, 1 para SNMPv2 y 3 para la SNMPv3.

Trap de SNMPv2 Cambia su formato respecto a la versión 1, tal como se muestra en la Figura.

Figura 2.2.3.2(b)

Identificador de solicitud: Es un número que sirve para asociar una respuesta a la petición adecuada. Esto evita problemas de duplicidad de PDU.

Estado de error: Indica que ha ocurrido un error con uno de los valores solicitados. En este caso al tratarse de una petición siempre irá el valor 0.

Índice de error: En este subcampo se indica el índice de la variable que produjo un error.

Tipo	Descripción
(0) noError	No existe error.
(1) tooBig	Respuesta demasiado larga.
(2) noSuchName	No se puede hallar el valor del OID solicitado.
(3) badValue	El valor enviado por SetRequest no coincide con el tipo, longitud o variable.
(4) readOnly	El valor que se está intentando modificar es de solo lectura.
(5) genErr	Error genérico que no coincide con los anteriores.

Tabla 2.2.3.2(c)

OID1, OID2,…OIDn: En este subcampo se envía los identificadores de objetos que se están solicitando.

2.2.3.3 PROTOCOLO SIMPLE DE GESTIÓN DE RED VERSIÓN 3 (SNMPv.3).

En 1998, a partir de estas dos versiones anteriores, surge SNMPv.3 con las siguientes características:

- SNMPv.3 no modifica las PDUs de SNMPv.2
- Define una serie de capacidades de seguridad y un marco que hace posible su uso junto con las PDUs de SNMPv.2
- SNMPv.3 es SNMPv.2 con mayor seguridad y administración

Para evitar la falta de seguridad en las transmisiones lo realiza (con cifrado y autenticación), y proporciona una capa o parche complemento a SNMPv1 y v2, que añade a los mensajes SNMP (v1 y v2) una cabecera adicional.

Formato de mensaje SNMPv3.

El formato de un mensaje SNMPv3 cambia respecto a las dos versiones anteriores en virtud de que debe involucrar parámetros para el proceso de autenticación y encriptación

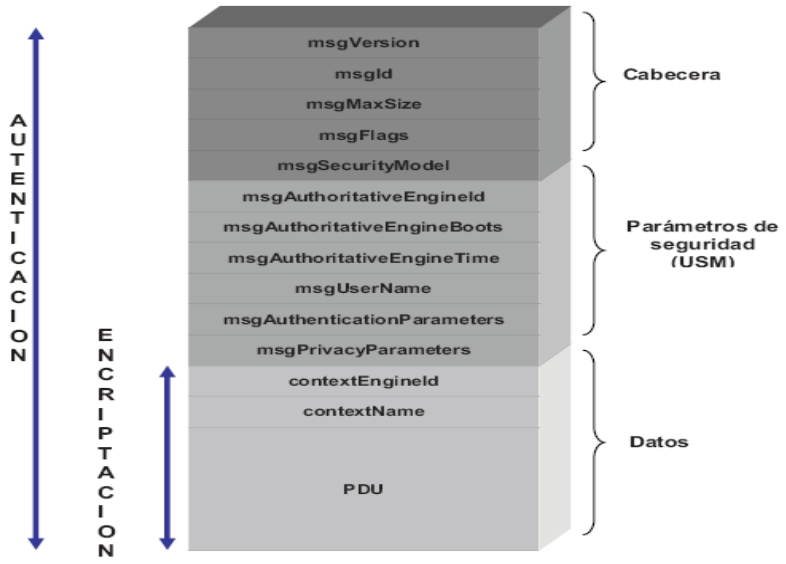


Figura 2.2.3.3

- **msgVersion:** Indica la versión de SNMP.

- **msgId:** Número de 32 bits que sirve para relacionar las peticiones con las respuestas.

- **msgMaxSize**: Número de 32 bits que indica la cantidad en bytes que puede recibir el emisor del mensaje.

- **msgFlags** : Número de 8 bits, pero solo usa los 3 bits menos significativos para indicar el nivel de seguridad a emplear:

  reportableFlag: El valor 1 en este subcampo indica que el receptor del mensaje debe enviar de vuelta un acuse de recibo.

  privFlag: El valor 1 indica que se debe encriptar el mensaje.

  authFlag: Cuando se asigna el valor 1 se debe aplicar autenticación al mensaje.

- **msgSecurityModel:** Indica el modelo de seguridad empleado para la emisión del mensaje. SNMPv1 (1), SNMPv2 (2) y USM de SNMPv3 (3).

La siguiente sección contiene parámetros exclusivos para la operación de USM:

- **msgAutoritativeEngineID:** Es un identificador que se asigna al motor de una entidad SNMP (agente o gestor) que responde a las peticiones o al que recibe las notificaciones, y que es el que sirve como referencia para el control de la sincronización entre agente y gestor. A este motor se lo denomina Motor Autorizado (AutoritativeEngine).

- **msgAuthoritativeEngineBoots:** Indica el número de ocasiones que un motor SNMP reinició desde su configuración original.

- **msgAutoritativeEngineTime**: Indica el tiempo en segundos desde que inició por última vez.

2.2.3.4 INTEROPERABILIDAD CON SNMPV.1

La versión 2 de SNMP permite ínteroperar con SNMPv.1, bajo las siguientes características:

- Los agentes pueden *emplear* tanto la versión 1 como la 2.

- Los gestores deben *emplear* tanto la versión 2 como la 1 para poder comunicarse con agentes de versión 1.

- El gestor intentará primero *emplear* la versión 2 con un agente, si:
  - El agente *emplea* versión 2 le contestará.
  - El agente *emplea* versión 1 responderá con un mensaje de error; el gestor entonces, lo intentará con versión 1, para ello deberá hacer conversiones (por ejemplo, convertir *getbulk* en *getnext).*

2.2.3.5 SEGURIDAD DE SNMP VERSION 2

SNMPv.2 define métodos para controlar las operaciones que están permitidas, a diferencia de SNMPv.1 que no incorpora ningún mecanismo de seguridad.

Pero luego, surgieron dos planteamientos diferentes en cuanto al modelo de seguridad, que han dado lugar a dos especificaciones conocidas como SNMPv.2

La versión SNMPv.2 proporciona niveles de seguridad adecuados, pero no alcanzó el necesario nivel de estandarización y aceptación por el IETF y SNMPv.2u La identificación se lleva a cabo mediante usuarios. Es decir que un mismo usuario puede estar definido en varias entidades SNMP diferentes).

2.2.3.6 PRINCIPALES CARACTERÍSTICAS DE SNMPV.2

Ésta versión ya permite el soporte de comunicación gestor-gestor (PDU-Protocol Data Unit- *informrequest*). La lectura de tablas completas en una sola operación (PDU *getbulkrequest*), es otra de las variantes. Pero la seguridad sigue siendo una falencia, ya que todavía se basa en “comunidades” (un nombre con un conjunto de operaciones permitidas) hasta la versión 3.

Los diferentes tipos de mensajes que se manejan en SNMPv.2 son los siguientes:

- get-request
- get-next-request
- get-bulk-request
- response
- set-request
- inform-request
- snmpV2-trap
- report (aparecía en los documentos de seguridad y después se eliminó junto con ellos).

2.2.3.7 ARQUITECTURA DE GESTIÓN DE RED EN SNMPV.3

SNMPv.3 define una arquitectura de gestión de red:

- Colección de entidades SNMP que interaccionan entre sí.
  - 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.

- Una entidad incluye un motor SNMP, este motor implementa funciones para enviar y recibir mensajes, autentificar y encriptar/desencriptar mensajes, y controlar el acceso a los objetos gestionados.

2.2.3.8 MEJORAS EN LA SEGURIDAD EN SNMPV.3

SNMPv.3 debió definir un conjunto de mecanismos de seguridad que incluyen la protección contra las amenazas de:

- Modificación de la información.
- Enmascaramiento.
- Modificación del flujo de mensajes.
- Revelación de contenidos.

Pero no se contempla la protección frente a:

- Denegación de servicio y Análisis de tráfico.

2.3 TERMINOS Y CONCEPTOS DE SNMP

2.3.1 ASN.1

ASN.1 es una norma para representar datos independientemente de la máquina que se esté usando y sus formas de representación internas. Es un protocolo de nivel de presentación en el modelo OSI.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 está diseñado para definir información estructurada (mensajes) de tal forma que sea independiente de la máquina utilizada. Para hacer esto ASN.1 define tipos de datos básicos, como enteros y strings, y permite construir nuevos tipos de datos a partir de los ya definidos. También utiliza palabras especiales (keywords) para para

definir sus procedimientos, definir nuevos tipos, asignar valores, definir macros y módulos. Aquí se presentarán algunos de ellos.

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

Tabla 2.3.1(a)

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

Tabla 2.3.1(b)

ASN.1 define tres tipos de datos generales, los tipos que se mencionan a continuación son enfocados en la gestión de red de Internet:

a) Simples o primitivos: Conocidos también como registros escalares, puesto que almacenan un único valor. Los tipos primitivos más importantes se muestran en la Tabla.

Tipo		Descripción
Integer		Número entero positivo o negativo de hasta 32 bits.
Octet String	Display String	Cadena de caracteres ASCII imprimibles.
	OctetBitString	Usado para cadenas de bits mayores a 32 bits.
	PhysAddress	Representan direcciones de la capa de enlace.
Object Identifier		Identificador de objeto, que marca la posición en la MIB.
Null		Representa la ausencia de valor.
Bolean		Representa un valor que puede ser verdadero o falso.

Tabla 2.3.1(c)

b) Estructurados o compuestos: Son registros vectoriales, puesto que sirve para definir filas y tablas. Son construidos a partir de otros tipos primitivos o compuestos. En la tabla se detalla los tipos de datos estructurados.

Tipo	Descripción
Sequence	Representa una fila, es decir una lista ordenada de tipos de datos diferentes. Son construidos a partir de tipos primitivos.
Sequence Of	Representa una tabla, es decir es una lista ordenada de varias filas iguales. Son construidos a partir de tipos compuestos.
Set	Es un tipo de dato similar a sequence, con la diferencia que la lista no está ordenada.
Set Of	Es un tipo de dato similar a sequence of, con la diferencia que la lista no está ordenada.
Choice	Es un tipo de dato en el que se debe escoger entre una lista previamente definida.

Tabla 2.3.1(d)

c) Definidos: Se construyen a partir de los tipos de datos anteriores (primitivos y compuestos) con la diferencia de que se les ha definido un nombre más descriptivo, se utiliza para distinguir los tipos dentro de una aplicación. Los tipos de datos definidos o etiquetados se detallan en la tabla

Tipo	Descripción
Network Address	Representa una dirección de red de cualquier familia de protocolos.
IpAddress	Representa la dirección de red de Internet (32 bits), definida en la pila de protocolos TCP/IP.
Counter	Representa un entero (Integer) positivo, el cual se incrementa monótonamente hasta alcanzar 2^{32}-1. Se reinicia cuando alcanza el valor máximo.
Gauge	Representa un entero (Integer) positivo, el cual se incrementa o decrementa monótonamente. Se reinicia cuando alcanza el valor máximo.
Time Ticks	Representa un entero (Integer) positivo, el cual cuenta el tiempo transcurrido en centésimas de segundo.
Opaque	Representa un Octet String al que se le puede pasar cualquier valor ASN.1.

Tabla 2.3.1(e)

2.3.2 BER

Las Reglas de Codificación Básicas son las reglas definidas originalmente en el estándar ASN.1 para codificar información abstracta en un flujo de bits único, esto es, que pueda ser interpretado en cualquier máquina de la misma manera. Las reglas, denominadas sintaxis de transferencia en el contexto de ASN.1, especifican las secuencias de octetos exactas para codificar un elemento de datos dado. La sintaxis define 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 (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.

2.3.3 PDU

PDUs (en inglés, Protocol Data Units), Unidades de Datos de Protocolo. Se utiliza para el intercambio entre unidades parejas, dentro de una capa del modelo OSI. Existen dos clases de PDUs:

- PDU de datos, que contiene los datos del usuario final (en el caso de la capa de aplicación) o la PDU del nivel inmediatamente superior.

- PDU de control, que sirven para gobernar el comportamiento completo del protocolo en sus funciones de establecimiento y ruptura de la conexión, control de flujo, control de errores, etc. No contienen información alguna proveniente del nivel N+1.

Cada capa del modelo OSI en el origen debe comunicarse con capa igual en el lugar destino. Esta forma de comunicación se conoce como comunicación de par-a-par.

Durante este proceso, cada protocolo de capa intercambia información en lo que se conoce como unidades de datos de protocolo (PDU), entre capas iguales. Cada capa de comunicación, en el computador origen, se comunica con un PDU específico de capa y con su capa igual en el computador destino.

2.3.4 SMI

En la computación, el Structure of Management Information (SMI), un subconjunto adaptado de la notación ASN.1, funciona en el Simple Network Management Protocol (SNMP) para definir los sistemas (“módulos”) de objetos manejados relacionados en un Management Information Base (MIB).

El SMI se subdivide en tres porciones: definiciones del módulo, definiciones del objeto, y definiciones de la notificación.

- Se utilizan las definiciones del módulo al describir los módulos de la información. Una macro de ASN .1, MODULE-IDENTITY, se utiliza para transportar sucinto la semántica de un módulo de la información.

    - Las definiciones del objeto describen objetos manejados. ASN.1 una macro, OBJECT-TYPE, se utiliza para transportar sucinto la sintaxis y la semántica de un objeto manejado.

- Se utilizan las definiciones de la notificación (el aka “atrapa”) al describir transmisiones no solicitadas de la información de gerencia. ASN.1 una macro, NOTIFICATION-TYPE, transporta sucinto el sintaxis y la semántica de una notificación.

2.3.5 NMS

(Network Management Station) o Estación de Gestión de la Red, es una combinación hardware y software y es la interfaz para la administración de la red.

2.4 COMANDOS BASICOS DEL PROTOCOLO SNMP

2.4.1 SNMPGET

La órden snmpget se puede utilizar para obtener datos de un host remoto dado su nombre de host, la información y un OID. Como ejemplo veamos la siguiente órden:

```
# snmpget -c public -v 2c localhost system.sysUpTime.0
SNMPv2-MIB::sysUpTime.0 = Timeticks: (326510) 0:54:25.10
```

En la orden anterior, localhost es el nombre del equipo con el que queremos hablar, utilizando la comunidadSNMP public y solicitamos el valor del OID system.sysUpTime.0.

Todas las utilidades que vamos a ver permiten utilizar abreviaturas de OID y realizan búsquedas de la porción pasada como argumento en todo el árbol por defecto para facilitar la tarea al usuario. De este modo podemos utilizar simplemente una porción del OID si así lo deseáramos:

```
# snmpget -c public -v 2c localhost sysUpTime.0
SNMPv2-MIB::sysUpTime.0 = Timeticks: (326510) 0:54:25.10
```

Un error habitual cuando utilizamos esta orden es olvidar poner el índice en los datos que estamos buscando. En las órdenes anteriores, la variable solicitada es un escalar y el índice de los datos escalares es siempre un simple 0, de ahí el '.0' en todos los OIDs anteriores. Si lo hubiéramos olvidado hubiéramos obtenido un error que es diferente en función de si estamos utilizando una versión u otra de SNMP:

```
# snmpget -c public -v 1 localhost system.sysUpTime
Error in packet
Reason: (noSuchName) There is no such variable name in this MIB.
Failed object: SNMPv2-MIB::sysUpTime
```

```
# snmpget -c public -v 2c localhost system.sysUpTime
SNMPv2-MIB::sysUpTime = No Such Instance currently exists at this OID
```

Se pueden realizar múltiples consultas en una única órden:

```
snmpget -c public -v 2c localhost system.sysUpTime.0 system.sysName.0
SNMPv2-MIB::sysUpTime.0 = Timeticks: (390447) 1:05:04.47
SNMPv2-MIB::sysName.0 = STRING: mago.aut.uah.es
```

2.4.2 SNMPTRANSLATE

Esta orden es una herramiente muy poderosa que permite explorar el árbol MIB de Diversas formas desde la línea de órdenes.

Su forma más básica permite pasar de un OID a la variable que representa (representación textual):

```
#snmptranslate .1.3.6.1.2.1.1.3.0
SNMPv2-MIB::sysUpTime.0
```

También permite pasar del nombre de la variable al OID que representa:
`#snmptranslate -On SNMPv2-MIB::system.sysUpTime.0 .1.3.6.1.2.1.1.3.0`

La variable que pasamos como argumento a snmptranslate se puede expresar en cualquier formato numérico, texto o una mezcla de ambos. El _ag -On simplemente sirve para indicar que queremos la salida en formato de OID:

```
#snmptranslate .iso.3.6.1.private.enterprises.2021.2.1.prNames.0
UCD-SNMP-MIB::prNames.0
#snmptranslate -On .iso.3.6.1.private.enterprises.2021.2.1.prNames.0
.1.3.6.1.4.1.2021.2.1.2.0
```

2.4.3 SNMPGETNEXT

Esta orden es similar al comando snmpget se utiliza para obtenet 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:

SNMPv2-MIB::sysContact.0=STRING:Root<root@localhost>(configure /etc/snmp/snmp.local.conf)

Esta órden también se puede utilizar para recorrer de forma manual el árbol de las mib en un host remoto, especi_cando siempre el último OID aparecido en la línea de órdenes para la siguiente órden:
snmpgetnext -v 2c -c public localhost system.sysUpTime.0
SNMPv2-MIB::sysContact.0 = STRING: Root <root@localhost>(configure /etc/snmp/snmp.local.conf)
[root@mago root]# snmpgetnext -v 2c -c public localhost system.sysContact.0
SNMPv2-MIB::sysName.0 = STRING: mago.aut.uah.es
[root@mago root]# snmpgetnext -v 2c -c public localhost system.sysName.0
SNMPv2-MIB::sysLocation.0 = STRING: Unknown (edit /etc/snmp/snmpd.conf)

2.4.4 SNMPWALK

La función snmpwalk() es usada para leer todos los valores de un agente SNMP especificado por el hostname. Community especifica la comunidad lectora para el agente. Un object_id nulo se toma como la raíz del arbol de los objetos SNMP y todos los objetos por debajo de ese arbol son devueltos como una matriz. Si object_id es especificado, todos los objetos SNMP por debajo de object_id son devueltos.

Esta órden realiza una serie completa de getnexts automáticamente y se detiene cuando devuelve resultados que no están en el rango del OID especiado originalmente.

Ejemplo:

```
Snmpwalk –v 1 –c public 10.0.1.2 interfaces.ifTable.ifEntry.ifPhysAddress
```

commando que muestra la información almacenada en el grupo system del MIB de una máquina:

```
# snmpwalk -v 2c -c public localhost System

SNMPv2-MIB::sysDescr.0 = STRING: Linux mago.aut.uah.es 2.6.0-test11 #27
Tue
Dec 16 11:39:03 CET 2003 i686
SNMPv2-MIB::sysObjectID.0 = OID: NET-SNMP-MIB::netSnmpAgentOIDs.10
SNMPv2-MIB::sysUpTime.0 = Timeticks: (120246) 0:20:02.46
SNMPv2-MIB::sysContact.0 = STRING: Root (configure
/etc/snmp/snmp.local.conf)
SNMPv2-MIB::sysName.0 = STRING: mago.aut.uah.es
SNMPv2-MIB::sysLocation.0 = STRING: Unknown (edit /etc/snmp/snmpd.conf)
SNMPv2-MIB::sysORLastChange.0 = Timeticks: (4) 0:00:00.04
SNMPv2-MIB::sysORID.1 = OID: IF-MIB::ifMIB
SNMPv2-MIB::sysORID.2 = OID: SNMPv2-MIB::snmpMIB
SNMPv2-MIB::sysORID.3 = OID: TCP-MIB::tcpMIB
SNMPv2-MIB::sysORID.4 = OID: IP-MIB::ip
SNMPv2-MIB::sysORID.5 = OID: UDP-MIB::udpMIB
SNMPv2-MIB::sysORID.6 = OID: SNMP-VIEW-BASED-ACM-MIB::vacmBasicGroup
SNMPv2-MIB::sysORID.7 = OID: SNMP-FRAMEWORK-
MIB::snmpFrameworkMIBCompliance
SNMPv2-MIB::sysORID.8 = OID: SNMP-MPD-MIB::snmpMPDCompliance
SNMPv2-MIB::sysORID.9 = OID: SNMP-USER-BASED-SM-MIB::usmMIBCompliance
SNMPv2-MIB::sysORDescr.1 = STRING: The MIB module to describe generic
objects
for network interface sub-layers
SNMPv2-MIB::sysORDescr.2 = STRING: The MIB module for SNMPv2 entities
SNMPv2-MIB::sysORDescr.3 = STRING: The MIB module for managing TCP
implementations
SNMPv2-MIB::sysORDescr.4 = STRING: The MIB module for managing IP and
```

```
ICMP
implementations
SNMPv2-MIB::sysORDescr.5 = STRING: The MIB module for managing UDP
implementations
SNMPv2-MIB::sysORDescr.6 = STRING: View-based Access Control Model for
SNMP.
SNMPv2-MIB::sysORDescr.7 = STRING: The SNMP Management Architecture MIB.
SNMPv2-MIB::sysORDescr.8 = STRING: The MIB for Message Processing and
Dispatching.
SNMPv2-MIB::sysORDescr.9 = STRING: The management information definitions
for the SNMP User-based Security Model.
SNMPv2-MIB::sysORUpTime.1 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.2 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.3 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.4 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.5 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.6 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.7 = Timeticks: (4) 0:00:00.04
SNMPv2-MIB::sysORUpTime.8 = Timeticks: (4) 0:00:00.04
SNMPv2-MIB::sysORUpTime.9 = Timeticks: (4) 0:00:00.04
```

2.4.5 SNMPSET

Esta orden 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.

Ejemplo:

snmpset -v2c -c public 192.168.100.1 1.3.6.1.4.1.1166.1.19.3.1.18.0 LA MAC

2.4.5 SNMPTRAPS

SNMPTRAPs: informa de fallos en el agente (como perdida de la comunicación, caída de un servicio, problemas con la interfaz, etc). Es transmitido por el agente (o nodo administrado) y recibido por el nms (o nodo administrador).

CAPÍTULO 3

BASE DE INFORMACIÓN ADMINISTRATIVA ”MIB” Y CONFIGURACIÓN DE UN AGENTE

3.1 BASE DE INFORMACIÓN ADMINISTRATIVA MIB

Los MIB tienen un formato común de modo que aún cuando los dispositivos sean de fabricantes distintos puedan ser administrados con un protocolo muy general. Además la MIB define los objetos de la red operados por el protocolo de administración de red, y las operaciones que pueden aplicarse a cada objeto. Una variable u objeto MIB se define especificando la sintaxis, el acceso, el estado y la descripción de la misma.

3.1.1 BASE DE INFORMACIÓN DE GESTIÓN

Una MIB define un modelo conceptual de la información requerida para tomar decisiones de administración de red. La información que la MIB incluye tiene número de paquetes transmitidos, número de conexiones intentadas, datos de contabilidad, entre otros.

Esta información es parte de la gestión de red definida en el modelo OSI. Se definen variables usadas por el protocolo SNMP para supervisar y controlar los componentes de una red. Y está compuesta por una serie de objetos que representan los dispositivos (como enrutadores y conmutadores) en la red.

3.1.2 ESPECIFICACIONES DE LA MIB

La MIB define tanto los objetos de la red operados por el protocolo de administración de red, como las operaciones que pueden aplicarse a cada objeto.

La MIB no incluye información de administración para aplicaciones como Telnet, FTP o SMTP, debido a los inconvenientes que se presentan al instrumentar aplicaciones de este tipo para la MIB por parte de las compañías fabricantes.
Para definir una variable u objeto MIB es necesario especificar lo siguiente:

- ***Sintaxis:*** Especifica el tipo de datos de la variable, un valor entero, etc.
- ***Acceso:*** Especifica el tipo de permiso como: Leer, leer y escribir, escribir, no accesible.
- ***Estado:*** Define si la variable es obligatoria u opcional.
- ***Descripción:*** Describe textualmente a la variable.

3.1.3 ESTRUCTURA DE LA MIB

• Cada tipo concreto de objeto tiene un identificador único que sirve para nombrarlo. Además como el valor asociado a cada identificador es jerárquico (una secuencia de enteros), dichos identificadores también definen la estructura de la MIB (es una estructura en forma de árbol).

• Empezando por la raíz, existen tres nodos de primer nivel: iso, ccitt y join-iso-ccitt.Como ejemplo, bajo iso, un subárbol se reserva para uso de otras organizaciones, y una de ellas es el departamento de defensa de EEUU (dod). El RFC 1155 reserva un subárbol de dod para la Internet Activities Board (IAB) de la siguiente manera:

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

• Es decir, el nodo internet tiene el valor de identificador de objeto 1.3.6.1, que valdrá como prefijo para nodos a niveles más bajos del Árbol Estructura.

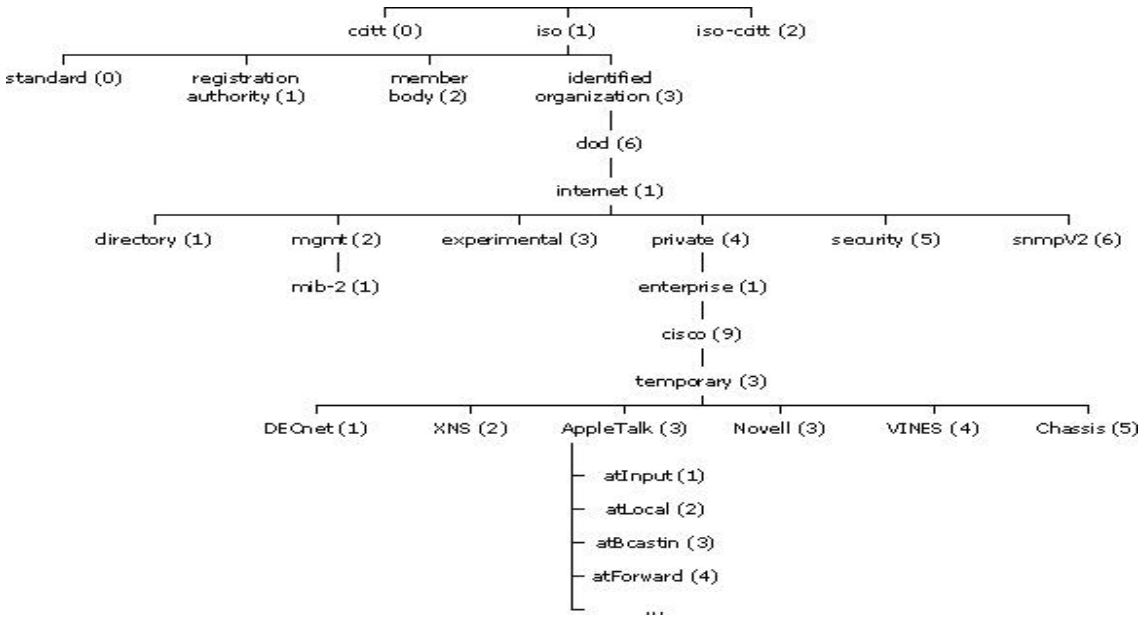


Figura 3.1.3

El mgmt contiene definiciones de información aprobada por el IAB.

Actualmente existen dos versiones de la MIB: mib-1 y mib-2. La segunda es una extensión de la primera. Como tienen el mismo identificador sólo uno puede estar presente.

• El objeto private actualmente sólo tiene un subárbol (enterprises) donde los fabricantes pueden almacenar extensiones propias. Cada fabricante registrado tiene su propio subárbol bajo enterprises.

La MIB-II se compone de los siguientes nodos estructurales:

- System: de este nodo cuelgan objetos que proporcionan información genérica del sistema gestionado. Por ejemplo, dónde se encuentra el sistema, quién lo administra.
- Interfaces: En este grupo está la información de los interfaces de red presentes en el sistema. Incorpora estadísticas de los eventos ocurridos en el mismo.
- At (address translation o traducción de direcciones): Este nodo es obsoleto, pero se mantiene para preservar la compatibilidad con la MIB-I. En él se almacenan las direcciones de nivel de enlace correspondientes a una dirección IP.
- Ip: En este grupo se almacena la información relativa a la capa IP, tanto de configuración como de estadísticas.
- Icmp: En este nodo se almacenan contadores de los paquetes ICMP entrantes y salientes.
- Tcp: En este grupo está la información relativa a la configuración, estadísticas y estado actual del protocolo TCP.
- Udp: En este nodo está la información relativa a la configuración, estadísticas del protocolo UDP.
- Egp: Aquí está agrupada la información relativa a la configuración y operación del protocolo EGP.
- Transmission: De este nodo cuelgan grupos referidos a las distintas tecnologías del nivel de enlace implementadas en las interfaces de red del sistema gestionado.

