SISTEMA DE GESTIÓN DE REDES EHAS

Arnau Sánchez Sala, Oscar Ramos Moreno, Eva Juliana Maya Ortiz

Fundación EHAS, Universidad Politécnica de Madrid, Universidad del Cauca

arnau@ehas.org, oscarr@ehas.org, emaya@unicauca.edu.co

RESUMEN

El proyecto EHAS busca mejorar los servicios de salud en las zonas rurales de los países de América Latina a través del uso de tecnologías adecuadas de información y comunicaciones, y para cumplir con este objetivo proporciona redes de comunicaciones y servicios de información. Esto hace evidente la necesidad de un sistema de gestión que permita realizar tareas de monitoreo y control sobre las redes EHAS para cumplir con el objetivo planteado. En este artículo se describe la arquitectura del sistema de gestión, las diferentes alternativas para gestionar estaciones *VHF*, *HF* y enrutadores *Wi-Fi*, se describe la herramienta utilizada, y finalmente los componentes desarrollados para el gestor y el equipo gestionado además de las adaptaciones realizadas a la herramienta de gestión seleccionada para lograr la gestión de las redes EHAS.

1. INTRODUCCIÓN

El principal objetivo del proyecto EHAS es mejorar los servicios de salud en las zonas rurales de los países de América Latina. Para lograr este objetivo el proyecto proporciona redes de comunicaciones y ofrece diferentes servicios de información a través de esas redes. Las redes de comunicaciones permiten conectar los puestos de salud con los centros de salud y hospitales, y a estos a Internet, y los servicios permiten la comunicación de voz y datos.

Las tecnologías que se utilizan en las redes EHAS son *VHF*, *HF* y *Wi-Fi* ya que son tecnologías de bajo costo, adecuadas para resolver los problemas de conectividad de las diferentes regiones en las que actúa el proyecto. Los principales componentes de estas redes son las estaciones tanto *VHF* y *HF*, y los enrutadores inalámbricos pero de ninguno de estos equipos se conoce su estado, su desempeño, sus fallas, y no es posible realizar ninguna acción sobre ellos de una forma automatizada.

Teniendo en cuenta el objetivo que busca alcanzar el proyecto y la ausencia de un sistema que permita monitorear y controlar las redes EHAS fue clara la necesidad de tener un sistema de gestión de red que permita saber si existe algún problema para resolverlo oportunamente, detectar su causa, tomar decisiones al respecto y realizar acciones sobre los equipos, y así lograr que la red opere adecuadamente la mayor parte del tiempo.

2. ARQUITECTURA

El sistema de gestión de red va a permitir gestionar estaciones *VHF*, *HF* y enrutadores inalámbricos. Las estaciones *VHF* y *HF* por naturaleza están desconectadas y los enrutadores inalámbricos aunque idealmente siempre están conectados pueden tener períodos de desconexión. Teniendo en cuenta esto se pensó que en el sistema de gestión el gestor no debe depender de la conectividad con los equipos gestionados para poder realizar sus tareas de monitoreo y control, y que los equipos gestionados deben recolectar su propia información de gestión y además tomar la iniciativa de enviarla en el momento que tengan conexión, por tanto este sistema debe basarse en el uso de correo electrónico. La arquitectura del sistema de gestión se muestra en la siguiente figura:

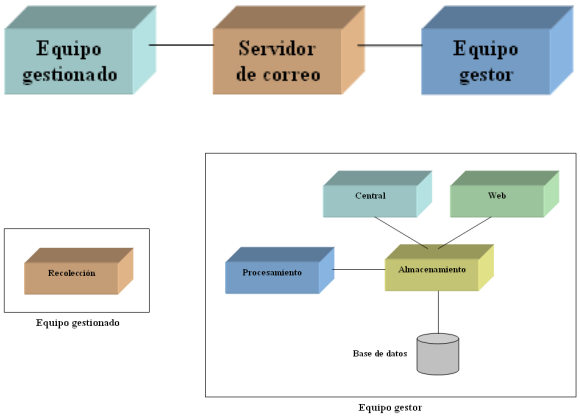

Figura1. Arquitectura del sistema de gestión

El módulo de recolección obtiene la información de gestión del equipo, la comprima y la envía como adjunto en un correo electrónico al gestor. El módulo de procesamiento realiza la adición de equipos al sistema, además obtiene los correos que llegan al gestor y que contienen información de gestión, consigue el adjunto de cada correo, lo descomprime y lo procesa para obtener los datos de gestión deseados e insertarlos en la base de datos. El módulo central realiza el monitoreo y control, es decir, es el que realiza las tareas del gestor, el módulo de almacenamiento es el encargado de todo lo relacionado con la base de datos e interactúa con el módulo de procesamiento, el módulo central y el módulo Web, y el módulo Web permite el despliegue de la información que se encuentra en la base de datos una forma agradable para el usuario.

Gracias a la interacción de estos módulos el sistema de gestión debe permitir conocer los equipos gestionados, desplegar la información de gestión en forma gráfica y textual, manejar eventos, generar alertas y alarmas, y permitir la ejecución de acciones sobre los equipos gestionados para tener un sistema de monitoreo y control adecuado para resolver las necesidades de gestión detectadas.