3.1.4 GRUPOS DE LA MIB

La MIB-1 define 126 objetos de administración, divididos en los siguientes grupos:

- **Grupo de Sistemas**

  Usado para registrar información del sistema, por ejemplo:
  Compañía fabricante del sistema.
  Tipo de Software.
  Tiempo que el sistema ha estado operando.

- **Grupo de Interfaces**

  Registra la información genérica acerca de cada interfaz de red, como el número de mensajes erróneos en la entrada y salida, el número de paquetes transmitidos y recibidos, el número de paquetes de broadcast enviados, MTU del dispositivo, etc.

- **Grupo de traducción de dirección**

  Comprende las relaciones entre direcciones IP y direcciones específicas de la red que deben soportar, como la tabla ARP, que relaciona direcciones IP con direcciones físicas de la red LAN.

- **Grupo IP**

  Almacena información propia de la capa IP, como datagramas transmitidos y recibidos, conteo de datagramas erróneos, etc. También contiene información de variables de control que permite a las aplicaciones remotas ajustar el TTL (Time To Live) de omisión de IP y manipular las tablas de ruteo de IP.

- **Grupo TCP**

  Este grupo incluye información propia del protocolo TCP, como estadísticas del número de segmentos transmitidos y recibidos, información acerca de conexiones activas como dirección IP, puerto o estado actual.

- **Grupo de ICMP y UDP**

  Lo mismo que el grupo IP y TCP.

- **Grupo EGP**

  En este grupo se requieren sistemas (ruteadores) que soporten EGP(Protocolo de Gateway o Salida Exterior).

3.1.5 BASE DE INFORMACIÓN DE GESTIÓN II (MIB-II)

La MIB-II se crea para extender los datos de administración de red empleados en redes Ethernet y WAN(Wide Area Network) usando ruteadores para un enfoque a múltiples medios de administración en redes LAN y WAN. Se agregan dos grupos:

- **Grupo de Transmisión**

  Soporta múltiples tipos de medios de transmisión, como cable coaxial, cable UTP, cable de fibra óptica y sistemas T1/E1.

- **Grupo SNMP**

  Incluye estadísticas sobre tráfico de red SNMP.

3.1.6 FORMATO DEL ÁRBOL DE IDENTIFICADORES DE OBJETO (OID)

Un OID, o Identificador de Objeto, es una secuencia de números que se asignan jerárquicamente y que permite identificar objetos en la red, siendo usados con gran cantidad de protocolos.

El nombre mediante el cual puedan ser identificados de manera única los dispositivos se le denomina identificador de objeto (object identifier). Éste es escrito como una secuencia de enteros separados por puntos; por ejemplo, la secuencia 1.3.6.1.2.1.1.1, especifica la descripción del sistema dentro del grupo system, del subárbol sysDescr.

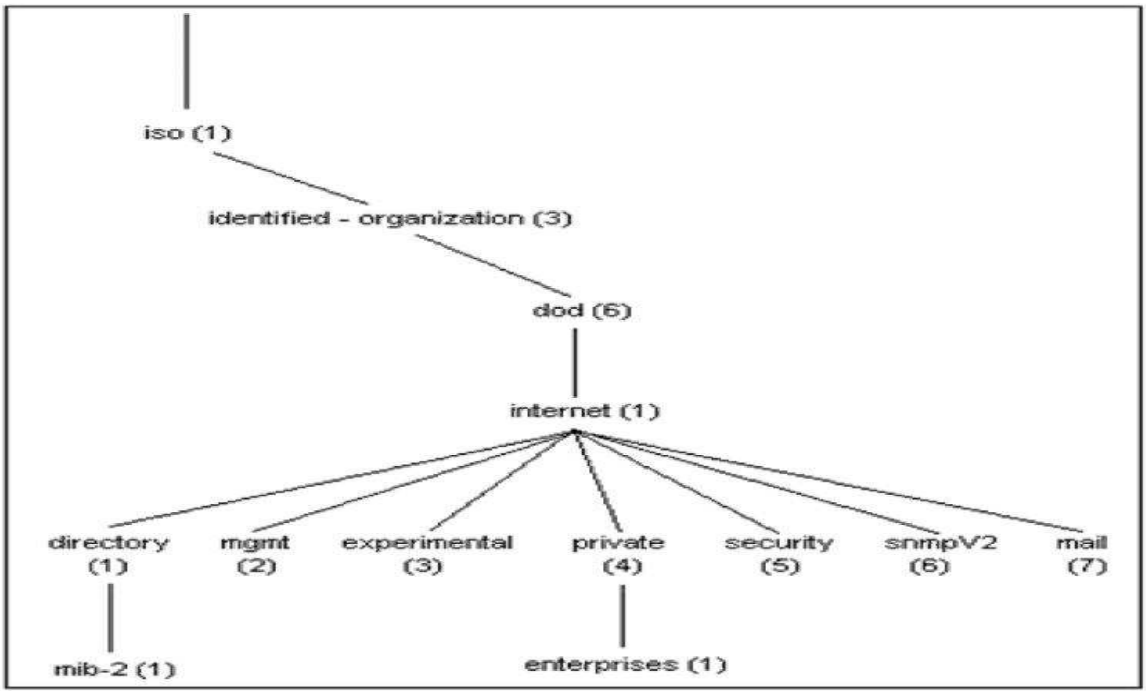


Figura 3.1.6

3.2 AGENTES POR HARDWARE y SOFTWARE

3.2.1 QUE ES UN AGENTE

El agente SNMP proporciona un control remoto o local y un seguimiento con control limitado de las los recursos de software en una red IP. El agente SNMP se encuentra en una pila IP y responde a SNMP "se" y "fija" de un gestor SNMP.

El nodo que ejecuta el agente SNMP se llama el nodo administrado. El nodo administrado es el sistema que contiene las tablas o el software de recursos de HMP que desea administrar de forma remota, el SNMP agente pone a disposición de las funciones administrativas de forma remota en una red IP.

3.2.2 VISION GENERAL DE UN AGENTE

Los agentes SNMP son programas de gestión que interactúan con una aplicación a través de un conjunto de atributos definidos por la aplicación. La aplicación define estos atributos en un archivo MIB (Management Information Base).

El agente SNMP no es una aplicación de gestión; es un programa de interfaz a través del cual la aplicación de gestión SNMP realiza peticiones para recuperar datos y establecer atributos de gestión. El agente SNMP se comunica con una aplicación de gestión SNMP utilizando el protocolo UDP. El protocolo UDP también permite al agente SNMP y a la aplicación de gestión SNMP residir en la misma máquina o en máquinas diferentes.

Además del acceso a nivel de máquina, los agentes SNMP también proporcionan comprobaciones de autorizaciones. La comprobación de autorizaciones se implementa utilizando nombres de comunidad. Hay dos tipos de nombres de comunidad:

- Los nombres de comunidad de lectura permiten a una aplicación de gestión SNMP recuperar valores de datos desde el MIB del agente SNMP.

- Los nombres de comunidad de grabación permiten a una aplicación de gestión SNMP establecer valores de datos desde el MIB del agente SNMP.

Para poder comunicarse con un agente SNMP, el programa emisor debe tener autorización en el agente SNMP. La autorización se realiza de dos formas:

- Las llamadas de SNMP se realizan desde una máquina host al agente SNMP. El nombre de host desde el que se realizan las llamadas debe estar definido en la tabla de acceso a los agentes SNMP.

- Los programas que llaman a un agente SNMP deben proporcionar códigos de acceso conocidos como nombres de comunidad.

3.2.3 AGENTES POR SOFTWARE

Los agentes por software es un pedazo del software que actúa para el usuario como comunicación entre el programa y el hardware en una relación de agencia. Tal "acción a nombre de" implica la autoridad para decidir a cuál (y si) es apropiada la acción. La idea es que los agentes no están invocados terminantemente para una tarea.

En la actualidad existen un sin número de aplicaciones SNMP entre ellas tenemos:

- PRTG
- JFFNS
- MRTG

3.2.3.1 AGENTE PRTG

PRTG Network Monitor es una potente herramienta de monitorización de la Paessler AG. Asegura la disponibilidad de componentes de red y mide el tráfico y el uso de la red. Ahorra costos ayudando a evitar fallos, optimizar conexiones, economizando tiempo de implementación y controlando acuerdos de nivel de servicio (SLAs).

Monitorización continua de redes y servidores facilita discernir y resolver problemas antes de que se conviertan en una amenaza al negocio:

· Evitar estrangulamientos de ancho de banda y de rendimiento de servidor.

· Proporcionar una mejor calidad de servicio a sus usuarios de manera pre activa.

· Reducir costos comprando el ancho de banda y el equipo necesario basándose en cargas efectivas.

· Incrementar ganancias evitando pérdidas causadas por fallos de sistema no descubiertos.

· Ganar tranquilidad: mientras PRTG no se comunique con el usuario mediante correo electrónico, SMS, radiolocalizador, etc. Se puede estar seguro que todo está funcionando correctamente y de esta manera se puede dedicar a otros negocios importantes.

Los componentes indispensables del software son:

Sensor:

Es un instrumento encargado de monitorear individualmente cada aspecto de un dispositivo de red.

Aparato:

Es un instrumento lógico que se ubica en cada punto de red (dirección IP) en donde se realiza el monitoreo. Cada aparato puede tener uno o más sensores.

Grupo:

Es el conjunto de aparatos que se pueden clasificar según sus características, las cuales ayudan al monitoreo mas fácil y efectivo.

Después de instalar el programa en la computadora en la que se va a realizar el monitoreo de la red, se crea automáticamente un acceso directo en el escritorio en el cual hacemos doble clic para acceder a la interfaz de usuario para el registro.

Reportes:

Se utilizan para analizar los datos de seguimiento, uno a la vez o en intervalos específicos de tiempos. Se Pueden definir cualquier cantidad de informes, en donde se especifique los sensores, seleccionando una plantilla y ejecutarlos en cualquier intervalo que desee desee el administrador, por ejemplo : diaria, semanal o mensual.

3.2.3.2 AGENTE JFFMNS

El JFFNMS es un software desarrollado por Javier Szyszlican, en el año 2002, para monitoreo de dispositivos, que integra varias utilidades que interrogan y capturan los datos de los dispositivos. Los programas utilizados por JFFNMS para el monitoreo, son los siguientes:

POLLER: Actúa como manager del protocolo SNMP interrogando a los agentes instalados en los dispositivos a monitorear.

SNMPTRAPD: Recibe los "traps" desde los dispositivos monitoreados. Los "traps" son eventos o alarmas que los dispositivos envían sin necesidad de ser consultados.

SYSLOG: Servicio que recoge la información de auditoría (mensajes logs) de los dispositivos. No forma parte del protocolo SNMP. Se basa en los registros de auditoría de los sistemas UNIX.

También utiliza programas como nmap, fping y otros para probar conectividad, puertos abiertos y otros parámetros de red. En tal sentido, JFFNMS es un sistema integrador para el monitoreo.

Además de estos programas fundamentales que realizan la consulta y captura de los datos, JFFNMS requiere de una base de datos robusta para almacenarlos (PostgreSQL ó MySQL), un Servidor Web para presentarlos (Apache), el intérprete PHP bajo el cual se ejecutará JFFNMS, utilidades y programas para generar gráficos (RRDtool, Graphviz, etc.)

Las características del JFFNMS, son:

· Permite monitorizar una red IP mediante SNMP.

· Puede ser utilizado para monitorizar cualquier dispositivo SNMP (servidor, router, puerto TCP y UDP).

· JFFNMS esta escrito en PHP, el cual funciona en Sistemas Operativos GNU/Linux, FreeBSD y Windows 2000/XP.

· Tiene soporte de base de datos (MySQL o PostgreSQL), integra logs de Syslog.

· JFFNMS se basa en las tecnologías: Apache, Cron, MySQL, PHP, RDDTool y SNMP.

· Necesita la instalación y configuración del complemento (agente) SNMP en los clientes.

3.2.3.3 AGENTE MRTG

MRTG (Multi Router Traffic Grapher) es una herramienta, escrita en C y Perl por Tobias Oetiker y Dave Rand, que se utiliza para supervisar la carga de tráfico de interfaces de red. MRTG genera un informe en formato HTML con gráficas que proveen una representación visual de la evolución del tráfico a lo largo del tiempo.

MRTG es un script en Perl que utiliza SNMP para leer cualquiera de los atributos de los objetos (contadores) de los routers y un programa rápido en C que procesa la información para visualizarla gráficamente en tiempo real.

Además, MRTG guarda la información por semanas, meses y años, monitorización hasta 200 enlaces.

MRTG se utiliza generalmente para monitorizar la carga del sistema, sesiones establecidas, tráfico, errores, etc

3.2.4 AGENTE POR HARDWARE

Los agentes por hardware son dispositivos encargados de administrar, monitorear componentes dentro de una red.

Entre los agentes por hardware existen de la misma manera un sin número de ellos en el mercado que encargados de aplicaciones específicas para los cuales fueron diseñados, entre ellos tenemos:

- Agente X300
- Agente PWR
- Agente SensorHubs
- Agente SensorSoft

3.2.4.1 AGENTE X300

Es un instrumento que registra la temperatura de equipos y es de gran alcance porque permite que el administrador supervise y que controle temperaturas vía red por medio de la dirección IP de los equipos de la red. Hasta 8 sensores de temperatura se pueden conectar a la vez, y las temperaturas se pueden ver en tiempo real usando un web browser. Además, el X-300™ tiene muchas características avanzadas incluyendo alarmas del email, control del relais, un intérprete del BASIC, y mucho más.

Figura 3.2.4.1

Características:

- El web browser no basó ningún software requerido, en otras palabras se puede verificar el equipo en un navegador si necesidad de software extra.
- Alarmas del email
- Tres, relais 3-Amp
- Campo de temperaturas actualizable
- Reloj en tiempo real incorporado con el respaldo de un condensador
- Medición de la temperatura en tiempo real
- Paginas HTTP de las ayudas: TCP, SNMP, Modbus/TCP
- Registro y gerencias adicionales de datos de las ayudas de los servicios
- Medición Interna de la temperatura y monitor de voltaje para los diagnósticos

- Gama de temperaturas ancha de funcionamiento
- Conector terminal desprendible para el cableado conveniente

3.2.4.2 AGENTE PWR

El equipo Hwg-PWR es un dispositivo Ethernet que permite el control remoto de mediciones del consumo de la electricidad, del calor, del agua o del gas usando los sensores de electricidad, sensores contadores de agua, sensores de gas, de calor y otros sensores equipados de la interfaz de M-BUS. Además de la medición, el dispositivo también apoya alarmar a través de email o las trampas del SNMP siempre que se excedan los valores permitidos.

Figura 3.2.4.2

Características básicas HWg-PWR

- Interfaz Ethernet: RJ45 (10BaseT)
- Acceso WEB: Web server encajado/GUI
- Trabajos hasta con tres sensores en puerto M-BUS (sensores de electricidad, sensor de gas…)
- Número ilimitado de sensores
    - Entrada de energía instantánea
    - Consumo total de voltaje
    - Línea voltaje
    - Línea corriente

- Velocidad del flujo
- etc.

- Detección automática de sensores y de cantidades medidas
- Ayuda para los sensores certificados y calibrados
- Ayudas monofásicas y sensores polifásicos de la electricidad
- Registro de valores medidos con la opción de trazar gráficos
- Configuración de la gama permitida de cantidades medidas
- Contadores independientes de la energía para las lecturas periódicas (diario, semanal, mensual, anualmente,…)
- Envío por correo electrónico periódico de valores adquiridos sobre el HTTP
- Protocolos de comunicación de M2M: SNMP, XML y Modbus/TCP
- Respuesta a los umbrales: Trampa del SNMP, email

3.2.4.3 AGENTE SENSORHUBS

El SensorHubs es un dispositivo de actualización que proporciona la capacidad de controlar a los ambientes de redes críticos. El SH permite posee sensor de temperatura que se encuentra incluido o de humedad en el medio. Con esta unidad proporcionamos un sensor de temperatura. Estos sensores le darán la libertad para supervisar cualquier situación crítica, si sea temperatura, humedad, voltaje, seguridad del sitio, o circulación de aire siempre constante.

Figura 3.2.4.3

Características

- Email
- Protección de contraseña completa
- Capacidades completas del SNMP
- Tres umbrales definidos por el usuario: Temperatura, Humedad, Voltaje
- Representación gráfica de los datos
- Encripción de datos
- 2 puertos externos del sensor para sensar otros dispositivos
- Sensor de temperatura incluido

3.2.4.4 AGENTE SENSORSOFT

La alarma de Sensorsoft es una aplicación de red independiente que permite que al administrador proteger el equipo y los productos que se puedan dañar por condiciones ambientales indeseadas. La alarma de Sensorsoft puede utilizar la serie completa de dispositivos de Sensorsoft para supervisar temperatura, humedad, la inundación de información, el apagón y contactos secos de equipos industriales.

Usando la tecnología enchufable incorporada de Sensorsoft, la alarma puede apoyar casi cualquier dispositivo serial. Esto permite que se supervise una amplia gama de variables de medidas eléctricas, a amenazas más complejas del producto químico, biológicas y nucleares. La alarma de Sensorsoft se puede manejar de un software del web browser o de la gerencia del SNMP. Funciona como una aplicación independiente que recoge y exhibe las lecturas de los sensores, datos de registros, y las alarmas personales cuando las condiciones ambientales llegan a ser inaceptables.

Figura 3.2.4.4

Características generales

- Web server incorporado para la gerencia en Internet
- SNMP directo manejable vía un MIB fácil de utilizar
- Las acciones alertas incluyen el email, trampas del SNMP, el control de relais, y la captura de la imagen de la cámara
- Configuración de los umbrales para la advertencia,
- Compatible con dispositivos detectables del NIST Sensorsoft
- Los datos del sensor se pueden representar gráficamente vía una herramienta de representación gráfico gráficamente en Internet integrada
- El intervalo del registro de los datos del sensor es seleccionable para días o meses
- La salida de XML permite que los datos vivos sean exportados a los usos externos
- Protegida por contraseña del administrador y usuario inalterable
- Se Registra toda la historia de alertas en ficheros
- Interfaz de Ethernet RJ-45 10/100
- Consumo de energía baja - dibuja menos de 3 vatios (5 VDC @ 0.6 amperios de máximo)
- La capacidad del montaje del NFS permite registros del tamaño ilimitado

3.3 CONFIGURACIÓN DE UN AGENTE

A continuación se muestra las configuraciones de agentes en dispositivos para la administración y gestión del mismo

3.3.1 CONFIGURACIÓN PARA UN SISTEMA LINUX

Empezamos con la instalación del paquete para el agente SNMP, abrimos una Terminal y digitamos.

Figura 3.3.1(a)

El siguiente paso es editar el archivo **/etc/deafault/snmpd** y borrar la dirección de loopback 127.0.0.1 para poder monitorear otras maquinas.

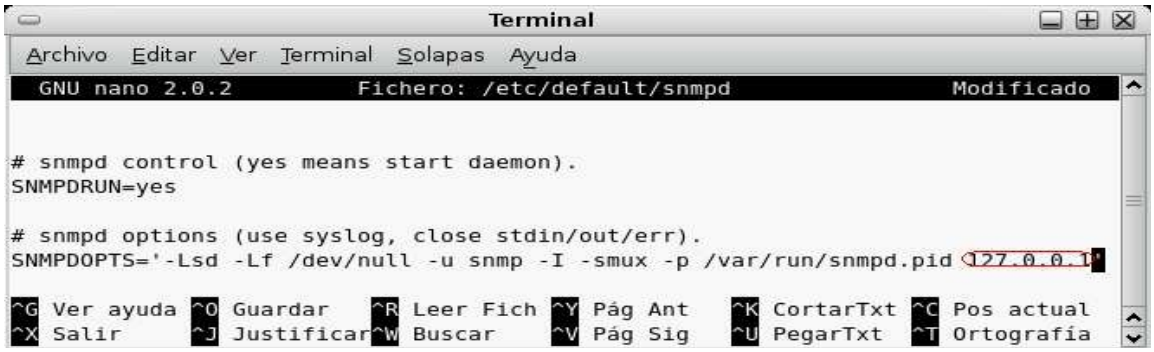


Figura 3.3.1(b)

Luego buscamos la línea sec.name source, agregamos al final la línea marcada en la imagen en el archivo de configuración de nuestro agente **/etc/snmp/snmpd.conf**, donde mired es el nombre del grupo, seguido de

192.168.0.0/24 que es el identificador de la red y finalmente public que es el nombre de la comunidad

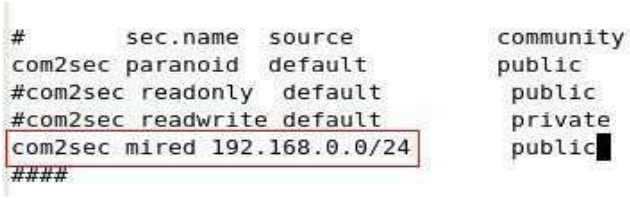


Figura 3.3.1(c)

Estando en el mismo archivo buscamos la línea sec.model sec.name y agregamos el grupo de líneas marcadas en la imagen, donde mired es el grupo, v1 y v2 es la versión del snmp y rocommunity para establecer que la comunidad publica es de solo lectura.

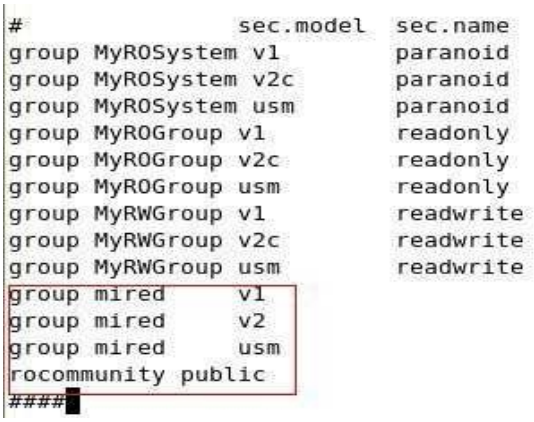


Figura 3.3.1(d)

Con eso ya quedaría instalado y configurado el agente SNMP para que nuestra maquina pueda ser monitoreada.

3.3.2 CONFIGURACIÓN PARA UN SISTEMA WINDOWS SERVER 2003

Lo Primero que hacemos es ir a inicio, panel de control, agregar o quitar programas.

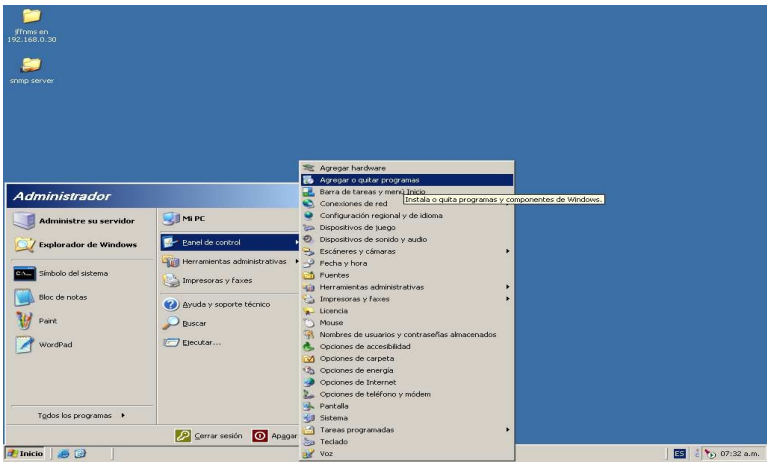


Figura 3.3.2(a)

En la siguiente opción damos clic en agregar o quitar componentes de windows.

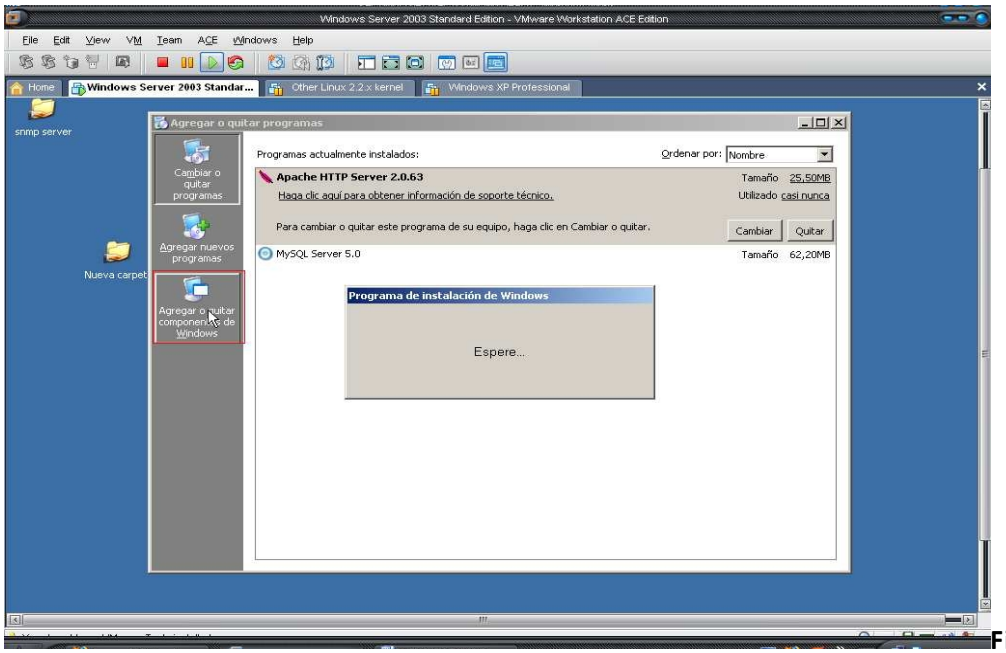


Figura 3.3.2(b)

En la siguiente imagen nos sale los componentes de windows y le damos clic en herramientas administración y supervisión, damos clic en detalles.

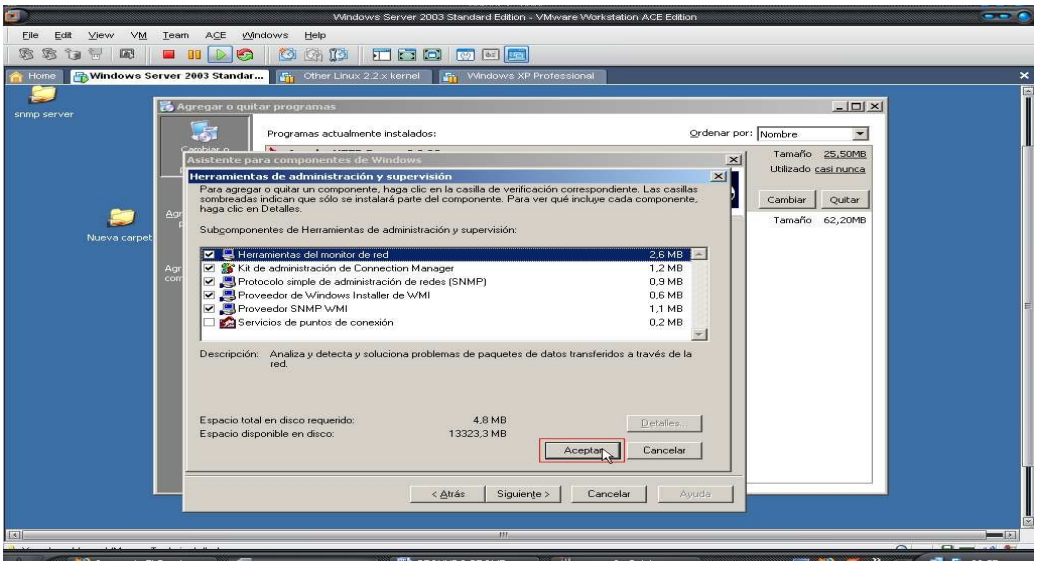


Figura 3.3.2(c)

Damos clic en aceptar, y después damos clic en siguiente.

- Esperamos a que termine nuestra instalación.

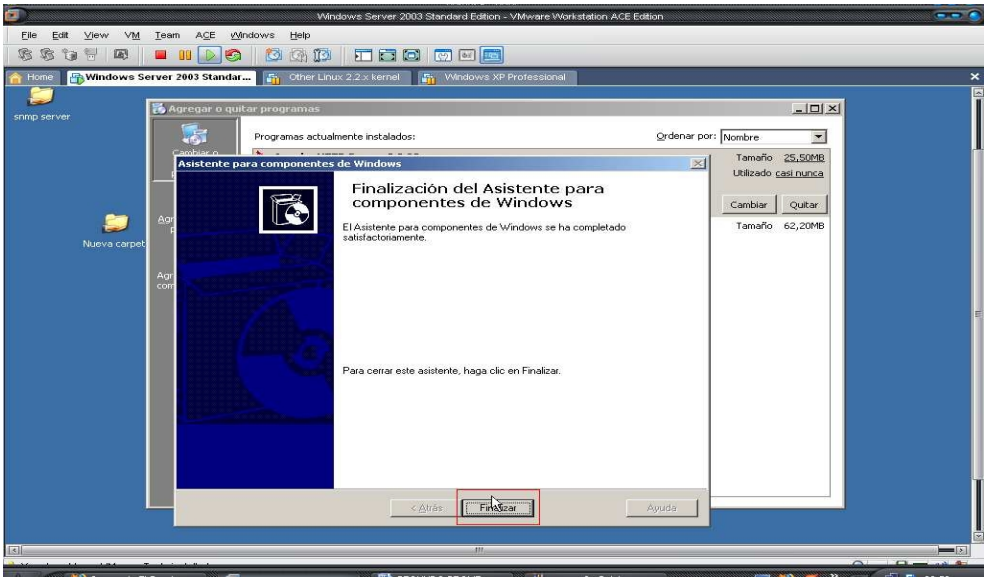


Figura 3.3.2(d)

Ahora nos vamos para inicio, herramientas administrativas y damos clic en servicios.

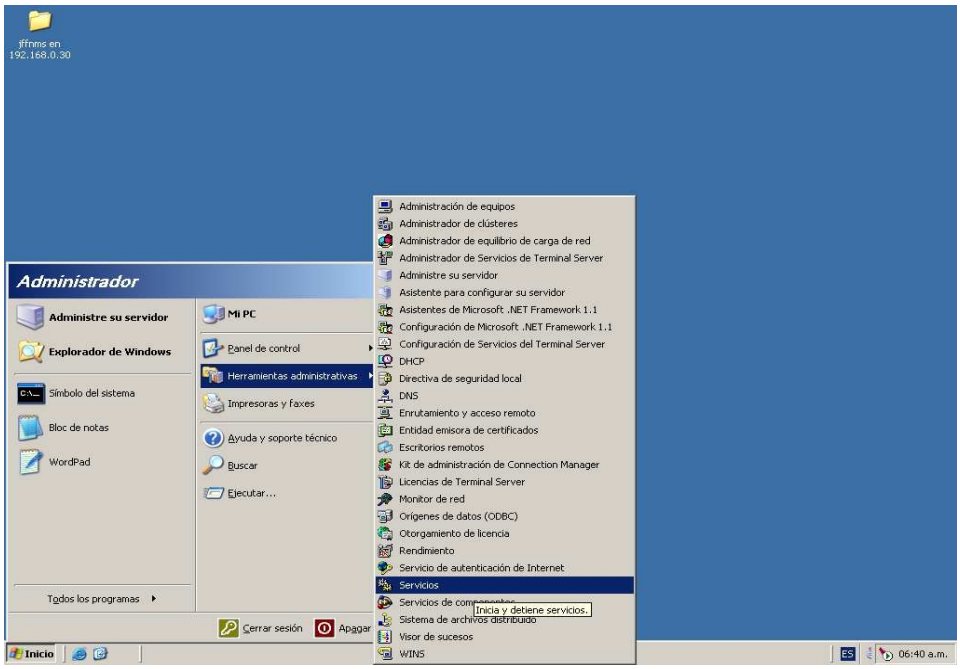


Figura 3.3.2(e)

Ahora buscamos servicio SNMP y damos clic derecho y elegimos propiedades

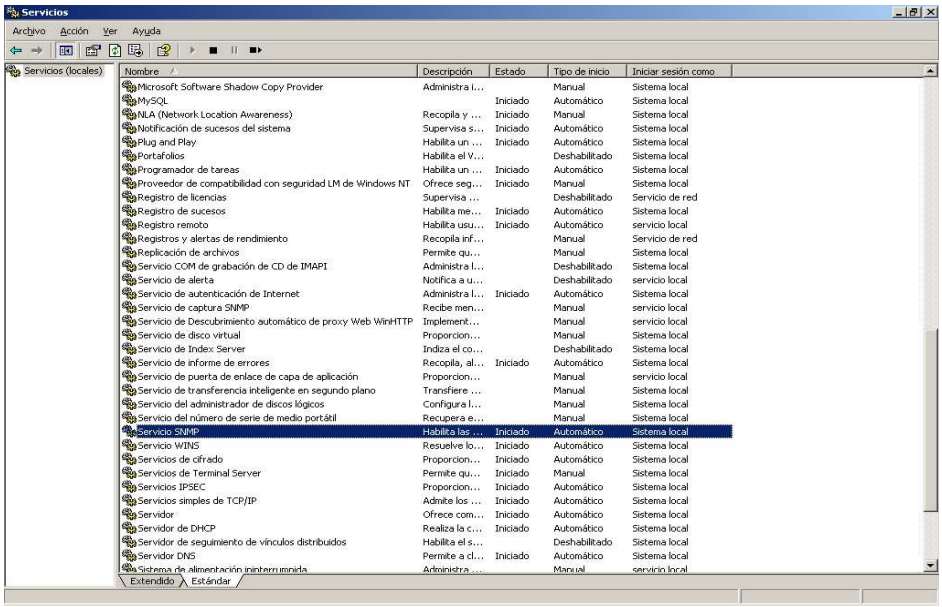


Figura 3.3.2(f)

Ahora nos sale las propiedades del SNMP.

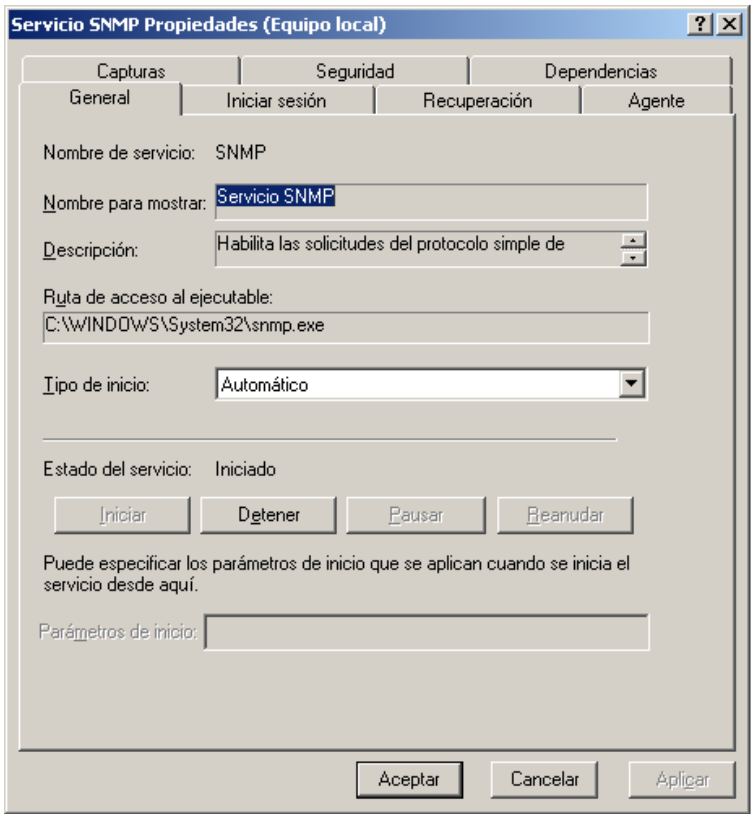


Figura 3.3.2(g)

-Damos clic en Agente y lo modificamos.

-En contacto, colocamos Administrador.

-En ubicación, colocamos donde estamos ubicados en este caso es sala55.

-En servicio escogemos las opciones que queremos para nuestra red.

Físico: especifica si el equipo administra dispositivos físicos como una partición de disco duro.

Aplicaciones: especifica si el equipo utiliza programas que envían datos a través de TCP/IP.

Vínculo de datos y subred: especifica si este equipo administra una subred o un vínculo de datos TCP/IP, como un puente.

Internet: especifica si este equipo actúa como una puerta de enlace IP (enrutador).

De un extremo a otro: especifica si este equipo actúa como un host IP

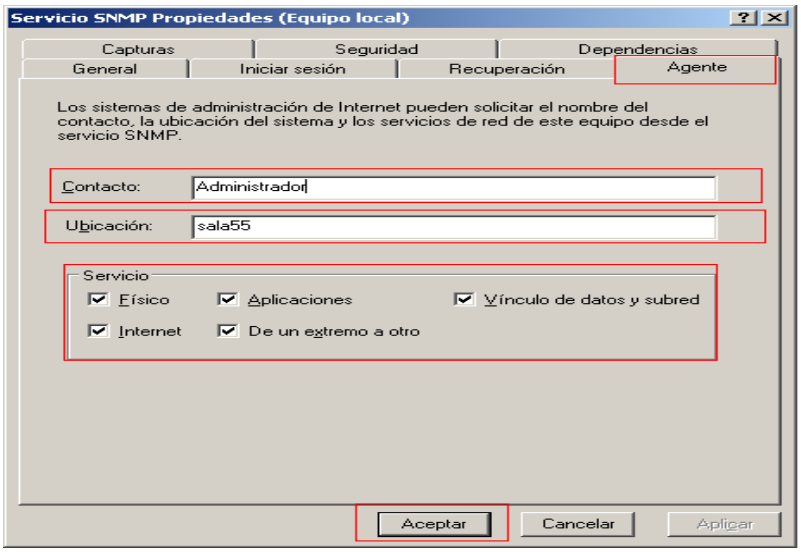


Figura 3.3.2(h)

Y damos clic en Aceptar.

-Ahora damos clic en Seguridad, y lo modificamos.

Le damos clic en enviar captura de autenticación.

En nombres de comunidad aceptados, damos agregar y colocamos el nombre de la comunidad, la cual es public.

Y seleccionamos la opción Aceptar paquetes SNMP de cualquier host.

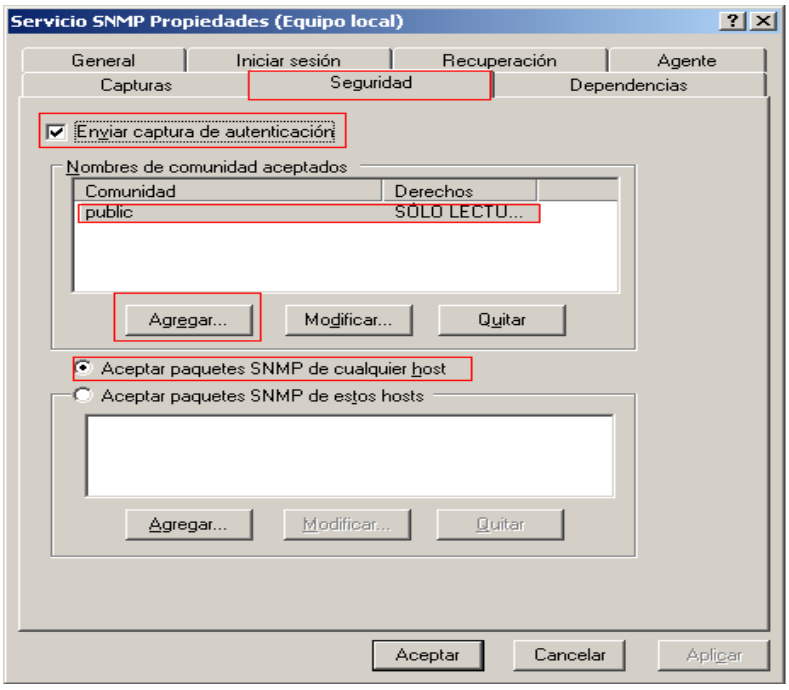


Figura 3.3.2(i)

-Ahora damos clic en Capturas.

En nombre de comunidad, colocamos public y damos clic en agregar a lista. En Destinos de capturas damos agregar y colocamos la IP del servidor donde se encuentra instalado el software de administración por ejemplo puede ser 192.168.0.1

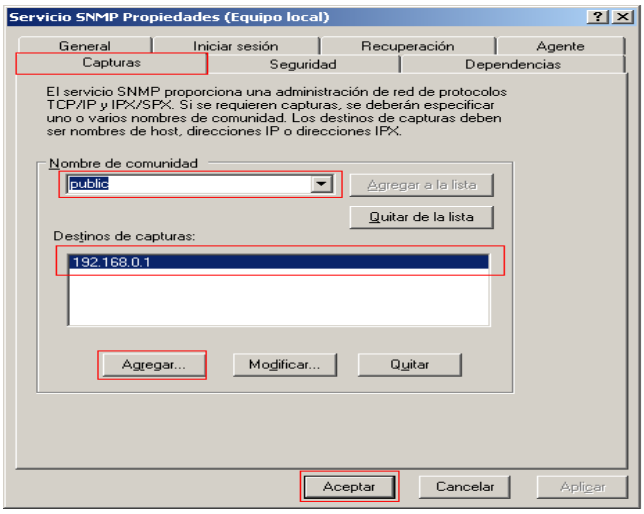


Figura 3.3.2(j)

-Por ultimo damos clic en Aceptar.

-Procedemos a reiniciar el servicio SNMP, damos clic derecho en servicio SNMP y le damos clic en la opción reiniciar, y esperamos un momento. Y listo

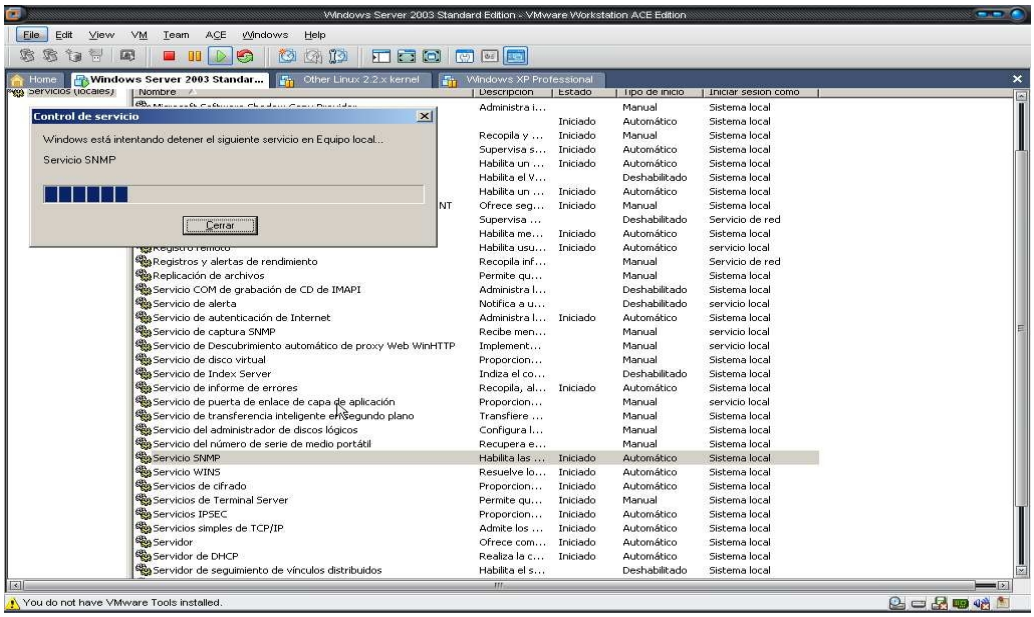


Figura 3.3.2(k)

3.3.3 CONFIGURACIÓN PARA UN SISTEMA WINDOWS XP

-Nos vamos para inicio, y damos clic en panel de control.

Figura 3.3.3(a)

-Damos clic en agregar o quitar componentes de windows.

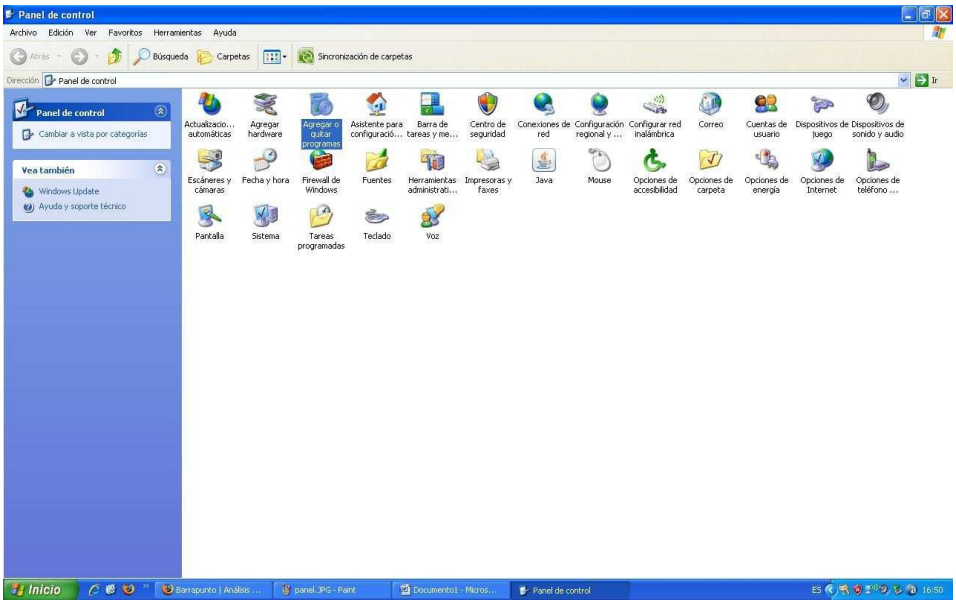


Figura 3.3.3(b)

Clic en Agregar o quitar Componentes de Windows, para instalar componentes que no vienen en una instalación por defecto.

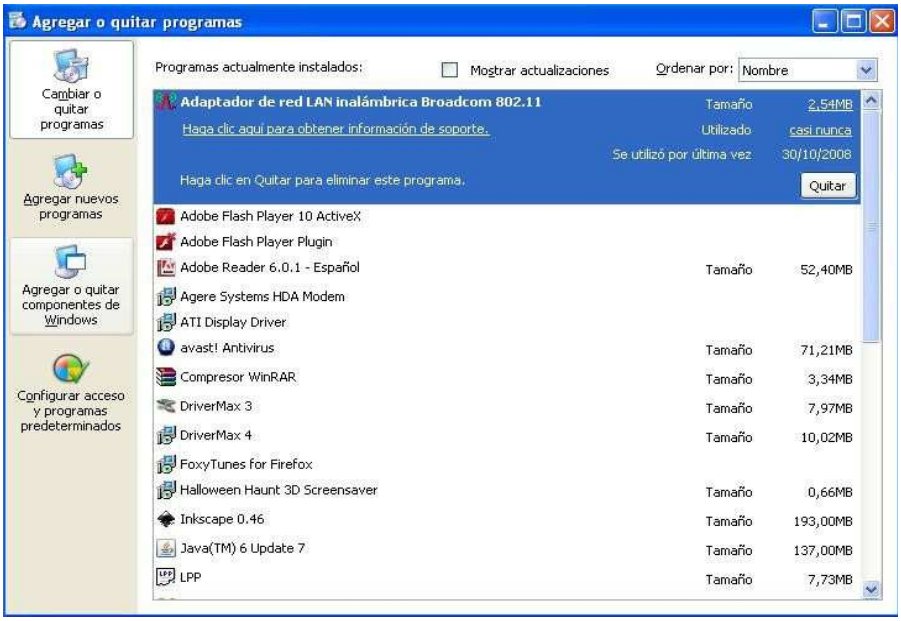


Figura 3.3.3(c)

Luego buscar la opción Resaltada en la imagen y hacer clic en detalles.

Figura 3.3.3(d)

La opción marcada de azul en la imagen, damos clic en el check box hasta que sea seleccionada.

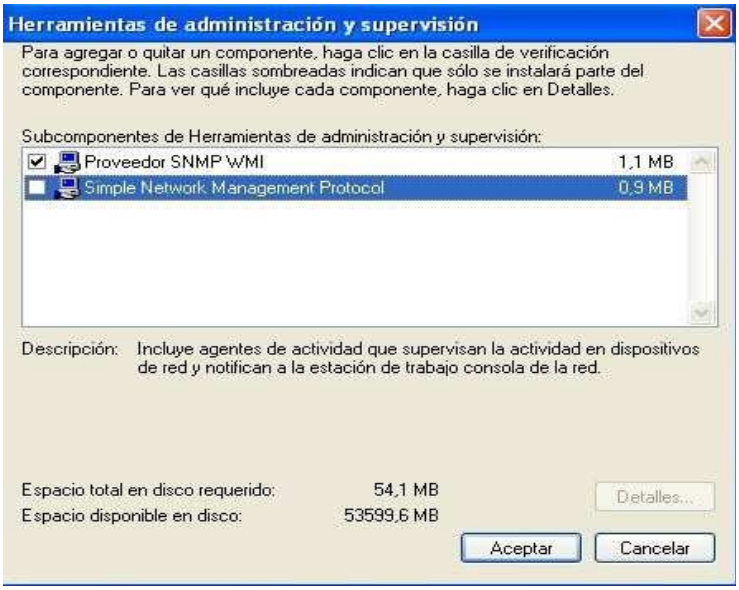


Figura 3.3.3(e)

Acto seguido dar clic en Aceptar – Siguiente y se habrá completado la instalación de nuestro agente SNMP en Windows.

Después de haber instalado el agente procederemos a su configuración. De nuevo clic en Panel de Control.

Figura 3.3.3(f)

Buscar la opción Herramientas Administrativas y hacer doble clic.

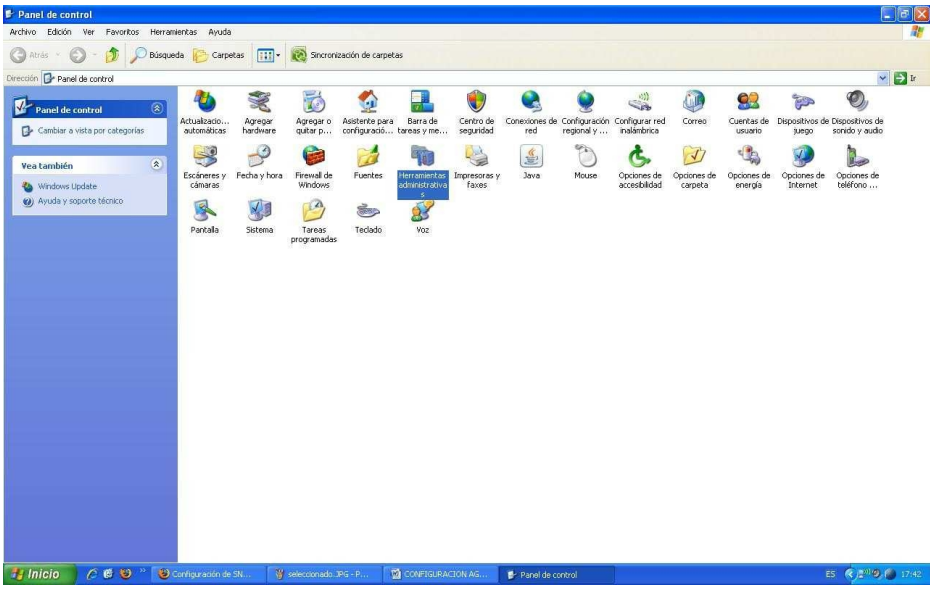


Figura 3.3.3(g)

Hacer doble clic en esta opción ya que aquí se encuentran todos los servicios que corren en Windows y nuestro agente también.

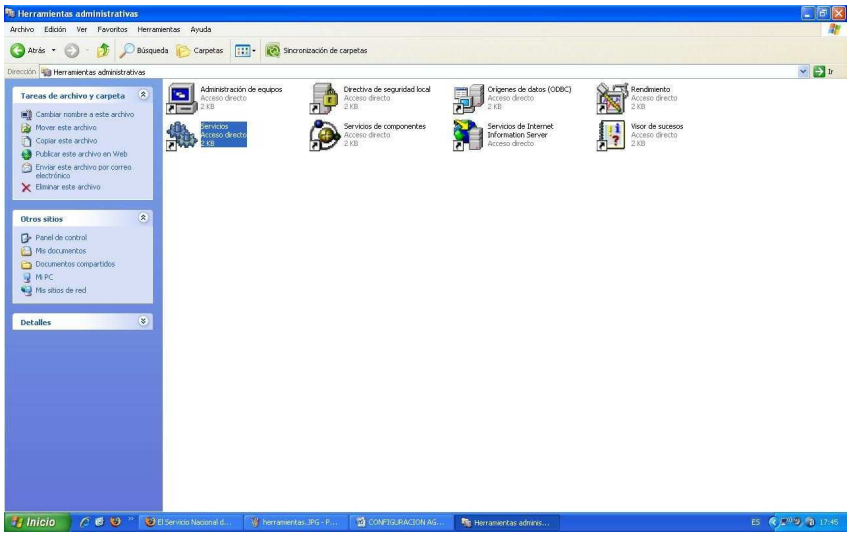


Figura 3.3.3(h)

Buscar la opción servicio SNMP y hacer clic derecho y elegir Propiedades.

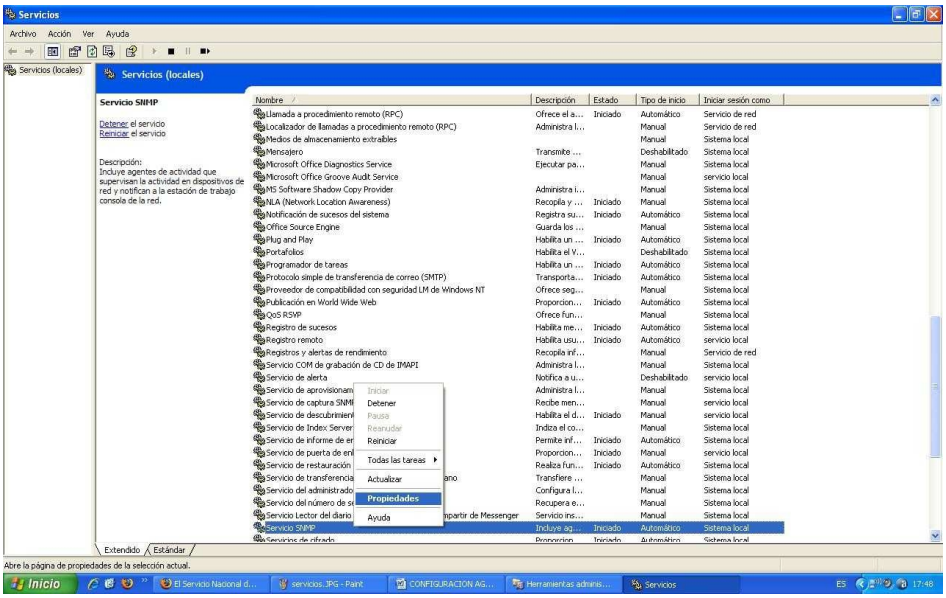


Figura 3.3.3(i)

Dar clic en la pestaña capturas para indicar el nombre de la comunidad, para este ejemplo utilice la comunidad pública que es la que viene por defecto.

Figura 3.3.3(j)

Después ir a la pestaña Seguridad, para darle los permisos a nuestra comunidad, inicialmente esta tendrá permisos de Lectura y Escritura en este caso la dejaremos de Solo Lectura.

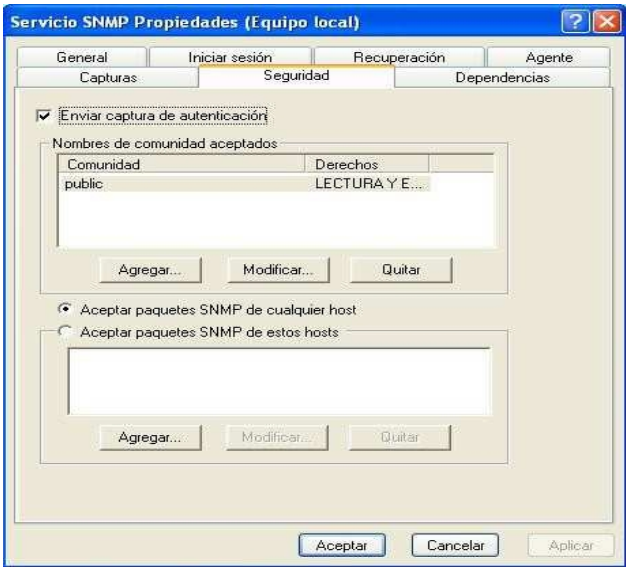


Figura 3.3.3(k)

Para cambiar los permisos seleccionamos la comunidad que creamos y dar clic en Modificar.

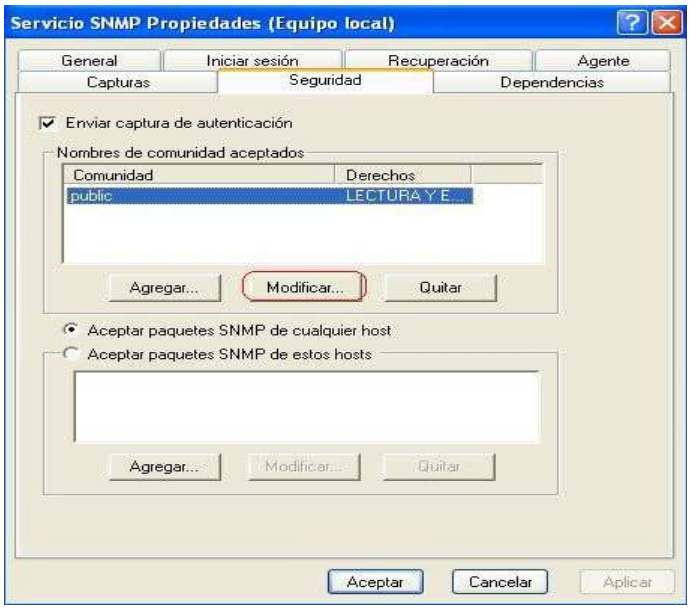


Figura 3.3.3(l)

Por último elegimos los permisos y la comunidad a la cual van a aplicar.

Figura 3.3.3(m)

3.3.4 CONFIGURACIÓN PARA UN SISTEMA WINDOWS VISTA

Ir a inicio – Panel de Control – Programas y Características y dar doble clic sobre este.

Figura 3.3.4(a)

Luego hacer clic en Activar o Desactivar las características de Windows.

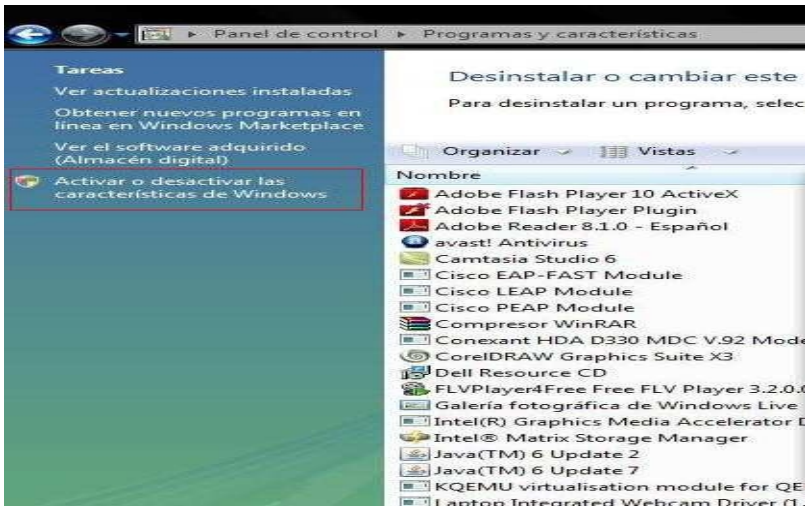


Figura 3.3.4(b)

Elegir la opción proveedor de SNMP de WMI y hacer clic en aceptar.

Figura 3.3.4(c)

Comenzarán a instalarse los componentes necesarios para el agente.

Figura 3.3.4(d)

Luego pedirá que reiniciemos el equipo para aplicar los cambios, dar clic en reiniciar.

Figura 3.3.4(e)

Por último vamos a panel de control y damos click en Herramienta Administrativas

Figura 3.3.4(f)

Luego Damos click en servicios:

Figura 3.3.4(g)

-Escogemos servicio SNMP y damos clic, y nos sale varias opciones para activar nuestro agente.

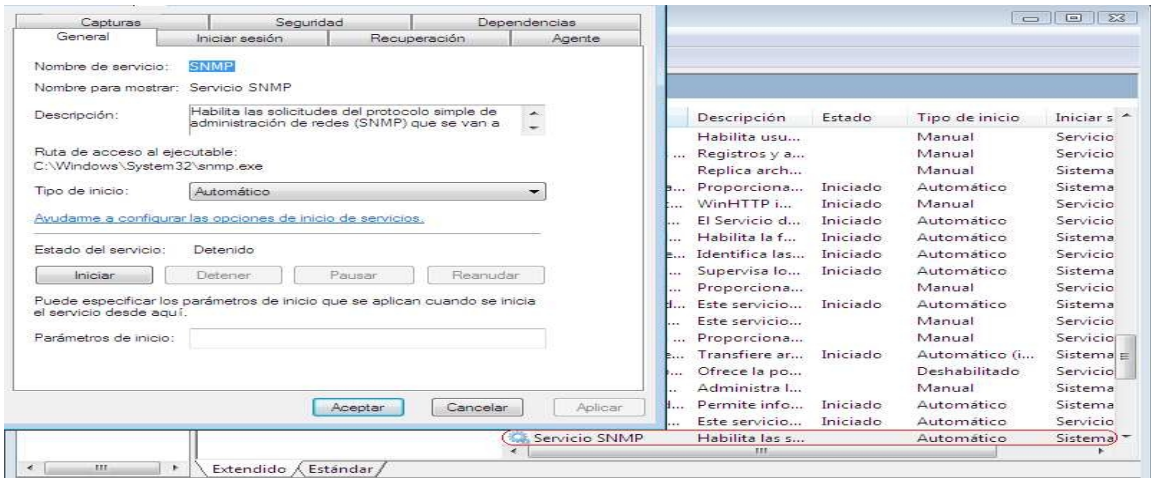


Figura 3.3.4(h)

Damos clic en Seguridad.

-Seleccionamos Enviar captura de autenticación

-Damos clic en agregar y colocamos el nombre de nuestra comunidad.

-Y habilitamos Aceptar paquetes SNMP de cualquier host.

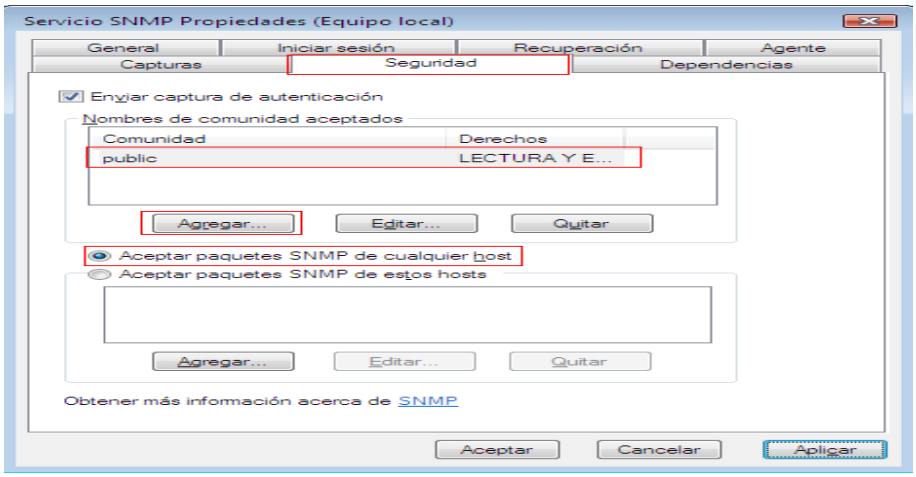


Figura 3.3.4(i)

-Damos clic en Capturas.

-En nombre de la comunidad, colocamos public y le damos clic en agregar a la lista.

-En destinos de capturas damos agregar y colocamos la dirección ip de nuestro equipo.

-Damos clic en aplicar y aceptar.

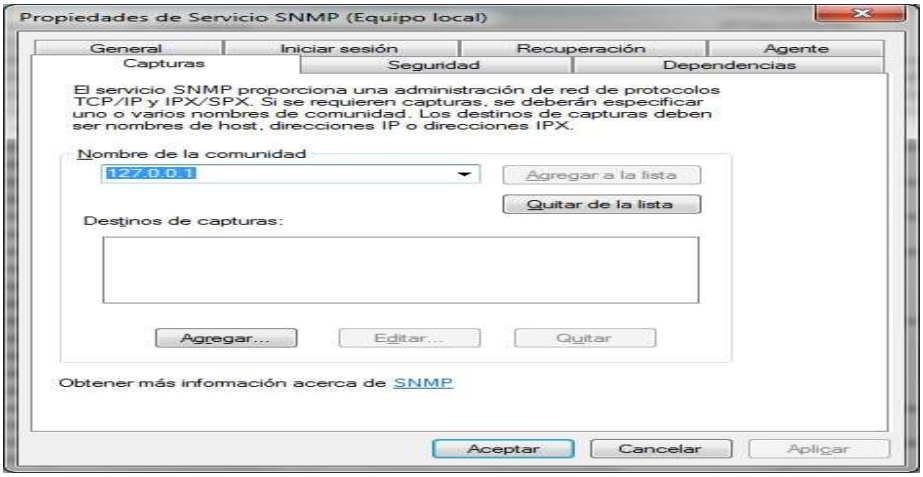


Figura 3.3.4(j)

Y listo quedaría habilitado el servicio SNMP

CAPÍTULO 4

MONITOREO DE LA RED FIEC

4.1 OBJETIVOS

Nuestro Objetivo es poder implementar una solución eficaz sobre Gestión y Administración de un sistema de monitoreo de redes basandonos en el Protocolo SNMP visto anteriormente en capítulos anteriores.

Con la finalidad de poder mejorar el rendimiento y disponibilidad de los dispositivos que tiene la Red Fiec entre los dispositivos son: Router , Swithces, Computadores, se procederá a impletar un software basado en SNMP.

4.2 CONFIGURACIÓN DE LA RED

La Red Fiec según los datos proporcionados por los Administradores se maneja con las siguientes Topologías:

4.2.1 TIPO DE TOPOLOGIA ESTA IMPLEMENTADA EN LA RED FIEC

La Topología implementada en la Red Fiec es de Estrella Extendida.

Esta topología es igual a la topología en estrella, con la diferencia de que cada nodo que se conecta con el nodo central también es el centro de otra estrella.

Generalmente el nodo central está ocupado por un hub o un switch, y los nodos secundarios por hubs.

La topología en estrella extendida es sumamente jerárquica, y busca que la información se mantenga local. Esta es la forma de conexión utilizada actualmente por el sistema telefónico.

4.2.2 TIPO DE TOPOLOGIA IMPLEMENTADA EN LAS SUB REDES

Las subredes implementadas en la Red Fiec de la misma manera que en la Red principal se lo realiza con topología Estrella Extendida debido a que la ventaja de esto es que el cableado es más corto y limita la cantidad de dispositivos que se deben interconectar con cualquier nodo central.

La topología en estrella extendida es sumamente jerárquica, y busca que la información se mantenga local.

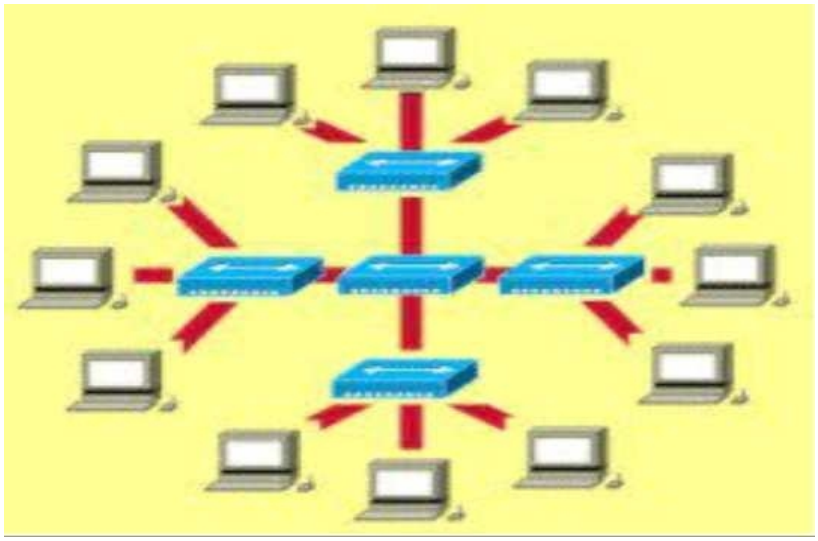


Figura 4.2.2

4.2.3 SUBREDES IMPLEMENTADAS EN LA RED FIEC

En La Red Fiec se pudo determinar que está conformada por:

- 5 Redes /24 tipo públicas y 1 red /24 tipo privada
- Existen 18 subredes dentro de la Red
- Además tienen 2 routers CISCO 3750G que en realidad son switches de capa 3 que pueden enrrutar paquetes entre redes o determinar la ruta que debe tomar el paquete de datos.

4.3 HARDWARE Y SOFTWARE UTILIZADO

Los Hardware que posee la Facultad según los datos que se nos proporcionó son los siguientes:

4.3.1 CLASES DE HADWARE DE LA RED FIEC

La red fiec para su funcionamiento está conformada por componentes como: Computadoras y Switches de capa 2.

Las Computadoras que conforman toda la red de la facultad son aproximadamente 800 PC en total entre las cuales se tienen computadoras HP, computadoras IBM, y las computadoras CLONES y además 12 Servidores utilizados en la Red

Los switches implementados de toda la red son un total de 37 tanto switches CISCO como DLINK entre ellos se tienen los siguientes:

- Switch Capa 3, CISCO 3750G

- Swtich, CISCO 2960G

- Switch, CISCO 2960

- Switch Dlink 3226L

- Swtich Dlink 3526

- Switch Dlink 3028

4.3.2 TIPOS DE SERVIDORES UTILIZADOS EN LA RED FIEC

Los servidores que se encuentran implementados en la red de la fiec según los datos que se nos proporciono son los siguientes:

- Servidor de Correo, web, openLdap, sistemas (CRM, reservarSalas, mensajeria, Controlac), dns, dhcp.

- Servidor de impresión, sistema SATT, base del sistema académico (replica del CSI), antivirus.

- Existen otros servidores de menor importancia para otros asuntos como IEEE, Academias, Ingeniería de Software.

4.3.3 CLASES SOFTWARE IMPLEMEMTADOS EN LA RED DE LA FIEC

Las plataformas que están implementadas en la red de la facultad son LINUX y WINDOWS.

Para las aplicaciones de los Servidores de Correo estas están implementadas bajo Linux debido a su rendimiento y bajo costo que lo convierten en la plataforma ideal para proporcionar, en un sólo servidor, servicios de correo desde 10 a 250.000 buzones.

En cambio para aplicaciones de Servidores de Impresión estos están implementados bajo Windows debido a que son tolerantes a fallos, con facilidades para la localización de dispositivos, que aumenta la fiabilidad, capacidad de gestión y seguridad del servicio de impresión en red.

4.4 RECOPILACIÓN DE LOS MIBS

En la realización del proyecto a continuación se muestra una recopilación los Mibs Cisco escritos en lenguaje asn1 que se obtuvieron:

4.4.1 MIB DE ROUTERS IMPLEMENTADOS EN LA RED FIEC

Entre los MIBS que se pudo investigar tenemos:

- CISCO-5800-HEALTH-MON-MIB-V1SMI.my
- CISCO-C2900-MIB-V1SMI.my

4.4.2 PROCESO DE IMPLEMENTACION DE LOS MIB UTILIZADOS

Como se vió con el punto anterior se mencionó los MIB de algunos de los equipos que posee la red fiec, acontinuacion mostramos el proceso de implementacion para que el sistema utilizado se los pueda anexar.

1) Previamente se tiene que descargar el programa ” Paessler MIB Importer “ que convierte un archivo MIB escrito en lenguaje asn1 a un código reconocible para el software PRTG en el que se implementara. Y ejecutarlo:

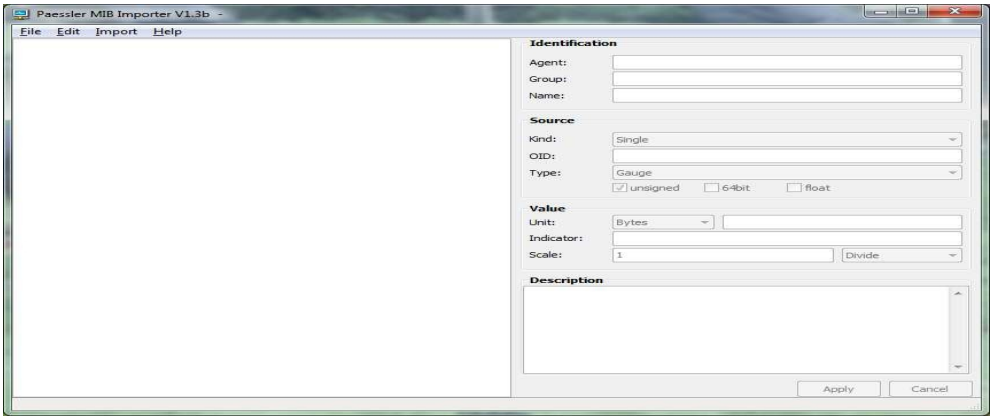


Figura 4.4.2(a)

2) Luego seleccionamos la pestaña Import :

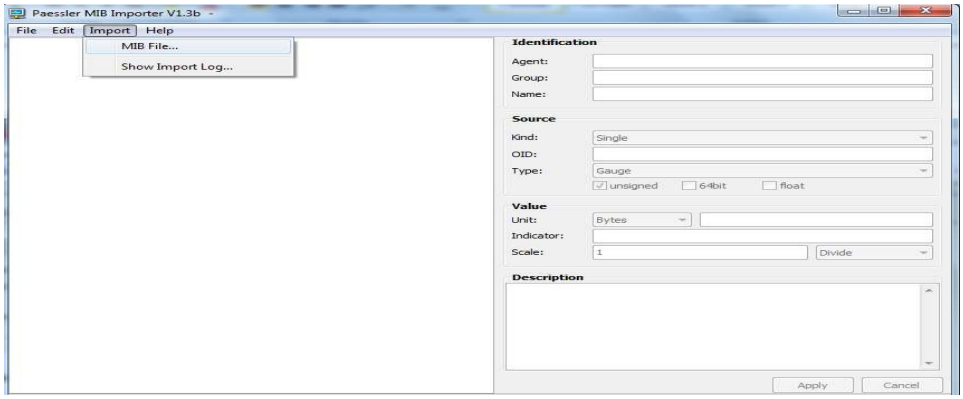


Figura 4.4.2(b)

3) Seleccionamos la MIB que deseamos convertir. Previamente se tiene que tener descargado las MIBS que se desea utilizar:

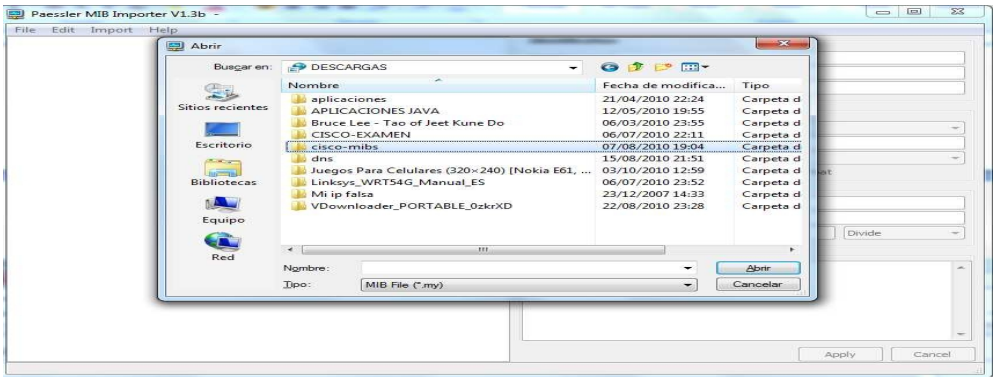


Figura 4.2.2(c)

4) En Type selecionamos MIB File (*my):

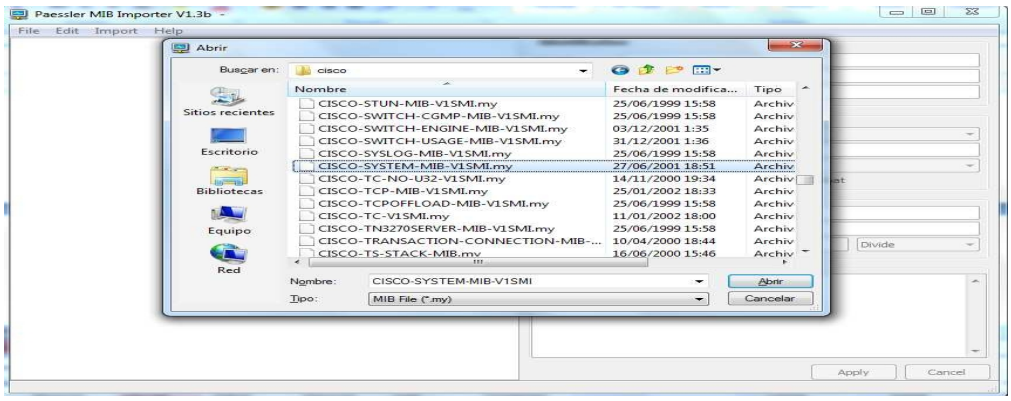


Figura 4.2.2(d)

5) Damos click en Abrir y nos mostrará que el proceso de convertir al formato reconocible por el software y despues click en close:

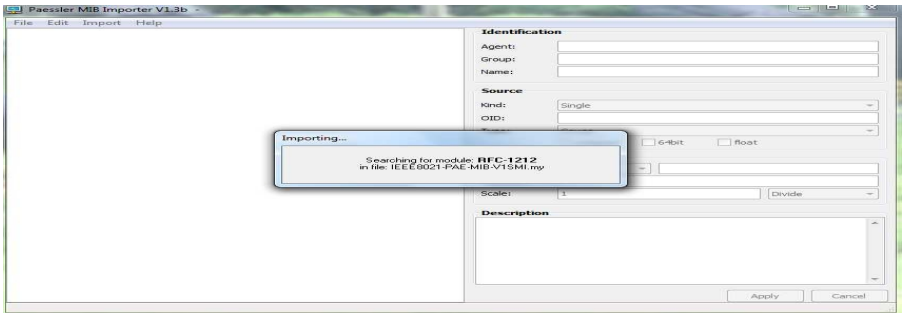


Figura 4.2.2(e)

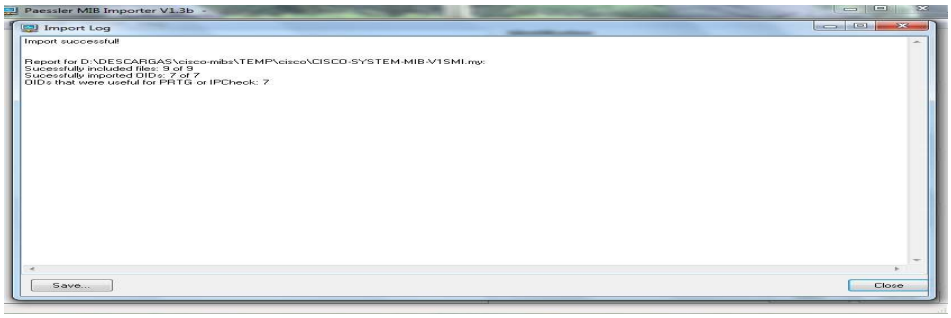


Figura 4.2.2(f)

6) Nos mostrará una ventana en que el proceso de convertir ha terminado

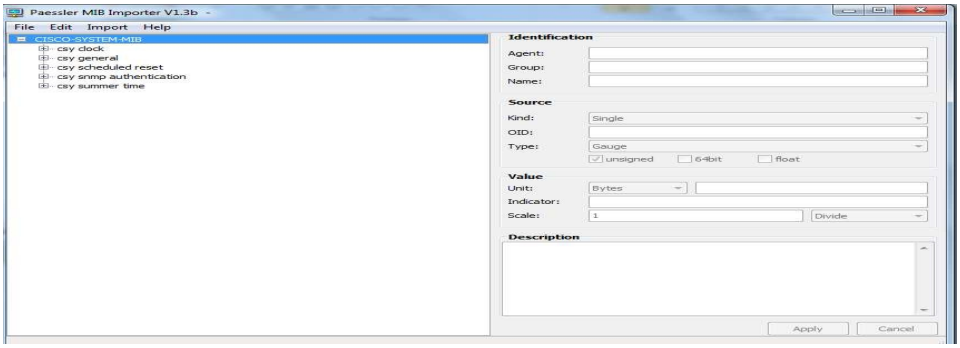


Figura 4.2.2(g)

7) Luego tenemos que proceder a grabar el archivo convertido en la base de informacion del PRTG, para ello selecionamos File -> Save As y lo guardamos en la siguiente direccion: C:\Program Files\PRTG Network Monitor\snmplibs\ y Guardar:

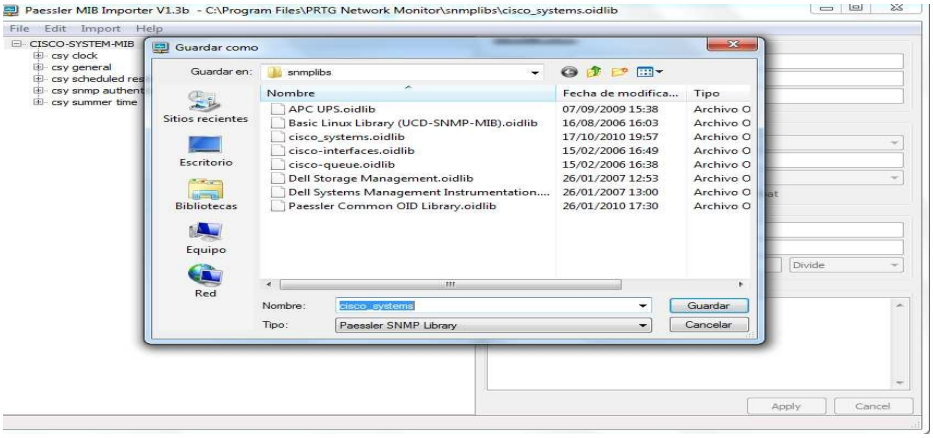


Figura 4.2.2(h)

CAPÍTULO 5

APLICACIÓN DEL SOFTWARE IMPLEMENTADO

5.1 PROCESO DE IMPLEMENTACIÓN DEL SNMP

Para el proceso de gestión y administración de la red FIEC se determinó que uno de los software más eficientes que aplican el protocolo SNMP es el PRTG.

5.1.1 APLICACIÓN IMPLEMENTADA EN LA RED

El sistema PRTG es una potente herramienta de monitorización de la Paessler AG que asegura la disponibilidad de componentes de red y mide el tráfico y el uso de la red. Ahorra costos ayudando a evitar fallos, optimizar conexiones, economizando tiempo de implementación y controlando acuerdos de nivel de servicio .

El programa opera 24horas, 7 días a la semana en una máquina basada en Windows, monitorizando parámetros de uso de red. Los datos de monitorización son guardados en una base de datos para poder generar reportes históricos.
El sistema tambien implementa SNMP (Simple Network Management Protocol) y WMI (Windows Management Instrumentation) que son usados para adquirir datos acerca del uso y el rendimiento de todos los sistemas que componen su red, incluyendo el uso de puertos individuales de switches y enrutadores.

5.1.2 UTILIZACION DE LA APLICACIÓN

Primero unos conceptos claves para entender el uso de PRTG:

Sensor:

Es un instrumento encargado de monitorear individualmente cada aspecto de un dispositivo de red.

Aparato:

Es un instrumento lógico que se ubica en cada punto de red (dirección IP) en donde se realiza el monitoreo. Cada aparato puede tener uno o más sensores.

Grupo:

Es el conjunto de aparatos que se pueden clasificar según sus características, las cuales ayudan al monitoreo más fácil y efectivo.

Proceso de Ejecución:

Después de instalar el programa en la computadora en la que se va a realizar el monitoreo de la red, se crea automáticamente un acceso directo en el escritorio en el cual hacemos doble clic para acceder a la interfaz de usuario para el registro.

Figura 5.1.2(a)

Ahora accedemos a la página de inicio de PRTG Network Monitor, en la cual se presentan varias opciones para empezar a usar el programa de manera sencilla.

Las opciones que nos interesan son: “Añadir sensores nuevos” y “Ejecutar descubrimiento automático”.

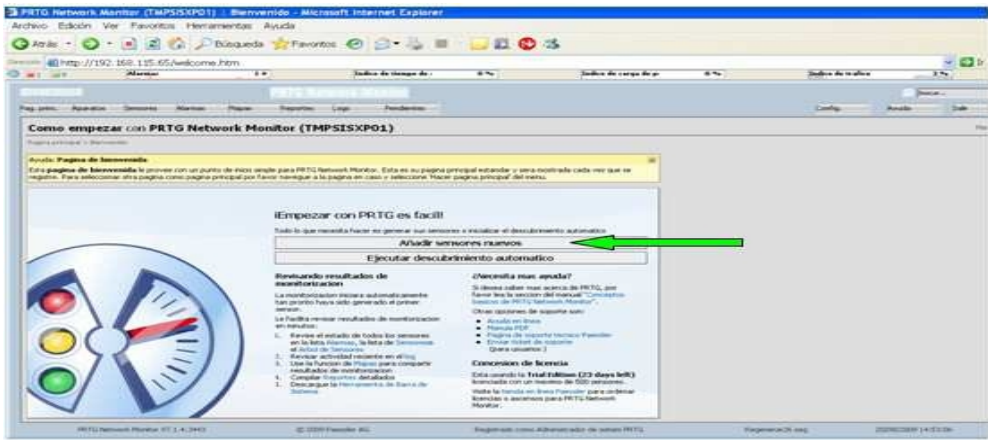


Figura 5.1.2(b)

Ejecutar descubrimiento automático:

Al elegir esta opción el sistema será el encargado de formar un único grupo genérico, en el cual añadirá la cantidad de aparatos necesaria para cada dispositivo y dentro de cada aparato todos los sensores que el punto de red permita instalar según ciertas configuraciones.

Esta es ideal para una red muy grande en la que se necesite saber todos los aspectos hasta el más mínimo de los dispositivos, pero su desventaja es que se necesita un conocimiento profundo de cómo analizar toda la información generada.

Añadir Aparatos Nuevos:

Esta es la opción que se va a usar para monitorear la red de la FIEC, ya que permite a uno mismo ir poniendo grupos, aparatos y sensores según la información que nosotros queremos generar y necesitemos.

Primero creamos un grupo general en el que se van a ordenar aparatos y sensores según nosotros deseamos.

En la misma pantalla, vamos a poner el mouse únicamente encima de la pestaña “Aparatos” (no hacer clic) y en el menú que se expande seleccionamos “Añadir Grupo”.

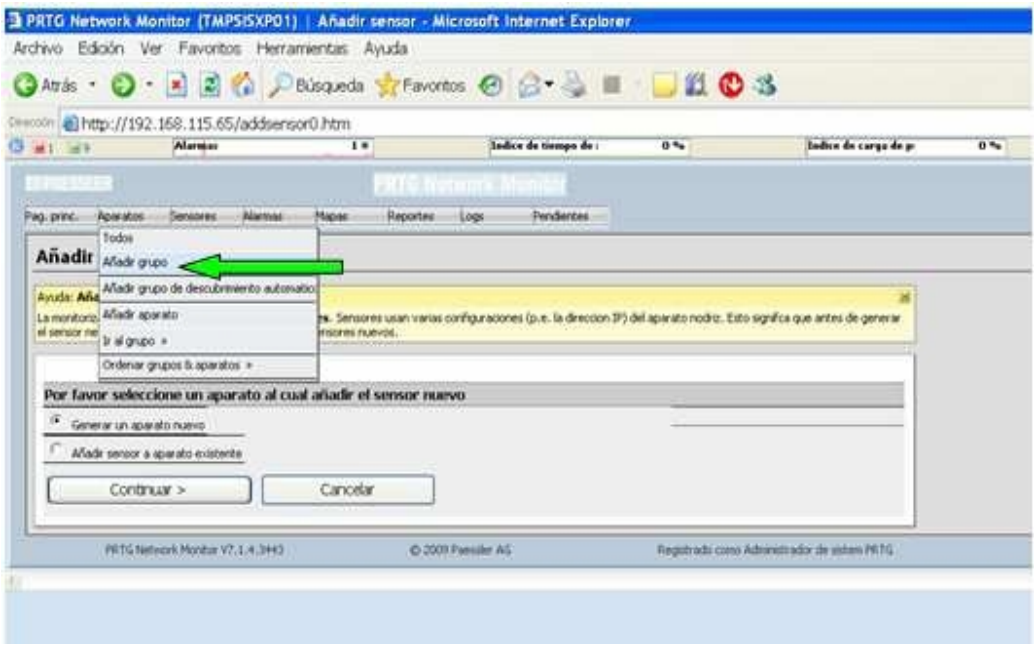


Figura 5.1.2(c)

Seleccionamos la opción Local Probe y damos clic en continuar.

Figura 5.1.2(d)

En la siguiente ventana le damos un nombre al grupo y si se desea se le puede asignar un identificador al grupo y damos clic en continuar.

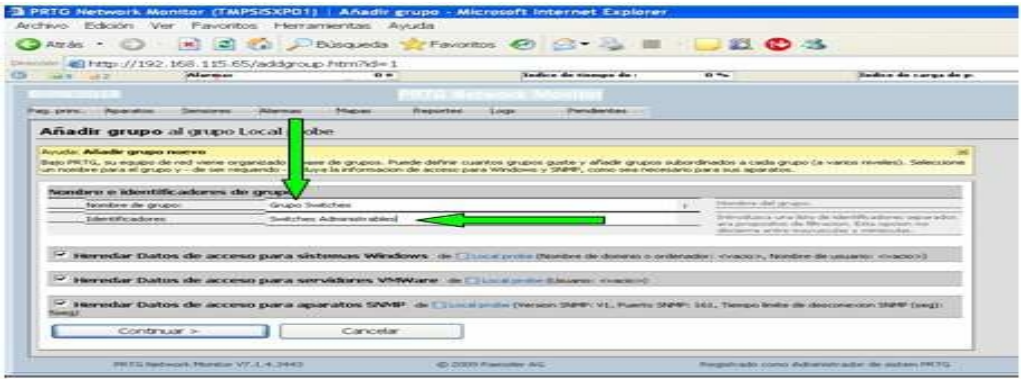


Figura 5.1.2(e)

Listo y se encuentra creado el grupo.

Para añadir Aparatos al grupo creado simplemente damos clic en “Añadir aparato” como se ve en la figura y se abrirá una nueva pantalla.

Figura 5.1.2(f)

Un aparto se debe colocar la IP a ser administrada por lo tanto, asignamos al nuevo aparato nombre, dirección IP y si se deseamos un identificador.

Además, en el manejo de sensores escogemos la opción “Manual (Sin descubrimiento automático)” y damos clic en continuar.

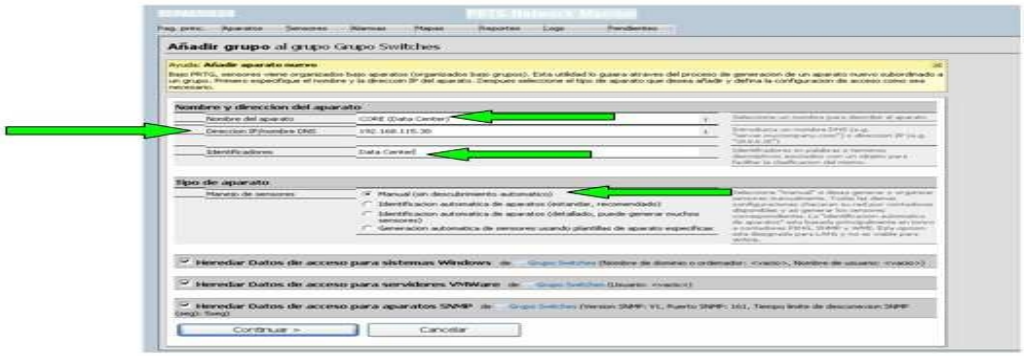


Figura 5.1.2(g)

Y listo el aparato a ser monitoreado se crea con su respectiva dirección IP.

5.2 CREACIÓN DEL MONITOREO

Para moniterar parte de la red fiec, especialmente en el laboratorio 6 se procedió a crear los siguientes sensores en algunas PC que forman parte de la red:

5.2.1 CREACION DE SENSORES

Sensor SNMP en Computadoras:

Para esto primero se tuvo que habilitar en las pc del laboratorio el protocolo SNMP que se mostró con anterioridad configurado un agente SNMP en cada computadora.

Después de eso se tiene crear el sensor SNMP para medir el Ancho de Banda consumido por cada equipo de la Red como se muestra a continuación:

Figura 5.2.1(a)

Luego en la sección Sensor type elegimos SNMP trafico

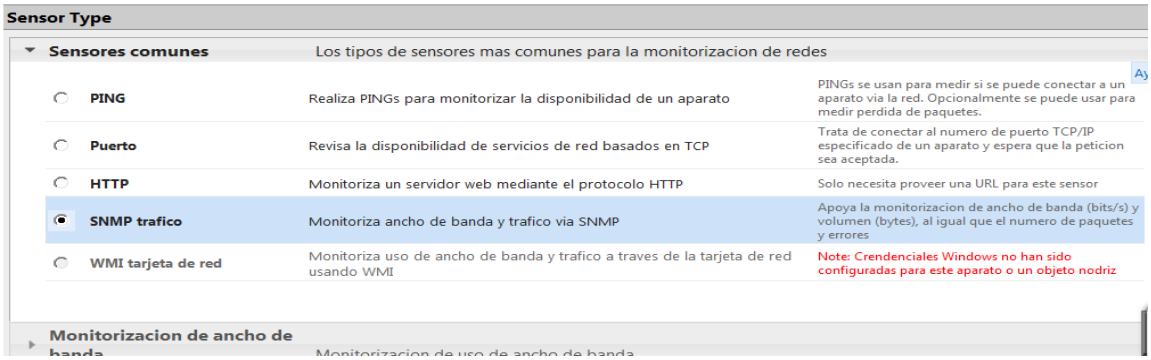


Figura 5.2.1(b)

Y luego nos muestra la sección en que muestra que el agente SNMP se configura

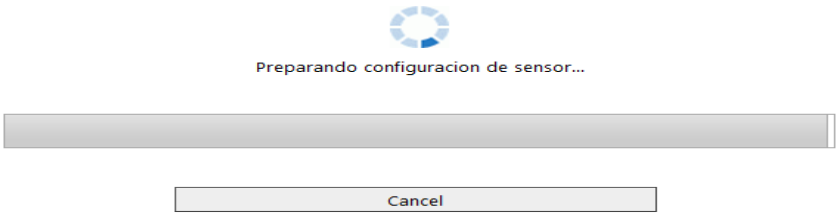


Figura 5.2.1(c)

Luego nos muestra el Identificador que seria el nombre del sensor y adicional tendremos que escoger en que tarjeta de red se realizaría el censo que sería la de velocidad 100Mbps o 1 Gbps y de tipo Ethernet esto varía dependiendo del tipo de tarjeta de red de la PC .

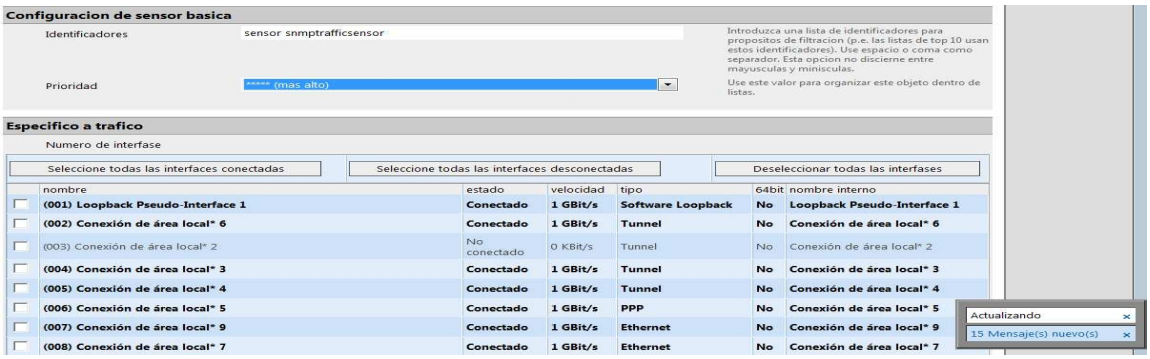


Figura 5.2.1(d)

En la parte inferior se eligue el tipo de sensor se puede tambien agregar para verificar errores de entrada In y salida Out al momento de establecer la conexión y click en continuar y queda agregado el sensor SNMP .

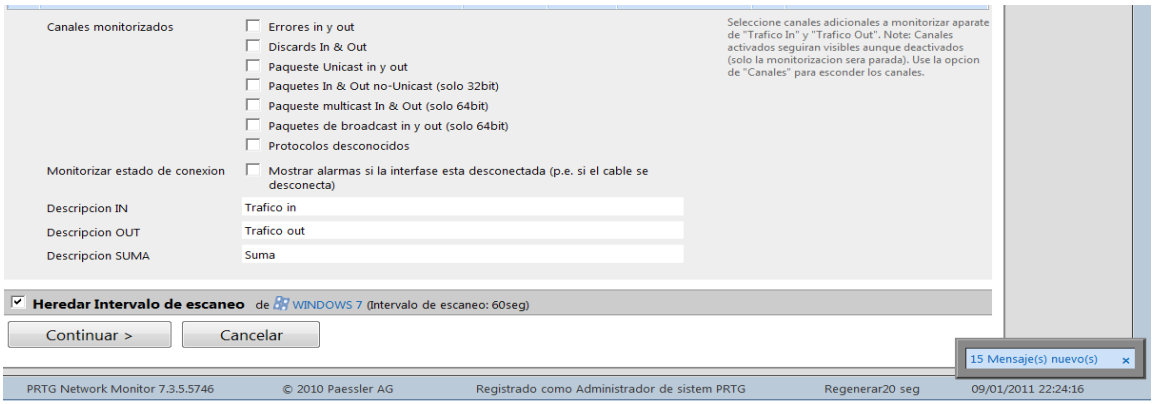


Figura 5.2.1(e)

5.2.2 CREACION DE SENSOR SNMP EN EQUIPOS CISCO

Para la creación de un sensor SNMP en equipos cisco se tiene que habilitar primero el Protocolo SNMP, a continuación se muestra como realizar la habilitación del SNMP en equipos CISCO:

Switch Cisco.

Ingresamos al switch con el cable de consola y empezamos y digitamos:

Switch>enable
Switch#configure terminal
Switch(config)#snmp-server enable traps

“Con esto habilitamos las interrupciones en el switch, para que avise al sistema sobre cualquier cambio a su estado.”

Switch(config)#snmp-server community public ro

“Especificamos la comunidad y damos permisos, ya sean de lectura escritura (rw) o de solo lectura (ro)”.

Así ya se habrá configurado el agente en nuestro switch.

Router Cisco.

Ingresamos a Router con el cable de consola y digitamos:

Router>enable

Router#configure terminal

Router(config)#snmp-server enable traps

"Con esto habilitamos las interrupciones en el router, para que avise al nms sobre cualquier cambio a su estado."

Router(config)#snmp-server community public ro

"Especificamos la comunidad y damos permisos, ya sean de lectura escritura (rw) o de solo lectura (ro)".

Así ya se habrá configurado el agente en nuestro router.

Luego se tiene que crear un equipo o aparato para ser monitoreado que previamente se explico cómo crear aparatos en el sistema PRTG (página 81 del proyecto)

Después de crear el aparato se procede a agregar el correspondiente sensor.

Después de haber habilitado el SNMP en los equipos CISCO se tiene que proceder a crear el sensor SNMP en el software PRTG así mismo previamente se tiene que crear el aparato con la respectiva dirección IP de equipo, luego cargar el correspondiente MIB del equipo.

5.3 VERIFICACIÓN DE EQUIPOS EN LA RED

Durante los días en que se realizó el monitoreo del ancho de banda que dispone la red fiec se determinó lo siguiente:

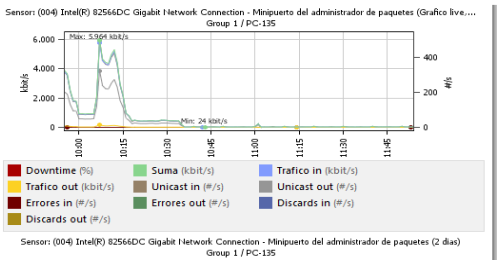
5.3.1 VERIFICACIÓN Y CONSUMO DEL ANCHO DE BANDA DE EQUIPOS:

A continuación se muestra los sensores SNMP creados en algunas de las computadoras del laboratorio:

Nombre de sensor	(004) Intel(R) 82566DC Gigabit Ne... (ID 2031)
Tipo (intervalo)	SNMP trafico (60 s)
Prioridad	★★★★★
Sonda nodriz	Local probe (Sonda remota en 127....
Grupo nodriz	Group 1
Aparato nodriz	PC-135

Estado

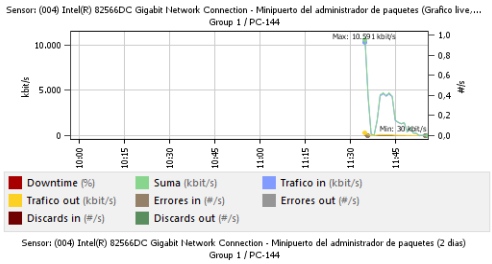
Ultimo mensaje	OK
Ultimo resultado	32 kbit/s
Ultimo escaneo	14/09/2010 11:53:47 [hace 26 s]
Ultimo disponible	14/09/2010 11:53:47 [hace 26 s]
Ultima falla	-
Tiempo disponible	100,0000% [2 h 2 m 37 s]
Tiempo de falla	0,0000% [0 s]
Total tiempo disponible & de falla	2 h 2 m 37 s [=100% cobertura]



Nombre de sensor	(004) Intel(R) 82566DC Gigabit Ne... (ID 2043)
Tipo (intervalo)	SNMP trafico (60 s)
Prioridad	★★★★★
Sonda nodriz	Local probe (Sonda remota en 127....
Grupo nodriz	Group 1
Aparato nodriz	PC-144

Estado

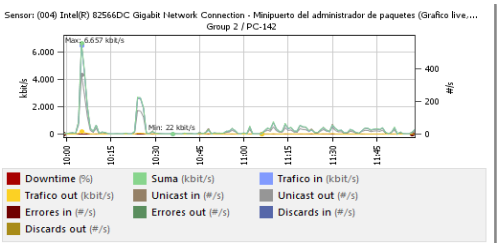
Ultimo mensaje	OK
Ultimo resultado	30 kbit/s
Ultimo escaneo	14/09/2010 11:55:41 [hace 25 s]
Ultimo disponible	14/09/2010 11:55:41 [hace 25 s]
Ultima falla	-
Tiempo disponible	100,0000% [21 m]
Tiempo de falla	0,0000% [0 s]

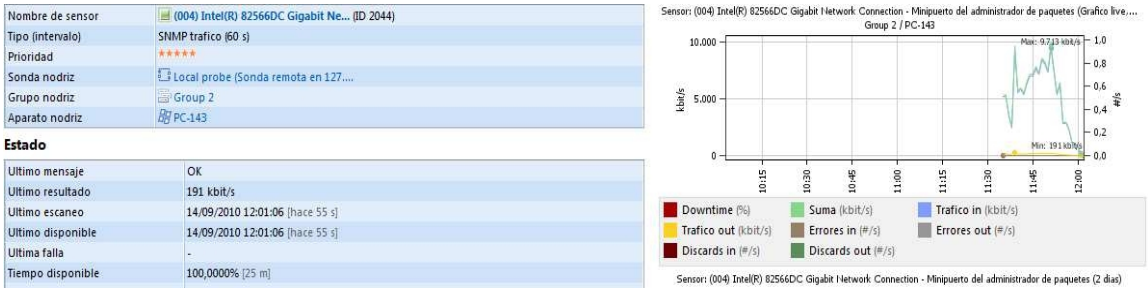


Nombre de sensor	(004) Intel(R) 82566DC Gigabit Ne... (ID 2032)
Tipo (intervalo)	SNMP trafico (60 s)
Prioridad	★★★★★
Sonda nodriz	Local probe (Sonda remota en 127....
Grupo nodriz	Group 2
Aparato nodriz	PC-142

Estado

Ultimo mensaje	OK
Ultimo resultado	206 kbit/s
Ultimo escaneo	14/09/2010 11:58:48 [hace 11 s]
Ultimo disponible	14/09/2010 11:58:48 [hace 11 s]
Ultima falla	-
Tiempo disponible	100,0000% [2 h 5 m 25 s]





Nombre de sensor	(004) Intel(R) 82566DC Gigabit Ne... (ID 2044)
Tipo (intervalo)	SNMP trafico (60 s)
Prioridad	*****
Sonda nodriz	Local probe (Sonda remota en 127....
Grupo nodriz	Group 2
Aparato nodriz	PC-143

Estado

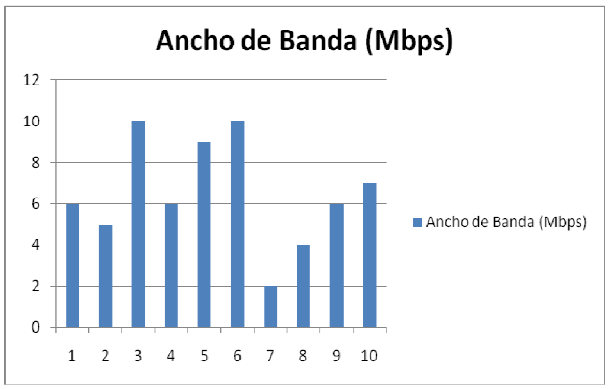
Ultimo mensaje	OK
Ultimo resultado	191 kbit/s
Ultimo escaneo	14/09/2010 12:01:06 [hace 55 s]
Ultimo disponible	14/09/2010 12:01:06 [hace 55 s]
Ultima falla	-
Tiempo disponible	100,0000% [25 m]

Figuras 5.3.1(a,b,c,d)

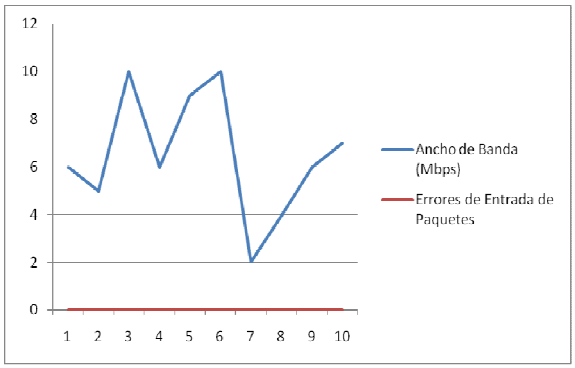
En la siguiente Tabla se muestra un resumen de los valores obtenidos con el Sensor SNMP:

PC	Ancho de Banda (Mbps)	Errores de Entrada de Paquetes	Errores de Salida de Paquetes
1	6	0	0
2	5	0,001	0
3	10	0	0
4	6	0	0
5	9	0	0
6	10	0	0,00001
7	2	0	0
8	4	0	0
9	6	0	0
10	7	0,0000001	0,0000001

Tabla 5.3.1(e)



X(PC) vs Y(BW)Tabla 5.3.1(f)



X(Errores de Entrada) vs Y(BW)Tabla 5.3.1(g)

5.3.2 VERFICACION DE EQUIPOS CONECTADOS EN LA RED

En la verificación de los equipos que se encuentran en la Red eso se lo determina al momento que se va creado los equipos a administrar.

Tal como se muestra el Grupo Root es donde se encuentra instalado el sistema y el resto se pueden crear grupos agrupados puede ser ROUTERS, SWITCHES, PC.

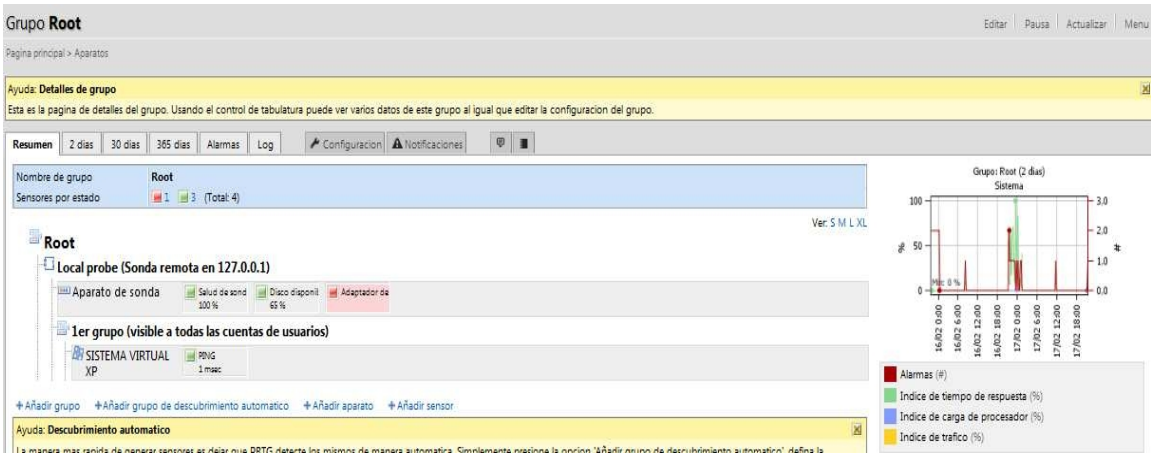


Figura 5.3.2(a)

La Dirección de Red del Laboratorio de la FIEC es: 200.9.176.0/24: Solo se pudo realizar la administración de la red con 10 computadores desde la maquina #135 hasta la maquina #144 .

El programa va censando cada 5 segundos que es lo que viene en la versión gratis(Tasa de muestreo) pero si se desea adquirir la versión por 1 año o más viene activado para censar cada segundo…Cuando esta todo bien osea cuando tiene respuesta del equipo a administar si pone de color verde pero si no tiene respuesta del equipo se muestra con color rojo y muestra un mensaje de error como se muestra:

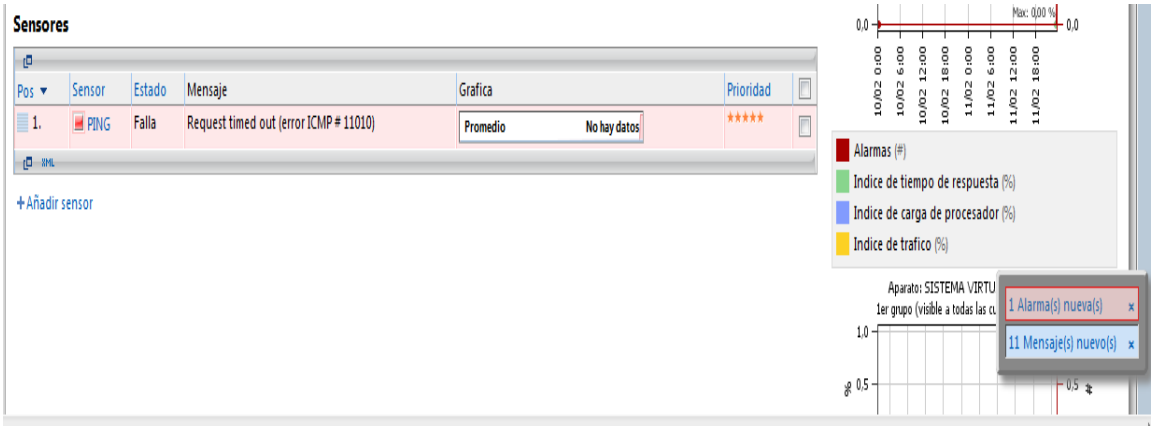


Figura 5.3.2(b)

5.3.3 VERIFICACIÓN DE UN OID ESPECÍFICO

- Luego de haber creado los grupos con los respectivos equipos a administrar con sus respectivos sensores a continuación mostramos como se veria un OID RF de un ROUTER INALAMBRICO el cual nos muestra el nivel nivel de radio frecuencia del enlace inalambrico.

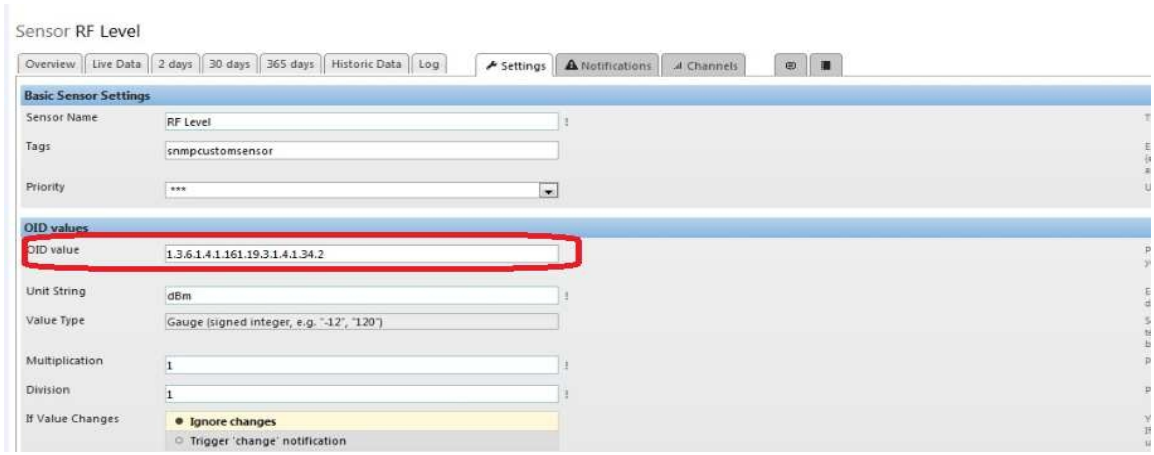


Figura 5.3.3(a)

Como se puede observar el nivel de radio frecuencia del enlace es 46dbm con el estado del último mensaje censado OK lo que quiere decir que el enlace esta operando normalmente.

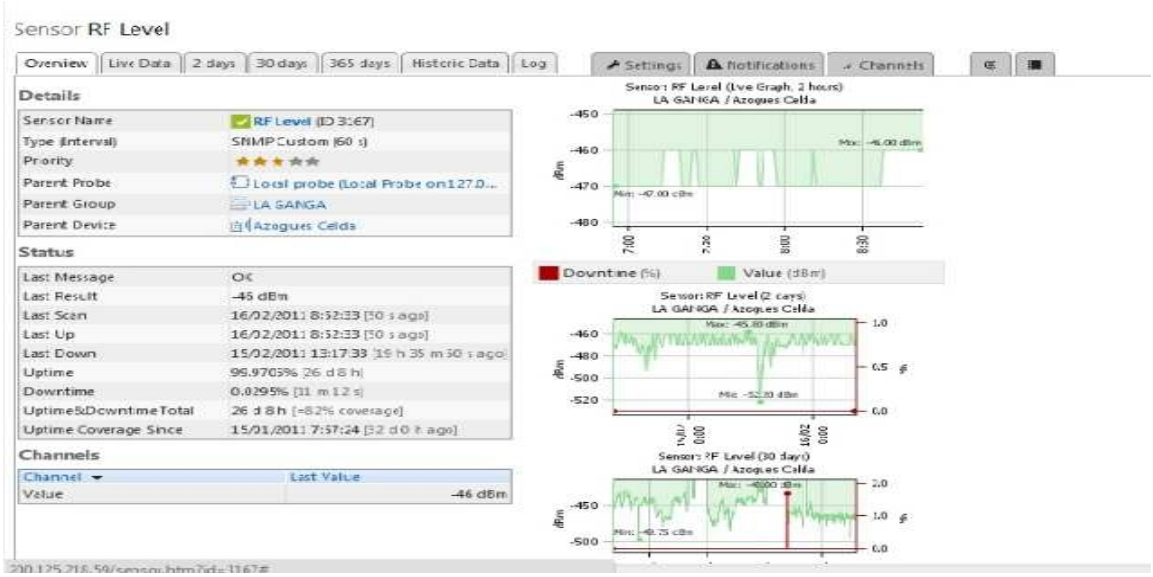


Figura 5.3.3(b)

Existe otra forma de poder acceder a un parámetro especifico dentro de los MIB se puede acceder por medio del nombre del campo que deseamos administrar en este caso se muestra en la figura un campo que se llama la capacidad tal como se muestra:

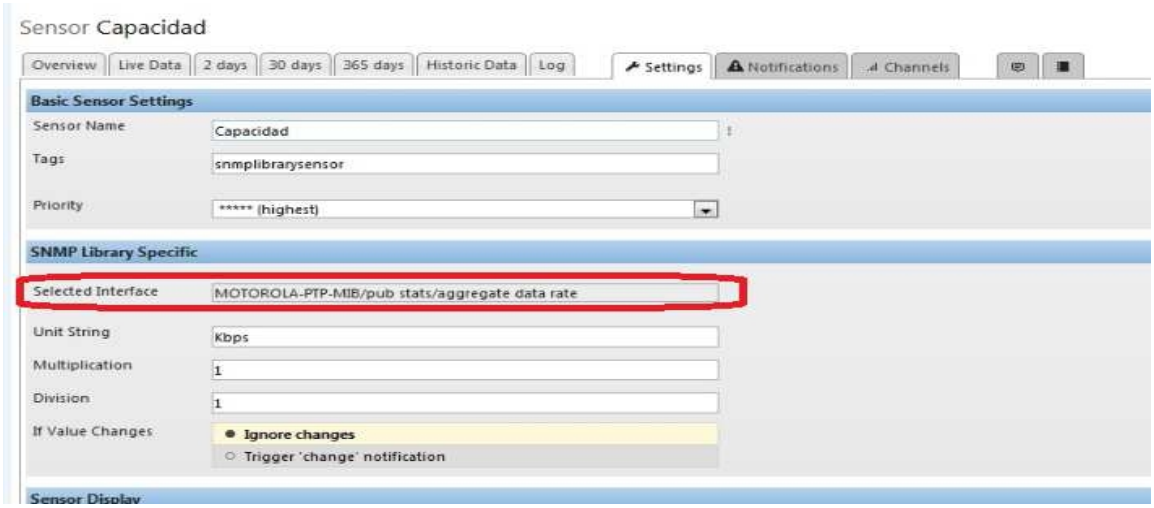


Figura 5.3.3(c)

Se puede observar que la capacidad del enlace es de 15010Kbps y adicional nos muestra es estado que es Ok que quiere decir que la capacidad es óptima para poder transmitir o recibir información

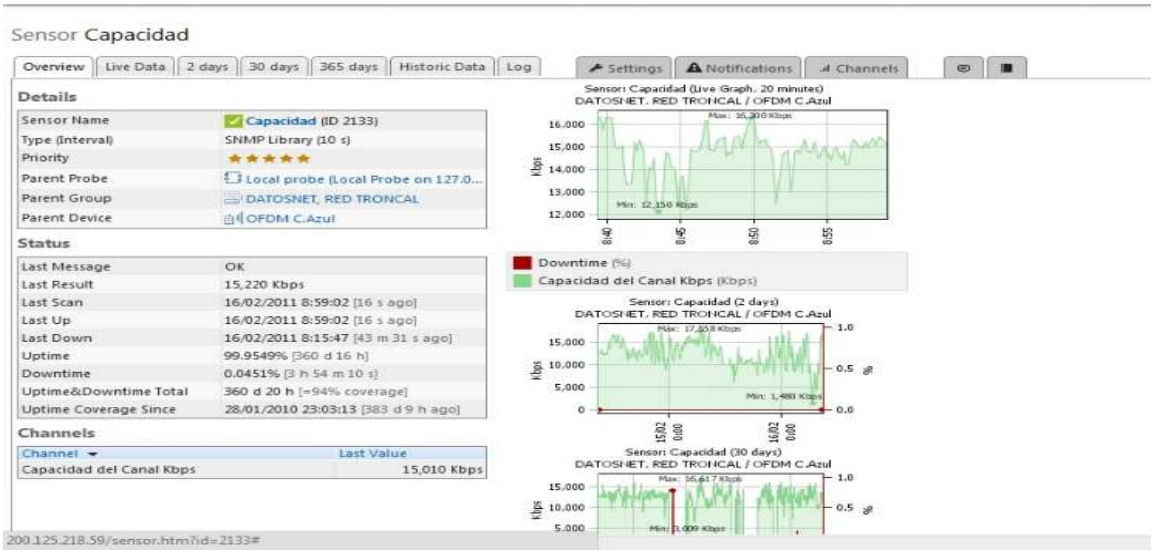


Figura 5.3.3(d)

5.4 ALERTAS Y GENERACIÓN DE REPORTES

Alertas: Las alertas son generadas al momento en que dentro de los equipos en la Red a administrar no responde cuando el sistema envía constantes peticiones para verificar que se encuentre en línea.

5.4.1 ALERTA DE EQUIPO CAIDO EN LA RED

En la sección superior se encuentra Alarmas donde se muestra todas las alarmas de los diferentes tipos de sensores que se haya agregado en la administración tal como se muestra:

Figura 5.4.1

Si deseamos saber cuales son los motivos de la alarma solo se da click sobre cualquiera de las alarmas generadas.

5.4.2 GENERACION DE REPORTES Y GRAFICOS.

Reportes: En esta sección se generan los eventos que se presentan en la administración de la red, estos reportes son administrables ósea se pueden crear cuando enviar por correo algún reporte de algún imprevisto, o ya sea guardando en el disco para luego ser revisado.

Para configurar que lleguen los reportes damos click en Reportes luego en Añadir Reporte como se muestra:

Figura 5.4.2(a)

Luego en la primera ventana de personalización del reporte esta ventana es grande por lo que enseñara por partes la edición de la misma.

Primero le damos un nombre al reporte, en plantilla debemos elegir la manera en cómo queremos generar la información. Sea esta con grafico y datos o únicamente grafico, además podemos seleccionar el rango en el que queremos la información.

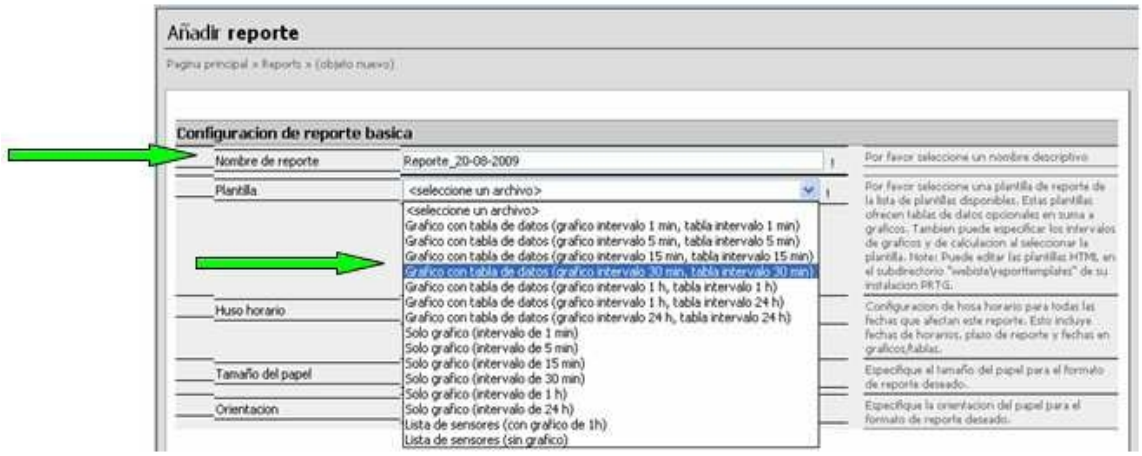


Figura 5.4.2(b)

A continuación, especificamos el uso horario, tamaño de papel y orientación que deseamos.

Figura 5.4.2(c)

La siguiente parte de la configuración es de fácil intuición y queda a criterio del administrador que va a generar el reporte. Si nos damos cuenta podemos marcar la sección de enviar por correo reporte.

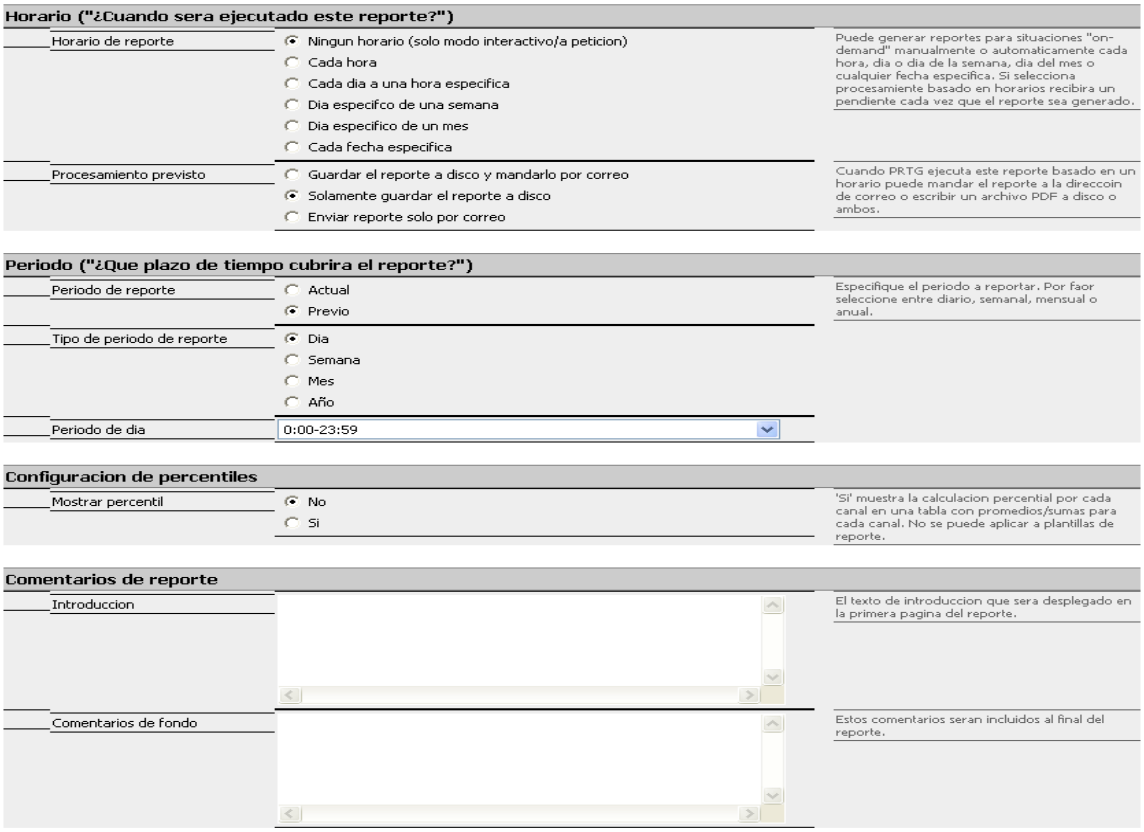


Figura 5.4.2(d)

Ahora si vamos a seleccionar los sensores que queremos incluir en el reporte. Esta opción es la elección manual de sensores.

En la parte inferior de la pantalla están todos los sensores que podemos añadir al reporte, para agregarlos se da clic en añadir, estos sensores que se selecciona se van adjuntando en la parte superior de la pantalla y en caso de ya no querer incluirlos en el reporte damos clic en eliminar y se quitaran de ahí.

Y click en continuar y nos muestra lo siguiente y elegimos Ejecutar Ahora

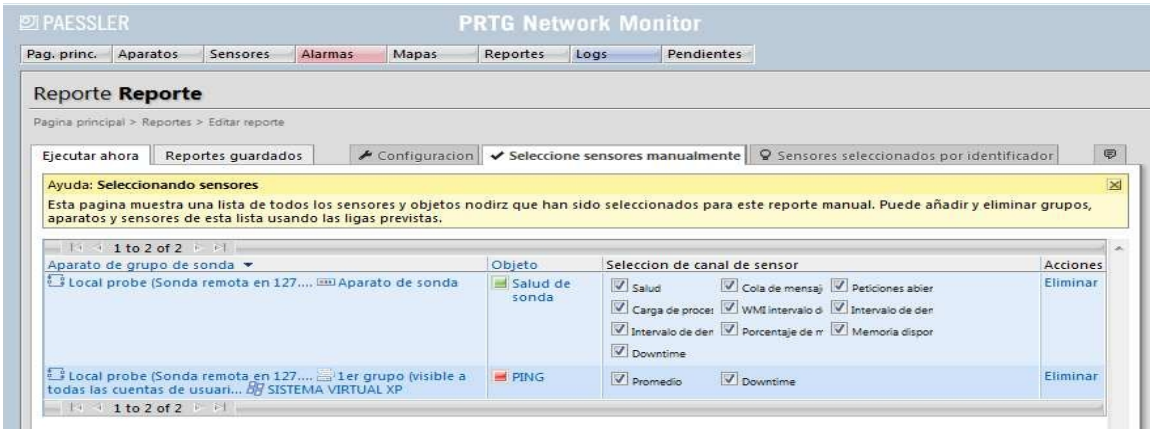


Figura 5.4.2(e)

En la siguiente ventana vamos a escoger el día del que deseamos obtener el reporte y el tipo de archivo que deseamos generar.

Figura 5.4.2(f)

Luego seleccionamos Generar, guardar Enviar archivo PDF y listo se configura los reportes.

CONCLUSIONES:

1) En la realización de este proyecto se pudo entender como trabaja el Protocolo de Administración de Redes SNMP este es de suma utilidad porque la administración de equipos es más fiable.

2) En las prácticas que se pudo realizar en la red de la fiec se pudo determinar el ancho de banda destinado haciendo uso del protocolo SNMP.

3) La diferencia entre Realizar un Testeo de Ancho de Banda con SNMP y un testeo normal de Ancho de banda, es que con SNMP este evalúa si existe algún error al recibir información de la red, mientras que cuando se realiza un testeo solo a través de otros software solo se verifica cuanta información fluye sin importar si existe errores.

4) Con el SNMP se crea un agente especialmente diseñado para administrar la red que se encarga de recorrer el Árbol MIB que posee la información detallada de cada equipo .

5) En algunos de los sistemas operativos que existen solo vienen implementada la versión 2 del protocolo SNMP debido a que es la manera más rápida y sencilla de realizar la administración.

6) Para realizar una Administración de una PC con la versión 3 del SNMP se tiene que descargar actualizaciones para cada sistema operativo en donde se agregan más seguridades al momento de acceder a la pc.

7) Con el uso del SNMP se puede especificar un determinado parámetro llamado OID que se desee administrar con esto se logró optimizar el uso de la Red al momento de detectar una falla en algún equipo.

8) Debido a que el protocolo IP es un servicio No Orientado a Conexión no se puede saber si existe fallas del paquete al momento de transferir información por ese motivo se hace uso del protocolo SNMP es cual consta en sus versiones con un parámetro que censa si existió problema al transmitir la trama que es una de las formas más sencillas de saber los errores de entrada y salida del paquete de información.

9) Los traps se generan al momento en que fallas algún equipo de la red dado el sensor que se crear para la administración.

10) Para realizar una administración de redes basados en el protocolo SNMP se la puede realizar mediante software que implementen el protocolo, o mediante hardware con equipo que especialmente son diseñados para determinadas funciones.

RECOMENDACIONES:

1) Se recomienda que al momento de querer administrar algún equipo de la red como routers o switches se debe primero tener el árbol MIB del equipo para después transfórmalo a un archivo reconocible por el software PRTG.

2) Antes de proceder a implementar alguna administración de red se tiene que saber que parámetros se va a tomar en cuenta al momento de verificar algún equipo de la red, por ejemplo se considero en este proyecto parámetros con la temperatura, errores de entrada y salida, relación de señal a ruido estos se encuentran en el arbolo MIB almacenados y nos ayudan a poder prevenir posibles falla del equipo.

3) Elegir que versión de SNMP se va a trabajar en la administración de la red, cada versión tiene características diferentes de seguridad.

4) Si se desea implementar administración de red basado en el protocolo SNMP se tiene que saber elegir equipos que soporten dicho protocolo, y elegir si se va a administrar mediante software o mediante hadware.

5) La ventaja fundamental de usar SNMP es que su diseño es simple por lo que su implementación es sencilla en grandes redes y la información de gestión que se necesita intercambiar ocupa pocos recursos de la red.

6) Además, nos permite al usuario elegir las variables que desea monitorizar.

7) Otra ventaja de SNMP es que en la actualidad es el sistema más extendido. Ha conseguido su popularidad debido a que fue el único protocolo que existió en un principio y por ello casi todos los fabricantes de dispositivos como puentes y enrutadores diseñan sus productos para soportar SNMP

8) Una ventaja del PRTG sobre el Nagios es al momento de administrar una red basado en el SNMP se tiene que configurar el nagios bajo comandos para que haga uso del SNMP en cambio en PRTG no se tiene que configurar porque ya viene habilitado el servicio.

9) La deficiencia de SNMP es que tiene grandes fallas de seguridad que pueden permitir a intrusos acceder a información que lleva la red. Y todavía peor, porque los intrusos pueden llegar a bloquear o deshabilitar terminales de los equipo de la red.

10) El problema del SNMP es que se considera tan simple que la información está poco organizada, lo que no lo hace muy acertado para gestionar las grandes redes de la actualidad. Esto se debe en gran parte a que SNMP se creó como un protocolo provisional y no ha sido sustituido por otro de entidad, de la misma manera se los soluciona con la evolución de la versiones.

ANEXOS:

Acontinuacion se muestran los MIB de algunos equipos cisco

- MIB CISCO SYSTEM:

```
-- MIB file created 27-Jun-2001 15:18:12, by
--      SMICng version 2.2.11-beta(PRO)(Solaris), January 20, 2001. Enterprise key cisco.com

CISCO-SYSTEM-MIB DEFINITIONS ::= BEGIN

-- From file: "CISCO-SYSTEM-MIB.my"
-- Compile options "4 7 F H N W 03 06 0B 0G 0N 0T"

IMPORTS
    DateAndTime, TruthValue, DisplayString
        FROM SNMPv2-TC-v1
    CountryCode
        FROM CISCO-TC
    InetAddressType, InetAddress
        FROM INET-ADDRESS-MIB
    ciscoMgmt
        FROM CISCO-SMI
    Counter
        FROM RFC1155-SMI
    OBJECT-TYPE
        FROM RFC-1212
    TRAP-TYPE
        FROM RFC-1215;

ciscoSystemMIB OBJECT IDENTIFIER ::= { ciscoMgmt 131 }
-- MODULE-IDENTITY
```

```
-- LastUpdated
--    200106220000Z
-- OrgName
--    Cisco Systems, Inc.
-- ContactInfo
--    Cisco Systems
--     Customer Service
--
--     Postal: 170 W Tasman Drive
--     San Jose, CA  95134
--     USA
--
--     Tel: +1 800 553-NETS
--
--     E-mail: cs-snmp@cisco.com
-- Descr
--    The systemGroup (see RFC 1907) provides a standard set of
--    basic system information.  This MIB module contains
--    Cisco-defined extensions to the systemGroup.
-- RevDate
--    200106220000Z
-- RevDescr
--    Added SNMP authentication failure objects and clock
--    changed notification.
-- RevDate
--    200001251700Z
-- RevDescr
--    Added Summertime and ScheduledReset objects.
-- RevDate
--    9902021700Z
-- RevDescr
--    Initial version of this MIB module.
```

```
ciscoSystemMIBObjects OBJECT IDENTIFIER ::= { ciscoSystemMIB 1 }
csyClock        OBJECT IDENTIFIER ::= { ciscoSystemMIBObjects 1 }
csyLocation     OBJECT IDENTIFIER ::= { ciscoSystemMIBObjects 2 }
csySummerTime   OBJECT IDENTIFIER ::= { ciscoSystemMIBObjects 3 }
csyScheduledReset OBJECT IDENTIFIER ::= { ciscoSystemMIBObjects 4 }
csySnmpAuthentication OBJECT IDENTIFIER ::= { ciscoSystemMIBObjects 5 }
csyGeneral      OBJECT IDENTIFIER ::= { ciscoSystemMIBObjects 6 }
ciscoSystemMIBNotificationPrefix OBJECT IDENTIFIER ::= { ciscoSystemMIB 2 }
ciscoSystemMIBNotifications OBJECT IDENTIFIER ::= {
ciscoSystemMIBNotificationPrefix 0 }
ciscoSystemMIBConformance OBJECT IDENTIFIER ::= { ciscoSystemMIB 3 }
ciscoSystemMIBCompliances OBJECT IDENTIFIER ::= {
ciscoSystemMIBConformance 1 }
ciscoSystemMIBGroups OBJECT IDENTIFIER ::= { ciscoSystemMIBConformance 2 }

csyClockDateAndTime OBJECT-TYPE
   SYNTAX DateAndTime
--    Rsyntax OCTET STRING(SIZE(8|11))
   ACCESS read-write
   STATUS mandatory
   DESCRIPTION
      "The current local date and time for the system.

      Setting this object is equivalent to setting an automated
      clock and calendar.  The value of the object will track the
      date and time from the value set.  Note that due to hardware
      limitations some systems may not be able to preserve such
      meaning across reboots of the system, as indicated by
      csyClockLostOnReboot.

      A constant value of all zeros and length 8 indicates the
      system is not aware of the present date and time.
```

```
        This object may be read-only on some systems."
    ::= { csyClock 1 }

csyClockLostOnReboot OBJECT-TYPE
    SYNTAX TruthValue
--    Rsyntax INTEGER {
--        true(1),
--        false(2)
--        }
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "Indication of whether the system can preserve knowledge of
        current date and time across a system reboot.

        A value of 'true' indicates the clock must be reset from some
        external source each time the system reboots.

        A value of 'false' indicates the system has the ability to
        keep time across reboots."
    ::= { csyClock 2 }

csyLocationCountry OBJECT-TYPE
    SYNTAX CountryCode
--    Rsyntax OCTET STRING(SIZE(0|2))
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The country where the system is physically located.

        On some systems and for some technologies this value affects
        behavior, such as standards for communication.  All such
        technologies should default to using the setting of this
```

```
        value, but may provide an override if necessary.

        The default value of this object is 'US'.  Systems destined
        for other countries may use a different default.  Systems in
        which the value does not affect operation should default to
        a zero-length value."
    ::= { csyLocation 1 }

csySummerTimeStatus OBJECT-TYPE
    SYNTAX TruthValue
--    Rsyntax INTEGER {
--        true(1),
--        false(2)
--        }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "An indication of whether the summertime feature
         is enabled on this device. When this object
         is set to true, then csySummerTimeOffset,
         csySummerTimeRecurringStart and
         csySummerTimeRecurringEnd  objects are set
         to default values. When this object is set
         to false, then csySummerTimeOffset,
         csySummerTimeRecurringStart,
         csySummerTimeRecurringEnd objects are
         not instantiated and the summertime feature
         is disabled"
    DEFVAL { false }
    ::= { csySummerTime 1 }

csySummerTimeOffset OBJECT-TYPE
    SYNTAX INTEGER(1..1440)
```

```
-- Units
--   Minutes
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The value of this object indicates number of
         minutes to add  or to subtract during summertime.
         This object is not instantiated when
         csySummerTimeStatus object is set to false."
    DEFVAL { 60 }
    ::= { csySummerTime 2 }

csySummerTimeRecurringStart OBJECT-TYPE
    SYNTAX OCTET STRING(SIZE(6))
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "Indicates summertime starts at this time every year.
         octets    contents     range
         1         week        1..5,ff last = ff
         2-3       day         1..7
                               where sunday = 1 saturday = 7
         4         month       1..12
                               where january = 1 december = 12
         5         hour        0..23
         6         min         0..59
         For example, the first Monday in Feb at 13:30pm
         should be given as
            01 00 02 02 0e 1e
         For the last Tuesday in dec at 1:20am should be
         given as
           ff 00 03 0c 01 14
         This object is not instantiated when
```

```
        csySummerTimeStatus object is set to false."
    DEFVAL { '010001040200'H }
    ::= { csySummerTime 3 }

csySummerTimeRecurringEnd OBJECT-TYPE
    SYNTAX OCTET STRING(SIZE(6))
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
       "Indicates summertime ends at this time every year.
        octets    contents      range
        1         week        1..5,ff where ff = last
        2-3       day         1..7
                              where sunday = 1 saturday = 7
        4         month       1..12
                              where january = 1 december = 12
        5         hour        0..23
        6         min         0..59
       For example, the third friday in February at 3:30am
       should be given as
            03 00 06 02 03 1e

        For the first Tuesday in May at 1:20am  should
        be given as
           01 00 03 05 01 14
        This object is not instantiated when
        csySummerTimeStatus object is set to false."
    DEFVAL { 'ff00010a0200'H }
    ::= { csySummerTime 4 }

csyScheduledResetTime OBJECT-TYPE
    SYNTAX DateAndTime
--    Rsyntax OCTET STRING(SIZE(8|11))
```

```
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The scheduled date and time the switch will be
         reset at. The system will only take octet strings
         with length 8 for this object which indicates the
         local time of the switch. The maximum scheduled
         time is 24 days from the current system clock time.

         Setting this object value to be before the current
         system clock time or beyond the maximum scheduled
         time limit will be rejected by the system. Setting
         the object to all-zero octet strings will cancel the
         previously scheduled reset time and then the system
         will have no pending scheduled reset time. Setting
         this object value to be any valid octet strings
         other than the above cases will override the
         previously scheduled reset time and cause the system
         to be reset at the newly specified time.


        After the system has accepted the scheduled reset
        time, if the system clock is advanced ahead of the
        scheduled reset time, then reset will happen
        approximately 5 minutes after the current clock."
    ::= { csyScheduledReset 1 }

csyScheduledResetAction OBJECT-TYPE
    SYNTAX INTEGER {
        reset(1),
        resetMinDown(2)
        }
    ACCESS read-write
```

```
    STATUS mandatory
    DESCRIPTION
        "Writing reset(1) to this object perform the normal
        reset operation on the active supervisor module.

        Writing resetMinDown(2) to this object resets the
        system with the minimal system down time at the
        scheduled time. The resetMinDown(2) is only
        supported in systems with redundant supervisors."
    DEFVAL { reset }
    ::= { csyScheduledReset 2 }

csyScheduledResetReason OBJECT-TYPE
    SYNTAX DisplayString
--    Rsyntax OCTET STRING(SIZE(0..255))
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "Indicates the reason users input when issuing
        system's scheduled reset.  After the system
        is reset, this object value  will be empty octet
        string."
    ::= { csyScheduledReset 3 }

csySnmpAuthFail OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The total number of SNMP messages received by the SNMP engine
        that were not properly authenticated."
    ::= { csySnmpAuthentication 1 }
```

```
csySnmpAuthFailAddressType OBJECT-TYPE
    SYNTAX InetAddressType
--    Rsyntax INTEGER {
--        ?? enum value of zero may cause problems
--        unknown(0),
--        ipv4(1),
--        ipv6(2),
--        dns(16)
--        }
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The type of Internet address by which the last received
        SNMP message that is not properly authenticated.

        The value of this object is irrelevant if the value of
        csySnmpAuthFail is zero."
    ::= { csySnmpAuthentication 2 }

csySnmpAuthFailAddress OBJECT-TYPE
    SYNTAX InetAddress
--    Rsyntax OCTET STRING(SIZE(0..255))
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The internet address of the SNMP entity which sent the
        last received SNMP message that is not properly authenticated.

        The value of this object is irrelevant if the value of
        csySnmpAuthFail is zero."
    ::= { csySnmpAuthentication 3 }

csyNotificationsEnable OBJECT-TYPE
```

```
    SYNTAX TruthValue
--    Rsyntax INTEGER {
--        true(1),
--        false(2)
--        }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "This object indicates whether the system produces the
        notifications defined by the ciscoSystemNotificationsGroup.
        A false value will prevent notifications from being generated
        by this system."
    DEFVAL { false }
    ::= { csyGeneral 1 }

ciscoSystemClockChanged TRAP-TYPE
-- Reverse mappable trap
    ENTERPRISE ciscoSystemMIBNotificationPrefix
    VARIABLES {
        csyClockDateAndTime }
--  Status
--    mandatory
    DESCRIPTION
        "A clock changed notification is generated when the current
        local date and time for the system has been manually changed.
        The value of csyClockDateAndTime reflects new date and time."
    ::= 1

ciscoSystemClockGroup OBJECT IDENTIFIER ::= { ciscoSystemMIBGroups 1 }
-- OBJECT-GROUP
--  Status
--    mandatory
--  Descr
```

```
--    Clock attributes.
--  objects
--    csyClockDateAndTime, csyClockLostOnReboot

ciscoSystemLocationGroup OBJECT IDENTIFIER ::= { ciscoSystemMIBGroups 2 }
-- OBJECT-GROUP
--  Status
--    mandatory
--  Descr
--    Physical location attributes.
--  objects
--    csyLocationCountry

ciscoSystemSummerTimeGroup OBJECT IDENTIFIER ::= { ciscoSystemMIBGroups 3
}
-- OBJECT-GROUP
--  Status
--    mandatory
--  Descr
--    A collection of objects used to set Summertime.
--
--     Implementation of this group is optional. If the
--     generic Summertime feature is supported, the entire
--     group should be implemented.
--  objects
--    csySummerTimeStatus, csySummerTimeOffset,
--    csySummerTimeRecurringStart, csySummerTimeRecurringEnd

ciscoSystemScheduledResetGroup OBJECT IDENTIFIER ::= {
ciscoSystemMIBGroups 4 }
-- OBJECT-GROUP
--  Status
--    mandatory
```

```
--  Descr
--    A collection of objects used to set scheduled reset time.
--
--    Implementation of this group is optional. If the system
--    scheduled reset feature is supported, the entire group
--    should be implemented.
--  objects
--    csyScheduledResetTime, csyScheduledResetAction,
--    csyScheduledResetReason

ciscoSystemSnmpAuthGroup OBJECT IDENTIFIER ::= { ciscoSystemMIBGroups 5 }
-- OBJECT-GROUP
--  Status
--    mandatory
--  Descr
--    A collection of objects which provide information
--    about SNMP message that is not properly authenticated.
--  objects
--    csySnmpAuthFail, csySnmpAuthFailAddressType,
--    csySnmpAuthFailAddress

ciscoSystemGeneralGroup OBJECT IDENTIFIER ::= { ciscoSystemMIBGroups 6 }
-- OBJECT-GROUP
--  Status
--    mandatory
--  Descr
--    A collection of objects which provide information
--    about general configuration within this MIB module.
--  objects
--    csyNotificationsEnable

ciscoSystemNotificationsGroup OBJECT IDENTIFIER ::= { ciscoSystemMIBGroups 7 }
-- NOTIFICATION-GROUP
```

```
--  Status
--    mandatory
--  Descr
--    A collection of notifications in this MIB module.
--  notifications
--    ciscoSystemClockChanged

ciscoSystemMIBCompliance OBJECT IDENTIFIER ::= { ciscoSystemMIBCompliances
1 }
-- MODULE-COMPLIANCE
--  Status
--    deprecated
--  Descr
--    The compliance statement for entities which implement
--    the Cisco System MIB.  Adherence to this compliance
--    statement is expected of all Cisco systems.
--  Module
--    >>current<<
--   MandGroup
--    ciscoSystemClockGroup
--   MandGroup
--    ciscoSystemLocationGroup
--   ObjVar
--    csyClockDateAndTime

ciscoSystemMIBCompliance2 OBJECT IDENTIFIER ::= {
ciscoSystemMIBCompliances 2 }
-- MODULE-COMPLIANCE
--  Status
--    deprecated
--  Descr
--    The compliance statement for entities which implement
--    the Cisco System MIB.  Adherence to this compliance
```

```
--    statement is expected of all Cisco systems.
--  Module
--    >>current<<
--   MandGroup
--    ciscoSystemClockGroup
--   MandGroup
--    ciscoSystemLocationGroup
--   ObjVar
--    csyClockDateAndTime
--   OptGroup
--    ciscoSystemSummerTimeGroup
--   OptGroup
--    ciscoSystemScheduledResetGroup

ciscoSystemMIBCompliance3 OBJECT IDENTIFIER ::= {
ciscoSystemMIBCompliances 3 }
-- MODULE-COMPLIANCE
--  Status
--    mandatory
--  Descr
--    The compliance statement for entities which implement
--    the Cisco System MIB.  Adherence to this compliance
--    statement is expected of all Cisco systems.
--  Module
--    >>current<<
--   MandGroup
--    ciscoSystemClockGroup
--   MandGroup
--    ciscoSystemLocationGroup
--   ObjVar
--    csyClockDateAndTime
--   OptGroup
--    ciscoSystemSummerTimeGroup
```

```
--   OptGroup
--    ciscoSystemScheduledResetGroup
--   OptGroup
--    ciscoSystemSnmpAuthGroup
--   ObjVar
--    csySnmpAuthFailAddressType
--   ObjVar
--    csySnmpAuthFailAddress
--   OptGroup
--    ciscoSystemGeneralGroup
--   OptGroup
--    ciscoSystemNotificationsGroup


END
```

- MIB SERIES 2900:

```
-- MIB created 2/09/100 15:55:24, by
--   SMIC (the next generation) version 1.6.29, November 22, 1994.

CISCO-C2900-MIB DEFINITIONS ::= BEGIN

-- From file: "CISCO-C2900-MIB.my"

IMPORTS
    Counter32, Gauge32, Integer32
        FROM SNMPv2-SMI-v1
    OBJECT-TYPE
        FROM RFC-1212
    TRAP-TYPE
        FROM RFC-1215
    DateAndTime, TruthValue
```

```
        FROM SNMPv2-TC-v1
    InterfaceIndex
        FROM IF-MIB
    ciscoMgmt
        FROM CISCO-SMI
    DisplayString
        FROM RFC1213-MIB;

ciscoC2900MIB OBJECT IDENTIFIER ::= { ciscoMgmt 87 }
-- MODULE-IDENTITY
--  LastUpdated
--    9804290000Z
--  OrgName
--    Cisco Systems, Inc.
--  ContactInfo
--    Postal: Cisco Systems, Inc.
--     170 West Tasman Drive
--     San Jose, CA 95134-1706
--     USA
--
--     Tel: +1 800 553-NETS
--
--     E-mail: switchsnmp@cisco.com
--  Descr
--    The MIB module for Catalyst 2900 enterprise specific information.
--  RevDate
--    9804300000Z
--  RevDescr
--    Added the c2900PortNoMonitorDestinationPort MIB object
--     to remove a port from the monitored list.
--
--     Removed enumerated value securityDynamic(3) from the
--     MIB object c2900PortUsageApplication
```

```
c2900MIBObjects OBJECT IDENTIFIER ::= { ciscoC2900MIB 1 }
c2900SysInfo    OBJECT IDENTIFIER ::= { c2900MIBObjects 1 }
c2900SysConfig  OBJECT IDENTIFIER ::= { c2900MIBObjects 2 }
c2900Port       OBJECT IDENTIFIER ::= { c2900MIBObjects 4 }
c2900BandwidthUsage OBJECT IDENTIFIER ::= { c2900MIBObjects 5 }
c2900MibNotifications OBJECT IDENTIFIER ::= { ciscoC2900MIB 2 }
c2900MibNotificationsPrefix OBJECT IDENTIFIER ::= { c2900MibNotifications 0 }
c2900MIBConformance OBJECT IDENTIFIER ::= { ciscoC2900MIB 3 }
c2900MIBCompliances OBJECT IDENTIFIER ::= { c2900MIBConformance 1 }
c2900MIBGroups  OBJECT IDENTIFIER ::= { c2900MIBConformance 2 }

c2900InfoBoardRevision OBJECT-TYPE
   SYNTAX Integer32
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
       "Returns the revision number of the main board
        on which the FastSwitch firmware resides."
   ::= { c2900SysInfo 1 }

c2900InfoPeakBuffersUsed OBJECT-TYPE
   SYNTAX Gauge32
--  Units
--    buffers
   ACCESS read-only
   STATUS mandatory
   DESCRIPTION
       "The maximum number of 64-byte buffers used in the
        main switch buffer pool."
   ::= { c2900SysInfo 2 }

c2900InfoTotalBufferDepth OBJECT-TYPE
```

```
    SYNTAX Integer32
--  Units
--    buffers
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "It represents the total number of 64-byte buffers
         in the Ethernet Controller."
    ::= { c2900SysInfo 3 }

c2900InfoAddrCapacity OBJECT-TYPE
    SYNTAX Integer32
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The system-wide maximum number of MAC addresses
         supported in the address table, a primary
         resource when forwarding frames through a bridge.
         The address table is dynamically updated with new
         learned addresses inserted and aged addresses removed.
         The address capacity represented by this object includes
         dynamic, secure, and static address types.

         To ensure optimal performance, the number of MAC addresses
         in the bridged local area network to which this bridge is
         connected should be less than the value of this object.

         The system administrator can refer to this object
         for the number of MAC addresses suppored by this box."
    ::= { c2900SysInfo 4 }

c2900InfoRestrictedStaticAddrCapacity OBJECT-TYPE
    SYNTAX Integer32
```

```
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The system-wide maximum number of static addresses
         supported.  A static address is one that has explicit
         source port filtering information assigned.

         This number limits the static table's entries configured by user."
    REFERENCE
        "IEEE 802.1D-1990: Section 6.7.2"
    ::= { c2900SysInfo 5 }

c2900InfoSelfTestFailed OBJECT-TYPE
    SYNTAX OCTET STRING(SIZE(8))
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A bit array where the presence of a particular bit
         indicates a failure of a specific Power On Self Test."
    ::= { c2900SysInfo 6 }

c2900InfoUtilDisplay OBJECT-TYPE
    SYNTAX Gauge32
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of utilization meter LEDs currently
         lit on the front panel, if the value of
         c2900InfoVisualIndicatorMode is selected as
         utilization(4).  More LEDs are lit
         as more total bandwidth through the switch
         is being utilized.
```

```
        The percentage of the utilization is calculated as follows:
          the number of LEDs lit/the total number of LEDs.

        The total number of the LEDs is twenty four for c2900 switch.

        This object is meaningful when the value of c2900InfoVisualIndicatorMode
        is utilization(4). If the value of c2900InfoVisualIndicatorMode is
        not utilization(4), the value of the object will be zero."
    ::= { c2900SysInfo 7 }

c2900InfoVisualIndicatorMode OBJECT-TYPE
    SYNTAX INTEGER {
        portStatus(1),
        fullDuplex(2),
        linkRate(3),
        utilization(4)
        }
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "This object reflects what is currently selected
         as the visual indication mode.

        The portStatus(1) mode uses the visual LEDS
        to indicate port link status.

        The fullDuplex(2) mode uses the visual LEDS
        to indicate that a port is running with full
        duplex or half duplex or no link status.

        The linkRate(3) mode uses the visual LEDS to
        indicate the rate of operation on a port:
        100 MBPS or 10 MBPS or no link.
```

```
        The utilization(4) mode uses the visual LEDS to
        indicate the utiliztion of the system as more total
        bandwidth through the switch is being utilized,
        more LED's are lit."
    ::= { c2900SysInfo 8 }

c2900InfoRedunantPowerSupplyInfo OBJECT-TYPE
    SYNTAX INTEGER {
        absent(1),
        connectedFunctional(2),
        connectedNotFunctional(3),
        functionalPrimaryFailed(4)
        }
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The switch allows a redundant power supply in addition
        to its local power supply. Only one power source can be
        supplying power to a unit.

        absent(1) :the redundant power supply is not connected
                to the switch.

        connectedFunctional(2) : the redundant power supply is
                connected to the switch and operational.

        connectedNotFunctional(3): the redundant power supply
                is connected to the switch, but cannot supply
                power to the system.

        functionalPrimaryFailed(4): the redundant power supply
                is installed, powered on, and operational,
```

```
                but a failure exists in the local power
                supply system."
    ::= { c2900SysInfo 9 }

c2900InfoBoardIdentifier OBJECT-TYPE
    SYNTAX Integer32
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "Returns the identifier of the main board
         on which the FastSwitch firmware resides."
    ::= { c2900SysInfo 10 }

c2900ConfigAddressViolationAction OBJECT-TYPE
    SYNTAX INTEGER {
        doNothing(1),
        disablePort(2),
        sendNotify(3),
        disablePortAndNotify(4)
        }
    ACCESS read-write
    STATUS deprecated
    DESCRIPTION
        "Indicates what action to take when an address violation
         (an address mismatch or duplication) occurs on a
         secure port.  The default action is to do nothing.

          doNothing(1)       : do nothing
          disablePort(2)     : disable port; the port can only be reenabled
                               by an explicit management action.
          sendNotify(3)      : generate address violation notification.
          disablePortAndNotify(4): disable port and send notification.
```

```
        Default value: doNothing(1).

        This object is deprecated. A separate object
        c2900PortAddressViolationAction is defined for each port."
    ::= { c2900SysConfig 1 }

c2900ConfigBroadcastStormAlarm OBJECT-TYPE
    SYNTAX TruthValue
--    Rsyntax INTEGER {
--        true(1),
--        false(2)
--        }
    ACCESS read-write
    STATUS deprecated
    DESCRIPTION
        "When set to true(1), the switch will generate
        a broadcastStorm notification upon detecting a port is
        receiving broadcast packets at a rate higher than
        or equal to the specified broadcast threshold.
        When set to false(2), no such trap will be issued.
        Default value: false(2).

        This object is deprecated. A separate object
        c2900PortBroadcastStormAlarm is defined for each port."
    ::= { c2900SysConfig 2 }

c2900ModuleTable OBJECT-TYPE
    SYNTAX SEQUENCE OF C2900ModuleEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "A list of module entries."
    ::= { c2900MIBObjects 3 }
```

```
c2900ModuleEntry OBJECT-TYPE
    SYNTAX C2900ModuleEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "Entry containing status information about one module in
         the c2900 chassis."
    INDEX { c2900ModuleIndex }
    ::= { c2900ModuleTable 1 }

C2900ModuleEntry ::= SEQUENCE {
    c2900ModuleIndex Integer32,
    c2900ModuleStatus INTEGER,
    c2900ModuleType INTEGER,
    c2900ModuleHwVersion DisplayString,
    c2900ModuleSwVersion DisplayString
    }

c2900ModuleIndex OBJECT-TYPE
    SYNTAX Integer32(1..64)
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "Module index into c2900ModuleTable ."
    ::= { c2900ModuleEntry 1 }

c2900ModuleStatus OBJECT-TYPE
    SYNTAX INTEGER {
        moduleNotInstalled(1),
        moduleInTest(2),
        moduleHealthy(3),
        moduleFaulty(4)
```

```
        }
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The overall status of of the module."
    ::= { c2900ModuleEntry 2 }

c2900ModuleType OBJECT-TYPE
    SYNTAX INTEGER {
        other(1),
        empty(2),
        wsx2914xl(3),
        wsx2922xl(4)
        }
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The type of module installed in malibu switch."
    ::= { c2900ModuleEntry 3 }

c2900ModuleHwVersion OBJECT-TYPE
    SYNTAX DisplayString(SIZE(0..12))
--    Rsyntax OCTET STRING(SIZE(0..12))
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The hardware version of the module. The format
         of the version string x.y.z where x,y, and z
         are hardware register field values."
    ::= { c2900ModuleEntry 4 }

c2900ModuleSwVersion OBJECT-TYPE
    SYNTAX DisplayString(SIZE(0..40))
```

```
--    Rsyntax OCTET STRING(SIZE(0..40))
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The software version of the module."
    ::= { c2900ModuleEntry 5 }

c2900PortTable OBJECT-TYPE
    SYNTAX SEQUENCE OF C2900PortEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "A list of port entries.  The number of entries is
         determined by the number of modules in the chassis
         and the number of ports on each module."
    ::= { c2900Port 1 }

c2900PortEntry OBJECT-TYPE
    SYNTAX C2900PortEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "Entry containing information for a particular
         switched port on a module installed. The entries
         are not created or deleted by management commands.
         In other words, it reflects the installed hardware and
         cannot change while running."
    INDEX { c2900PortModuleIndex, c2900PortIndex }
    ::= { c2900PortTable 1 }

C2900PortEntry ::= SEQUENCE {
    c2900PortModuleIndex Integer32,
    c2900PortIndex Integer32,
```

c2900PortUsageApplication INTEGER,
c2900PortGroupIndex Integer32,
c2900PortMayLearnAddress TruthValue,
c2900PortMayForwardFrames TruthValue,
c2900PortBufferCongestionControl TruthValue,
c2900PortBufferCongestionThreshholdPercent Integer32,
c2900PortFrameAge Integer32,
c2900PortAddrSecureMaxAddresses Integer32,
c2900PortAddrSecureCurrentAddresses Integer32,
c2900PortAddrSecureAddrViolations Counter32,
c2900PortNumberOfLearnedAddresses Gauge32,
c2900PortNumberOfDroppedAddresses Counter32,
c2900PortClearAddresses TruthValue,
c2900PortFloodUnknownMulticasts TruthValue,
c2900PortFloodUnknownUnicasts TruthValue,
c2900PortLinkbeatStatus INTEGER,
c2900PortBroadcastStormAction INTEGER,
c2900PortBroadcastRisingThreshold Integer32,
c2900PortBroadcastFallingThreshold Integer32,
c2900PortStatus INTEGER,
c2900PortTestResult TruthValue,
c2900PortVisualIndicator INTEGER,
c2900PortIfIndex InterfaceIndex,
c2900PortAddressViolationAction INTEGER,
c2900PortBroadcastStormAlarm TruthValue,
c2900PortMonitorDestinationPort Integer32,
c2900PortSwitchPortIndex Integer32,
c2900PortMonitoredPortMap OCTET STRING,
c2900PortDuplexState INTEGER,
c2900PortDuplexStatus INTEGER,
c2900PortAdminSpeed INTEGER,
c2900PortNoMonitorDestinationPort Integer32
}

```
c2900PortModuleIndex OBJECT-TYPE
    SYNTAX Integer32(0..64)
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "An index value that uniquely identifies the module
         where this port is located. The value is determined
         by the chassis slot number into which the module is
         plugged."
    ::= { c2900PortEntry 1 }

c2900PortIndex OBJECT-TYPE
    SYNTAX Integer32(1..64)
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "An index value that uniquely identifies this port
         within a module."
    ::= { c2900PortEntry 2 }

c2900PortUsageApplication OBJECT-TYPE
    SYNTAX INTEGER {
        standard(1),
        security(2),
        monitor(3),
        portGrouping(4),
        network(5),
        networkGroup(6)
        }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
```

"This object indicates how the port is to be used.
The variable usage applications are shown above.
These applications are defined such that they
are mutually exclusive. In other words, a port
using the security(2) application cannot also
use the monitor(3) application at the same time.

The default usage is standard(1), or no special
behavior (the port behaves as normal
switched port).

The security(2) usage adds addressing security
to the port, whereby all learned addresses are
permanently kept (secure address).

The monitor(3) usage provides network diagnosis
by reflecting traffic on other ports to this port.

The portGrouping(4) application treats this and other
ports in the same group as one(inter-switch)
connection for more bandwidth potential. This
value truly takes effect only when the corresponding
c2900PortGroupIndex is set to a valid portGroup
index value. Otherwise, the port will have standard
usage in all appearances.

The network(5) usage saves address table space when the port
is used as the link to a large network with many MAC addresses
by disabling address learning on the port and allowing
unknown unicasts packets received on other ports of the vlan
the port associated with to be forwarded only to the port.

The networkGroup(6) usage treates this and other

```
        ports in the same group as a network port group. All ports in
        port are network ports in the sense that address learning is
        disabled on them and unknown unicast packets received on other
        ports of the vlan are forwarded to the group."
    DEFVAL { standard }
    ::= { c2900PortEntry 3 }

c2900PortGroupIndex OBJECT-TYPE
    SYNTAX Integer32
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "This object is meaningful only when the
        corresponding c2900PortUsageApplication is portGrouping.
        Ports assigned with the same value of c2900PortGroupIndex
        belong to the same connection channel."
    ::= { c2900PortEntry 4 }

c2900PortMayLearnAddress OBJECT-TYPE
    SYNTAX TruthValue
--    Rsyntax INTEGER {
--        true(1),
--        false(2)
--        }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "This object reflects an internal state of the port
        with regard to its ability to learn new addresses.
        Certain port configurations such as learning time
        limit, security usage, etc., and some Spanning
        Tree Protocol states can temporarily prohibit the
        port from learning.  This object is true(1) if the
```

```
        port is allowed to learn. It is false(2) otherwise.

        Setting this object to true(1) will fail,
        if the port is not in normal state."
    DEFVAL { true }
    ::= { c2900PortEntry 5 }

c2900PortMayForwardFrames OBJECT-TYPE
    SYNTAX TruthValue
--    Rsyntax INTEGER {
--        true(1),
--        false(2)
--        }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "This object reflects an internal state of the port
        with regard to its ability to forward frames.
        A port sometimes stops forwarding frames when it
        is blocked by the Spanning Tree Protocol, or
        while it is undergoing temporary load balancing
        as part of the port grouping usage.
        This object is true(1) if the port is allowed to
        forward frames. It is false(2) otherwise.

        Setting this object to true(1) will fail,
        if the port is in the middle of being blocked by the Spanning Tree Protocol,
        or while it is undergoing temporary load balancing
        as part of the port grouping usage.

        This object is only supported for static access ports."
    DEFVAL { true }
    ::= { c2900PortEntry 6 }
```

```
c2900PortBufferCongestionControl OBJECT-TYPE
    SYNTAX TruthValue
--    Rsyntax INTEGER {
--        true(1),
--        false(2)
--        }
    ACCESS read-only
    STATUS deprecated
    DESCRIPTION
        "Seting this object to true(1) allows the switch
        to run its buffer congestion control algorithm on the
        port.  Setting the object to false(2) disallows
        such control. The buffer congestion control algorithm
        is summarized in the description of
        the c2900PortBufferCongestionThresholdPercent below.

        This object is deprecated"
    ::= { c2900PortEntry 7 }

c2900PortBufferCongestionThreshholdPercent OBJECT-TYPE
    SYNTAX Integer32(1..99)
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The port buffer congestion threshold provides an early
        warning to the switch that the port is about to
        exhaust all its guaranteed buffers, leading to congestion.
        This threshold is expressed as a percentage of the port's
        total guaranteed buffer depth.  Once this threshold has
        been crossed, the switch begins colliding with frames
        received on the port for a fixed period of time.
        After this time has expired, the switch determines
```

```
        whether the port's buffer congestion has been alleviated.
        If this situation has not changed, the switch resumes
        this congestion control algorithm on the port."
    DEFVAL { 80 }
    ::= { c2900PortEntry 8 }

c2900PortFrameAge OBJECT-TYPE
    SYNTAX Integer32(50..4000)
--  Units
--    milliseconds
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The aging interval in milliseconds after
        which old frames queued for transmission on this
        port are discarded."
    REFERENCE
        "Section 4.2 IEEE802.1D-1993"
    DEFVAL { 1000 }
    ::= { c2900PortEntry 9 }

c2900PortAddrSecureMaxAddresses OBJECT-TYPE
    SYNTAX Integer32
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The maximum number of secure addresses that can
        be learned on this port when it is a secure port.
        This number should be always higer than
        c2900PortAddrSecureCurrentAddresses"
    ::= { c2900PortEntry 10 }

c2900PortAddrSecureCurrentAddresses OBJECT-TYPE
```

```
    SYNTAX Integer32
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The current number of statically assigned
         unicast addresses on the port."
    ::= { c2900PortEntry 11 }

c2900PortAddrSecureAddrViolations OBJECT-TYPE
    SYNTAX Counter32
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of times a source address was
         seen on this port which duplicates a secured
         address configured on another port,
         plus the number times a source address was
         seen on this port which does not match any
         addresses secured for the port."
    ::= { c2900PortEntry 12 }

c2900PortNumberOfLearnedAddresses OBJECT-TYPE
    SYNTAX Gauge32
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The current number of dynamically learned
         addresses on the port."
    ::= { c2900PortEntry 13 }

c2900PortNumberOfDroppedAddresses OBJECT-TYPE
    SYNTAX Counter32
    ACCESS read-only
```

```
    STATUS mandatory
    DESCRIPTION
        "The number of addresses that could not be learned
         or assigned for the port because its address table
         was full at one time or another."
    ::= { c2900PortEntry 14 }

c2900PortClearAddresses OBJECT-TYPE
    SYNTAX TruthValue
--    Rsyntax INTEGER {
--        true(1),
--        false(2)
--        }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "Set to true(1) to delete all learned and assigned
         static unicast addresses the port currently has.
         Setting the object to false(2) has no effect.
         This object always returns false(2) when read."
    ::= { c2900PortEntry 15 }

c2900PortFloodUnknownMulticasts OBJECT-TYPE
    SYNTAX TruthValue
--    Rsyntax INTEGER {
--        true(1),
--        false(2)
--        }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "Set to true(1) to allow forwarding to this
         port frames addressed to multicast addresses
```

```
        that have not been configured for the port.
        Set to false(2) to filter and discard such frames."
    DEFVAL { true }
    ::= { c2900PortEntry 16 }

c2900PortFloodUnknownUnicasts OBJECT-TYPE
    SYNTAX TruthValue
--    Rsyntax INTEGER {
--        true(1),
--        false(2)
--        }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "This object controls the forwarding of unknown
        unicast frames to this port. When set to true(1),
        the switch will, upon receiving a frame with an
        unknown unicast destination address from
        another port, transmit the frame to this port.
        When set to false(2), switch will filter
        and not transmit said frames to this port.
        Default value: false(2) for ports  using
        the security or monitor application;
        true(1) for all other usage applications."
    ::= { c2900PortEntry 17 }

c2900PortLinkbeatStatus OBJECT-TYPE
    SYNTAX INTEGER {
        unknown(1),
        linkbeat(2),
        nolinkbeat(3)
        }
    ACCESS read-only
```

```
    STATUS mandatory
    DESCRIPTION
        "This object depends on the physical layer in use
         and indicates the current port linkbeat status:
         if the physical link between two devices is
         properly connected or not.
         If the value is linkbeat(1), there is linkbeat.
         If the value is nolinkbeat(2), there is no linkbeat.
         If the value is unknown(3), the information is not
         available."
    ::= { c2900PortEntry 18 }

c2900PortBroadcastStormAction OBJECT-TYPE
    SYNTAX INTEGER {
        stopBroadcastForwarding(1),
        forwardBroadcast(2),
        disablePort(3)
        }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "Indicates what action to take when the broadcast
         rising threshold for a port is reached.
         The default action is to forwardBroadcast(2) as usual
         for all broadcast frames received from the port. The
         other action is to stopBroadcastForwarding(1) frames
         until the broadcast reception rate falls to
         or below the falling threshold.
         The action disablePort(3) is not allowed."
    DEFVAL { forwardBroadcast }
    ::= { c2900PortEntry 19 }

c2900PortBroadcastRisingThreshold OBJECT-TYPE
```

```
    SYNTAX Integer32
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The broadcast rising threshold is measured in the number
         of broadcast frames received on a port in a second.

         When the number of broadcast frames received per
         second on this port crosses this threshold,
         the appropriate action as specified by the object
         c2900PortBroadcastStormAction will take place.
         See the description of c2900PortBroadcastStormAction."
    ::= { c2900PortEntry 20 }

c2900PortBroadcastFallingThreshold OBJECT-TYPE
    SYNTAX Integer32
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The broadcast falling threshold is measured in number of
         broadcast frames received on a port in a second.

         When the falling threshold is crossed and the
         c2900PortBroadcastStormAction was stopBroadcastForwarding(1)
         then broadcast forwarding will be re-enabled on the port.
         For the other values of c2900PortBroadcastStormAction,
         crossing the falling threshold has no affect."
    ::= { c2900PortEntry 21 }

c2900PortStatus OBJECT-TYPE
    SYNTAX INTEGER {
        other(1),
        disabled(2),
```

blocking(3),
listening(4),
learning(5),
preforwarding(6),
forwarding(7),
secureforwarding(8),
suspended(9),
broken(10)
}
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The port's current state as defined by application
of the Spanning Tree Protocol. This state controls
what action a port takes on reception of a frame.
If the switch has detected a port that is
malfunctioning it will place that port into the
broken(10) state. For ports which are
disabled(see dot1dStpPortEnable), this object will
have a value of disabled(2).
Since the switch implements three
additional states which are not part of IEEE Standard,
these additional states are possible values for
c2900PortStaus. The additional states are preforwarding(6),
secureforwarding(8), and suspended(9). Note: except
for these additional states, this object is the same
as the dot1dStpPortState object.
Default value: blocking(2).

This object is only supported for static access ports."
REFERENCE
"Section 4.5.5.2 IEEE802.1D-1990"
DEFVAL { blocking }

```
    ::= { c2900PortEntry 22 }

c2900PortTestResult OBJECT-TYPE
    SYNTAX TruthValue
--    Rsyntax INTEGER {
--        true(1),
--        false(2)
--        }
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "This object indicates if the port passed power on self test or not.
         If the value of this object is true(1), the port passed test.
         If the value of this object is false(2), the port failed test."
    ::= { c2900PortEntry 23 }

c2900PortVisualIndicator OBJECT-TYPE
    SYNTAX INTEGER {
        notused(1),
        black(2),
        amber(3),
        green(4)
        }
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "This object is used to indicate the current color of
         a LED. If a LED is flashing, the value of this object will
         represent the color of the LED at that instant in time."
    ::= { c2900PortEntry 24 }

c2900PortIfIndex OBJECT-TYPE
    SYNTAX InterfaceIndex(1..2147483647)
```

```
--    Rsyntax Integer32(1..2147483647)
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The value of the instance of the ifIndex object,
         defined in MIB-II, for the interface corresponding
         to this port."
    ::= { c2900PortEntry 25 }

c2900PortAddressViolationAction OBJECT-TYPE
    SYNTAX INTEGER {
        doNothing(1),
        disablePort(2),
        sendNotify(3),
        disablePortAndNotify(4)
        }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "Indicates what action to take when an address violation
         (an address mismatch or duplication) occurs on a
         secure port.  The default action is to do nothing.

          doNothing(1)       : do nothing
          disablePort(2)     : disable port; the port can only be reenabled
                               by an explicit management action.
          sendNotify(3)        : generate address violation notification.
          disablePortAndNotify(4): disable port and send notification.

          Default value: doNothing(1)."
    DEFVAL { doNothing }
    ::= { c2900PortEntry 26 }
```

```
c2900PortBroadcastStormAlarm OBJECT-TYPE
    SYNTAX TruthValue
--    Rsyntax INTEGER {
--        true(1),
--        false(2)
--        }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "When set to true(1), the switch will generate
        a broadcastStorm notification upon detecting a port is
        receiving broadcast packets at a rate higher than
        or equal to the specified broadcast threshold.
        When set to false(2), no such trap will be issued.
        Default value: false(2).
        c2900PortBroadcastStormAlarm is defined for each port."
    DEFVAL { false }
    ::= { c2900PortEntry 27 }

c2900PortMonitorDestinationPort OBJECT-TYPE
    SYNTAX Integer32(0..64)
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "Switch Port (c2900PortSwitchPortIndex) index value of the port
        that is to be monitored by this port. A value of zero can't be
        used to do set, it is used to return when this object is read.

        To remove a port from the monitored list use the object
        c2900PortNoMonitorDestinationPort."
    ::= { c2900PortEntry 28 }

c2900PortSwitchPortIndex OBJECT-TYPE
```

```
    SYNTAX Integer32(1..64)
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "Switch port index of a port is a value that uniquely
         identifies the port within a switch. This is obtained
         from the port index (c2900PortIndex) and the module
         index (c2900PortModuleIndex)."
    ::= { c2900PortEntry 29 }

c2900PortMonitoredPortMap OBJECT-TYPE
    SYNTAX OCTET STRING(SIZE(0..32))
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "Indicates which ports are actually being monitored.
         The octet string contains one bit per port. Each bit
         within the octet string represents one port of the device.
         The ordering of ports represented within the octet string
         is in the same order as in the RFC 1493 dot1dStpPortTable.

         The bit value interpretation is as follows:
                        1 = being monitored
                        0 = not being monitored"
    ::= { c2900PortEntry 30 }

c2900PortDuplexState OBJECT-TYPE
    SYNTAX INTEGER {
        fullduplex(1),
        halfduplex(2),
        autoNegotiate(3)
        }
    ACCESS read-write
```

```
    STATUS mandatory
    DESCRIPTION
        "Set to fullduplex(1) to operate in full duplex mode, port
         will allow simultaneous transmit and receive which can
         double its bandwidth.

         Set to halfduplex(2) to operate in half duplex mode.

         Set to autoNegotiate(3) to allow the switch to negoti-
         ate with the other end of the connection.

         The status of duplex mode on a port is available with
         c2900PortDuplexStatus object."
    DEFVAL { autoNegotiate }
    ::= { c2900PortEntry 31 }

c2900PortDuplexStatus OBJECT-TYPE
    SYNTAX INTEGER {
        fullduplex(1),
        halfduplex(2)
        }
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The status of duplex mode on this port. This shows
         the result of full duplex auto-negotiation when
         c2900PortDuplexState is set to auto-negotiate."
    ::= { c2900PortEntry 32 }

c2900PortAdminSpeed OBJECT-TYPE
    SYNTAX INTEGER {
        autoDetect(1),
        s10000000(10000000),
```

```
        s100000000(100000000)
        }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The object controls the speed of the port.
         The current operational speed of the port can be
         determined from ifSpeed."
    DEFVAL { autoDetect }
    ::= { c2900PortEntry 33 }

c2900PortNoMonitorDestinationPort OBJECT-TYPE
    SYNTAX Integer32(0..64)
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "Switch Port (c2900PortSwitchPortIndex) index value of the port
         that is to be removed from the monitored list. A value of zero
         can't be used to do set, it is used to return when this object
         is read.

         To add a port to the monitored list use the object
         c2900PortMonitorDestinationPort."
    ::= { c2900PortEntry 34 }

c2900PortStatsTable OBJECT-TYPE
    SYNTAX SEQUENCE OF C2900PortStatsEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "A list of port entries.  The number of entries is
         determined by the number of modules in the chassis
         and the number of ports on each module."
```

```
    ::= { c2900Port 2 }

c2900PortStatsEntry OBJECT-TYPE
    SYNTAX C2900PortStatsEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "Entry containing information for a particular
         switched port on a module installed. The entry
         cannot be created or deleted."
    INDEX { c2900PortModuleIndex, c2900PortIndex }
    ::= { c2900PortStatsTable 1 }

C2900PortStatsEntry ::= SEQUENCE {
    c2900PortRxNoBwFrames Counter32,
    c2900PortRxNoBufferFrames Counter32,
    c2900PortRxNoDestUniFrames Counter32,
    c2900PortRxNoDestMultiFrames Counter32,
    c2900PortRxSuppressBcastFrames Counter32,
    c2900PortRxFcsErrFrames Counter32,
    c2900PortCollFragFrames Counter32,
    c2900PortTxMulticastFrames Counter32,
    c2900PortTxBroadcastFrames Counter32
    }

c2900PortRxNoBwFrames OBJECT-TYPE
    SYNTAX Counter32
--  Units
--    frames
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A count of frames received on this port that were discarded
```

```
        due to a lack of bandwidth resources in the Catalyst Switch
        forwarding engine."
    ::= { c2900PortStatsEntry 1 }

c2900PortRxNoBufferFrames OBJECT-TYPE
    SYNTAX Counter32
--  Units
--    frames
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A count of frames received that were discarded due
        to a lack of frame buffer resources in the Catalyst Switch
        forwarding engine."
    ::= { c2900PortStatsEntry 2 }

c2900PortRxNoDestUniFrames OBJECT-TYPE
    SYNTAX Counter32
--  Units
--    frames
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A count of unicast frames received that were discarded,
        because the forwarding rules stipulate that they are
        not be forwarded."
    ::= { c2900PortStatsEntry 3 }

c2900PortRxNoDestMultiFrames OBJECT-TYPE
    SYNTAX Counter32
--  Units
--    frames
    ACCESS read-only
```

```
    STATUS mandatory
    DESCRIPTION
        "A count of multicast frames received that were discarded,
         because they have not been configured for the port."
    ::= { c2900PortStatsEntry 4 }

c2900PortRxSuppressBcastFrames OBJECT-TYPE
    SYNTAX Counter32
--  Units
--    frames
    ACCESS read-only
    STATUS deprecated
    DESCRIPTION
        "A count of broadcast frames received that were discarded
         because of the threshold-based broadcast suppression.

         This object is deprecated, because there is no way that the
         Malibu system to give the brodcast_supress frames"
    ::= { c2900PortStatsEntry 5 }

c2900PortRxFcsErrFrames OBJECT-TYPE
    SYNTAX Counter32
--  Units
--    frames
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The total number of frames received with FCS errors.
         This total includes all frames received with an FCS
         error and an integral number of bytes.
         Unlike RFC1650's dot3StatsFCSErrors,
         this object does not include frames which are less
         than the minimum packet size (such as collision fragments)."
```

```
    ::= { c2900PortStatsEntry 6 }

c2900PortCollFragFrames OBJECT-TYPE
    SYNTAX Counter32
--  Units
--    frames
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The total number of frames whose lengths are less than
         64 and have bad FCS values.
         The preamble and sfd fields are excluded from the byte
         count of a frame while the FCS field is included."
    ::= { c2900PortStatsEntry 7 }

c2900PortTxMulticastFrames OBJECT-TYPE
    SYNTAX Counter32
--  Units
--    frames
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A count of frames that are successfully transmitted and
         are directed to a multicast address.
         Unlike RFC1573's ifOutMulticastPkts, this object does not
         include those that were discarded or not sent."
    ::= { c2900PortStatsEntry 8 }

c2900PortTxBroadcastFrames OBJECT-TYPE
    SYNTAX Counter32
--  Units
--    frames
    ACCESS read-only
```

```
    STATUS mandatory
    DESCRIPTION
        "A count of frames that are successfully transmitted and
         are directed to the broadcast address.
         Unlike RFC1573's ifOutBroadcastPkts, this object does
         not include those that were discarded or not sent."
    ::= { c2900PortStatsEntry 9 }

c2900BandwidthUsageCurrent OBJECT-TYPE
    SYNTAX Gauge32
--  Units
--    megabits per second
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The current bandwidth consumed.  The measurement unit is
         in megabits per second (1,000,000 bits/second).
         This value gives a reasonable estimate of the amount of
         traffic currently flowing through the switch.

         It is calculated as follows:

         Octets*8 + Frames*(96 + 64)
         ---------------------------
         Measurement Interval * 1,000,000 * 2

        Where:
          Measurement Interval is the amount of time over which
          the Octets and Frames were collected, in seconds.

          Measurement Interval is always one second in current
          implementation.
```

Octets is the total number of octets transmitted
or received by all network interfaces, excluding
framing data but including FCS. This includes octets
in frames which were partially transmitted or
received (due to collisions, for example).

Frames is the total number of frames transmitted
or received by all network interfaces, including
frames with errors.

The number of frames is multiplied by 96 plus 64 in
order to estimate the delay between each frame for
Ethernet's IPG and preamble/SFD.

The '2' in the divisor makes this a forwarding bandwidth
counter. A frame received on one interface is typically
forwarded out another interface. In order to avoid double-
counting this frame's bandwidth, once on the receiving
interface and once on the transmitting interface, the
total bandwidth is divided by two.
Since multicast and broadcast frames can be sent to multiple
ports, the above is at best a lower bound."
::= { c2900BandwidthUsage 1 }

c2900BandwidthUsageMaxPeakEntries OBJECT-TYPE
SYNTAX Integer32
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The maximum number of entries c2900BandwidthUsagePeakTable
can have."
::= { c2900BandwidthUsage 2 }

```
c2900BandwidthUsagePeakInterval OBJECT-TYPE
    SYNTAX INTEGER {
        onehour(1),
        threehours(3),
        sixhours(6),
        twelvehours(12),
        oneday(24),
        twodays(48),
        threedays(72),
        fourdays(96),
        fivedays(120),
        sixdays(144),
        oneweek(168)
        }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "This object specifies the length of time
         which forms a peak bandwidth measurement interval.

         A write to this object with any new value
         restarts the peak bandwidth recording interval used
         by bandwidthUsagePeakTable.

         In other words, the bandwidthUsagePeakTable
         will be cleared and entry number one will record
         the peak with a new measurement interval."
    DEFVAL { oneday }
    ::= { c2900BandwidthUsage 3 }

c2900BandwidthUsagePeakRestart OBJECT-TYPE
    SYNTAX TruthValue
--    Rsyntax INTEGER {
```

```
--        true(1),
--        false(2)
--        }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "Set to true(1) to clear the c2900BandwidthUsagePeakTable
         and restart the peak bandwidth recording.
         No action will be taken if this object is set
         to false(2).  This object returns false(2)
         when read."
    ::= { c2900BandwidthUsage 4 }

c2900BandwidthUsageCurrentPeakEntry OBJECT-TYPE
    SYNTAX Integer32
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A value identifying an instance of the
         c2900BandwidthUsagePeakIndex where the
         peak bandwidth estimation is most recent."
    ::= { c2900BandwidthUsage 5 }

c2900BandwidthUsagePeakTable OBJECT-TYPE
    SYNTAX SEQUENCE OF C2900BandwidthUsagePeakEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "A list of entries containing peak bandwidth usages in
         a number of recording interval.

         After being cleared, entries are added to the bandwidthUsagePeakTable
         with ascending values of c2900BandwidthUsagePeakIndex starting at 1.
```

```
        When the number of entries reaches c2900BandwidthUsageMaxPeakEntries,
        each new recording interval is assigned the value of
        c2900BandwidthUsagePeakIndex corresponding to the oldest entry,
        overwriting the previous contents of that entry."
    ::= { c2900BandwidthUsage 6 }

c2900BandwidthUsagePeakEntry OBJECT-TYPE
    SYNTAX C2900BandwidthUsagePeakEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "Information about peak bandwidth usage in a recording interval."
    INDEX { c2900BandwidthUsagePeakIndex }
    ::= { c2900BandwidthUsagePeakTable 1 }

C2900BandwidthUsagePeakEntry ::= SEQUENCE {
    c2900BandwidthUsagePeakIndex Integer32,
    c2900BandwidthUsageStartTime DateAndTime,
    c2900BandwidthUsagePeak Gauge32,
    c2900BandwidthUsagePeakTime DateAndTime
    }

c2900BandwidthUsagePeakIndex OBJECT-TYPE
    SYNTAX Integer32(1..2147483647)
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "Number from one to c2900BandwidthUsageMaxPeakEntries
        identifying a particular c2900BandwithUsagePeakEntry."
    ::= { c2900BandwidthUsagePeakEntry 1 }

c2900BandwidthUsageStartTime OBJECT-TYPE
    SYNTAX DateAndTime
```

```
--    Rsyntax OCTET STRING(SIZE(8 | 11))
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The time that marks the start of this recording interval."
    ::= { c2900BandwidthUsagePeakEntry 2 }

c2900BandwidthUsagePeak OBJECT-TYPE
    SYNTAX Gauge32
--  Units
--    megabits per second
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The maximum bandwidth usage of any measurement interval
         within this recording interval.

         This value is an estimate of the highest amount of
         traffic flowing through the switch during this
         recording interval.

        It is calculated as follows:

        Octets*8 + Frames*(96 + 64)
        --------------------------
        Measurement Interval * 1,000,000 * 2

        Where:
          Measurement Interval is the amount of time over which
          the Octets and Frames were collected, in seconds.

          Measurement Interval is always one second in c2900's
          implementation.
```

```
            Octets is the total number of octets transmitted
            or received by all network interfaces, excluding
            framing data but including FCS. This includes octets
            in frames which were partially transmitted or
            received (due to collisions, for example).

            Frames is the total number of frames transmitted
            or received by all network interfaces, including
            frames with errors.

           The number of frames is multiplied by 96 plus 64 in
           order to estimate the delay between each frame for
           Ethernet's IPG and preamble/SFD.

           The '2' in the divisor makes this a forwarding bandwidth
           counter. A frame received on one interface is typically
           forwarded out another interface. In order to avoid double-
           counting this frame's bandwidth, once on the receiving
           interface and once on the transmitting interface, the
           total bandwidth is divided by two.
           Since multicast and broadcast frames can be sent to multiple
           ports, the above is at best a lower bound."
       ::= { c2900BandwidthUsagePeakEntry 3 }

c2900BandwidthUsagePeakTime OBJECT-TYPE
       SYNTAX DateAndTime
--     Rsyntax OCTET STRING(SIZE(8 | 11))
       ACCESS read-only
       STATUS mandatory
       DESCRIPTION
           "The start time of the measurement interval."
       ::= { c2900BandwidthUsagePeakEntry 4 }
```

```
c2900AddressViolation TRAP-TYPE
-- Reverse mappable trap
   ENTERPRISE c2900MibNotifications
   VARIABLES {
      c2900PortIfIndex }
--  Status
--    mandatory
   DESCRIPTION
      "The addressViolation notification is generated
       when an address violation is detected
       on a secured port. The generation of the
       addressViolation notification can be enabled
       or suppressed using the object
       c2900ConfigAddressViolationAction.
       The particular secured port is indicated by
       the value of c2900PortIfIndex."
   ::= 1

c2900BroadcastStorm TRAP-TYPE
-- Reverse mappable trap
   ENTERPRISE c2900MibNotifications
   VARIABLES {
      c2900PortBroadcastRisingThreshold }
--  Status
--    mandatory
   DESCRIPTION
      "The broadcastStorm notification is generated
       upon detecting a port is receiving
       broadcast packets at a rate crossing
       the specified broadcast threshold.
       This trap is only for the rising threshold.
       The particular port is indicated by the values of
```

```
        c2900PortModuleIndex and c2900PortIndex, and the
        value of the threshold is given by
        c2900PortBroadcastRisingThreshold."
    ::= 2

c2900SysInfoGroup OBJECT IDENTIFIER ::= { c2900MIBGroups 1 }
-- OBJECT-GROUP
--  Status
--    mandatory
--  Descr
--    The collection of objects which are used to provide
--     the general switch information.
--  objects
--    c2900InfoBoardRevision, c2900InfoPeakBuffersUsed,
--    c2900InfoTotalBufferDepth, c2900InfoAddrCapacity,
--    c2900InfoRestrictedStaticAddrCapacity,
--    c2900InfoSelfTestFailed, c2900InfoUtilDisplay,
--    c2900InfoVisualIndicatorMode,
--    c2900InfoRedunantPowerSupplyInfo, c2900InfoBoardIdentifier

c2900SysConfigGroup OBJECT IDENTIFIER ::= { c2900MIBGroups 2 }
-- OBJECT-GROUP
--  Status
--    deprecated
--  Descr
--    The collection of objects which are used to configure
--     the switch.
--  objects
--    c2900ConfigAddressViolationAction,
--    c2900ConfigBroadcastStormAlarm

c2900ModuleGroup OBJECT IDENTIFIER ::= { c2900MIBGroups 3 }
-- OBJECT-GROUP
```

```
--  Status
--    mandatory
--  Descr
--    The object is used to provide the module status.
--  objects
--    c2900ModuleStatus, c2900ModuleType, c2900ModuleHwVersion,
--    c2900ModuleSwVersion

c2900PortGroup OBJECT IDENTIFIER ::= { c2900MIBGroups 4 }
-- OBJECT-GROUP
--  Status
--    deprecated
--  Descr
--    The collection of objects which are used to
--     provide port status and configuration.
--  objects
--    c2900PortUsageApplication, c2900PortGroupIndex,
--    c2900PortMayLearnAddress, c2900PortMayForwardFrames,
--    c2900PortBufferCongestionControl,
--    c2900PortBufferCongestionThreshholdPercent, c2900PortFrameAge,
--    c2900PortAddrSecureMaxAddresses,
--    c2900PortAddrSecureCurrentAddresses,
--    c2900PortAddrSecureAddrViolations,
--    c2900PortNumberOfLearnedAddresses,
--    c2900PortNumberOfDroppedAddresses, c2900PortClearAddresses,
--    c2900PortFloodUnknownMulticasts,
--    c2900PortFloodUnknownUnicasts, c2900PortLinkbeatStatus,
--    c2900PortBroadcastStormAction,
--    c2900PortBroadcastRisingThreshold,
--    c2900PortBroadcastFallingThreshold, c2900PortStatus,
--    c2900PortTestResult, c2900PortVisualIndicator,
--    c2900PortIfIndex, c2900PortAddressViolationAction,
--    c2900PortBroadcastStormAlarm, c2900PortMonitorDestinationPort,
```

```
--    c2900PortSwitchPortIndex, c2900PortMonitoredPortMap,
--    c2900PortDuplexState, c2900PortDuplexStatus,
--    c2900PortAdminSpeed, c2900PortNoMonitorDestinationPort

c2900PortStatsGroup OBJECT IDENTIFIER ::= { c2900MIBGroups 5 }
-- OBJECT-GROUP
--  Status
--    deprecated
--  Descr
--    The collection of objects which are used to
--     provide port stats.
--  objects
--    c2900PortRxNoBwFrames, c2900PortRxNoBufferFrames,
--    c2900PortRxNoDestUniFrames, c2900PortRxNoDestMultiFrames,
--    c2900PortRxSuppressBcastFrames, c2900PortRxFcsErrFrames,
--    c2900PortCollFragFrames, c2900PortTxMulticastFrames,
--    c2900PortTxBroadcastFrames

c2900BandwidthUsageGroup OBJECT IDENTIFIER ::= { c2900MIBGroups 6 }
-- OBJECT-GROUP
--  Status
--    mandatory
--  Descr
--    The collection of objects which are used to
--     provide the bandwidth information.
--  objects
--    c2900BandwidthUsageCurrent, c2900BandwidthUsageMaxPeakEntries,
--    c2900BandwidthUsagePeakInterval,
--    c2900BandwidthUsagePeakRestart, c2900BandwidthUsagePeakIndex,
--    c2900BandwidthUsageStartTime, c2900BandwidthUsagePeak,
--    c2900BandwidthUsagePeakTime,
--    c2900BandwidthUsageCurrentPeakEntry
```

```
c2900PortGroupSA3 OBJECT IDENTIFIER ::= { c2900MIBGroups 7 }
-- OBJECT-GROUP
--  Status
--    mandatory
--  Descr
--    The collection of objects which are used to
--     provide port status and configuration.
--  objects
--    c2900PortUsageApplication, c2900PortGroupIndex,
--    c2900PortMayLearnAddress, c2900PortMayForwardFrames,
--    c2900PortBufferCongestionThreshholdPercent, c2900PortFrameAge,
--    c2900PortAddrSecureMaxAddresses,
--    c2900PortAddrSecureCurrentAddresses,
--    c2900PortAddrSecureAddrViolations,
--    c2900PortNumberOfLearnedAddresses,
--    c2900PortNumberOfDroppedAddresses, c2900PortClearAddresses,
--    c2900PortFloodUnknownMulticasts,
--    c2900PortFloodUnknownUnicasts, c2900PortLinkbeatStatus,
--    c2900PortBroadcastStormAction,
--    c2900PortBroadcastRisingThreshold,
--    c2900PortBroadcastFallingThreshold, c2900PortStatus,
--    c2900PortTestResult, c2900PortVisualIndicator,
--    c2900PortIfIndex, c2900PortAddressViolationAction,
--    c2900PortBroadcastStormAlarm, c2900PortMonitorDestinationPort,
--    c2900PortSwitchPortIndex, c2900PortMonitoredPortMap,
--    c2900PortDuplexState, c2900PortDuplexStatus,
--    c2900PortAdminSpeed, c2900PortNoMonitorDestinationPort
c2900PortStatsGroupSA3 OBJECT IDENTIFIER ::= { c2900MIBGroups 8 }
-- OBJECT-GROUP
--  Status
--    mandatory
--  Descr
--    The collection of objects which are used to
```

```
--     provide port stats.
--  objects
--     c2900PortRxNoBwFrames, c2900PortRxNoBufferFrames,
--     c2900PortRxNoDestUniFrames, c2900PortRxNoDestMultiFrames,
--     c2900PortRxFcsErrFrames, c2900PortCollFragFrames,
--     c2900PortTxMulticastFrames, c2900PortTxBroadcastFrames

c2900MIBCompliance OBJECT IDENTIFIER ::= { c2900MIBCompliances 1 }
-- MODULE-COMPLIANCE
--  Status
--     deprecated
--  Descr
--     The  compliance statement for all c2900 switch.
--  Module
--     >>current<<
--    MandGroup
--     c2900SysInfoGroup
--    MandGroup
--     c2900SysConfigGroup
--    MandGroup
--     c2900PortGroup
--    MandGroup
--     c2900BandwidthUsageGroup

c2900MIBComplianceSA3 OBJECT IDENTIFIER ::= { c2900MIBCompliances 2 }
-- MODULE-COMPLIANCE
--  Status
--     mandatory
--  Descr
--     The  compliance statement for all c2900 switch.
--  Module
--     >>current<<
--    MandGroup
```

```
--    c2900SysInfoGroup
--   MandGroup
--    c2900ModuleGroup
--   MandGroup
--    c2900BandwidthUsageGroup
--   MandGroup
--    c2900PortGroupSA3
--   MandGroup
--    c2900PortStatsGroupSA3
END
```

BIBLIOGRAFIA:

1) Wikipedia,ProtocoloSimpleSNMP, http://es.wikipedia.org/wiki/Simple_Network_Management_Protocol, 10/11/2010

2) RincondeVago,AdministraciondeRedes, http://html.rincondelvago.com/administracion-de-redes.html, 12/11/2010

3) Universidad de Alcala, http://it.aut.uah.es/mar/gestion/tema2.pdf, 12/12/2010

4) Universidad de Castilla, Mantenimiento de Redes, www.info-ab.uclm.es/asignaturas/42524/teoria/ar2Tema7x2.pdf, 15/12/2010

5) Universidad de Castilla, Protocolo de SNMPv3, http://www.info-ab.uclm.es/asignaturas/42621/transpas/dmrTema4-SNMPv3.pdf, 15/12/2010

6) Kioskea, Componentes Windows, http://es.kioskea.net/faq/2914-administrar-los-componentes-de-windows, 18/12/2010

7) Ajuca,ComandosSNMP, http://www.ajuca.com/modules.php?name=News&file=article&sid=1385/introducci%F3n-a-los-comandos-snmp,-snmpwalk,-snmpget,-snmptranslate...--, 18/12/2010

8) UniverisdaddeCatalunya,GestiondeLinux, www.uoc.edu/master/softwarelibre/esp/materials/Admin_GNULinux.pdf, 01/12/2010

9) Scribd,ConfiguracionSNMP, http://www.scribd.com/doc/8751367/manual-de-configuracion-Snmp-Grupo1, 20/12/2010

10) InfoSec,SNMP, http://www.infosecwriters.com/text_resources/pdf/SNMP_BMatt.pdf, 22/12/2010

11) UniversidaddeAlcala,TutorialSNMP, http://it.aut.uah.es/enrique/personal/documentos/tutorial-net-snmp.pdf, 23/12/2010

12) Wikepedia, MIB, http://es.wikipedia.org/wiki/Management_Information_Base, 29/12/2010

13) Monografias,AdministraciondeRedes, http://www.monografias.com/trabajos43/administracion-redes/administracion-redes2.shtml, 22/12/2010

14) Tamps, Redes, http://www.tamps.cinvestav.mx/~vjsosa/clases/redes/MIB.pdf , 20/12/2010

15) PierrickSIMIER;HadwareSNMP, http://www.snmplink.org/snmpappliance/hardware/ , 05/01/2011

16) Cisco,ConfigurarSNMP, http://www.cisco.com/en/US/tech/tk648/tk362/technologies_tech_note09186a0080094aa4.shtml, 01/01/2011