3. ALTERNATIVAS

Se decidió buscar la forma de gestionar los enrutadores inalámbricos y las estaciones. En principio se realizó esta división porque los enrutadores inalámbricos idealmente tienen conectividad permanente mientras que las estaciones *VHF* y *HF* no. Para la gestión de enrutadores inalámbricos se pensó en utilizar *SNMP* porque es uno de los protocolos de gestión de red más implementado, es muy adecuado para este caso porque es menos complejo que otros protocolos de gestión, y además sigue la filosofía del mejor esfuerzo. Entonces se buscó un agente *SNMP* que fuera software libre, que permitiera la gestión de tarjetas inalámbricas, y que además funcionara en *Pebble* que es el sistema operativo de los enrutadores inalámbricos, y se encontró que una empresa llamada *Avantcom* [1] proporciona el soporte de las *MIBs 802.11* y *802.11* de *Avantcom* para el agente *Net-SNMP*, y además proporciona el agente *SNMP* para *Pebble* con soporte para estas dos *MIBs*, bases de información de gestión para las tarjetas inalámbricas *Prism II* y *Atheros* que son las que se utilizan en el proyecto.

Se realizaron pruebas con este agente *SNMP* y se utilizó para conocer por medio de correos electrónicos el estado y la disponibilidad diaria de los enrutadores inalámbricos de uno de los enlaces que se tienen instalados. Se ejecuta una operación *SNMP* cada 5 minutos sobre cada enrutador para determinar si el enrutador está funcionando o no, si el enrutador está funcionando pero no lo había estado haciendo se envía un correo a los responsables de la red que les informa que el enrutador nuevamente está en funcionamiento, si el enrutador no estaba funcionando se determina el tiempo en el que el enrutador no ha estado funcionando porque si al cabo de 2 horas consecutivas no ha habido respuesta del enrutador se envía un correo a los responsables de la red que les informa que el enrutador dejó de funcionar. Con base en las respuestas de cada enrutador al final del día se obtiene el porcentaje en el que el enrutador estuvo disponible y se informa en un correo electrónico a los responsables de la red.

Una vez que se comprobó el funcionamiento del agente *SNMP* para los enrutadores se estudiaron diferentes herramientas de fuente abierta, entre ellas *MRTG*, una herramientas bastante conocida en el mundo de la gestión *SNMP*, *Cricket*, una herramienta que viene incluida en *Pebble*, y *Cacti*, una herramienta bastante potente y muy utilizada para la gestión a través de *SNMP* por sus más y mejores características con respecto a otras herramientas de este tipo. En los tres casos se encontró que era necesaria una adaptación de la herramienta para que incorporara correo electrónico que es el elemento principal de arquitectura planteada para el sistema de gestión y para que además permitiera gestionar los enrutadores de la forma más completa posible, lo que incluye la gestión de los clientes de las tarjetas inalámbricas que tienen un tratamiento especial en las *MIBs* inalámbricas. Teniendo en cuenta esto, se vió la posibilidad de desarrollar una aplicación propia, adecuada a las necesidades del proyecto y para eso se estudio la herramienta en la que se basan las herramientas mencionadas anteriormente y muchas de las herramientas de gestión, esa herramienta se llama *RRDTool*. *RRDTool* [2] es una herramienta que permite trabajar con bases de datos *Round Robin*, es decir, bases de datos que no crecen con el tiempo porque tienen un comportamiento circular, es decir, la base de datos tiene un tamaño determinado y cuando se escribe el último dato, el siguiente dato sobrescribe el primero y así sucesivamente. *RRDTool* es una herramienta que no solo se utiliza en el campo de la gestión.

RRTool permite crear, actualizar, extraer datos, graficar y desplegar datos, etc. de bases de datos *Round Robin*. En vista de la potencia de esta herramienta se realizó un prototipo que permite ver vía Web los enrutadores inalámbricos gestionados con sus clientes, permite la selección de un enrutador o de un cliente, de las variables y del período de tiempo del que desea ver el gráfico, y despliega el gráfico y los datos de las variables en el período de tiempo especificado. Este prototipo cumple con la arquitectura planteada para el sistema de gestión y fue desarrollado utilizando *Perl* y *PHP*, y requiere *Mysql* y *Apache*. En el equipo gestor se definen las interfaces a monitorear, las variables a monitorear de

cada interfaz y el tipo de dato de cada una de ellas, además se realizan operaciones *SNMP* como *get*, *walk* y *translate* para obtener los datos deseados tanto de las interfaces como de sus clientes, teniendo en cuenta el tratamiento que tienen éstos últimos en las *MIBs* inalámbricas, y finalmente, se utilizan diferentes utilidades de *RRDTool* para crear, actualizar y extraer datos de las bases de datos *Round Robin* creadas para cada interfaz y cada uno de sus clientes si los hay, además se comprime la información extraída de las bases de datos y se envía en un correo electrónico a una cuenta especificada.

En el gestor solo se obtienen los correos electrónicos que contienen información de gestión, se consigue su adjunto, y al igual que en el equipo gestionado se utilizan diferentes utilidades de *RRDTool* para crear una base de datos *Round Robin* para cada interfaz y para cliente, actualizar cada una de esas bases de datos y graficar las variables y desplegar los datos de las variables en el período de tiempo especificado para un enrutador o un cliente seleccionado por el usuario a través de una interfaz Web.

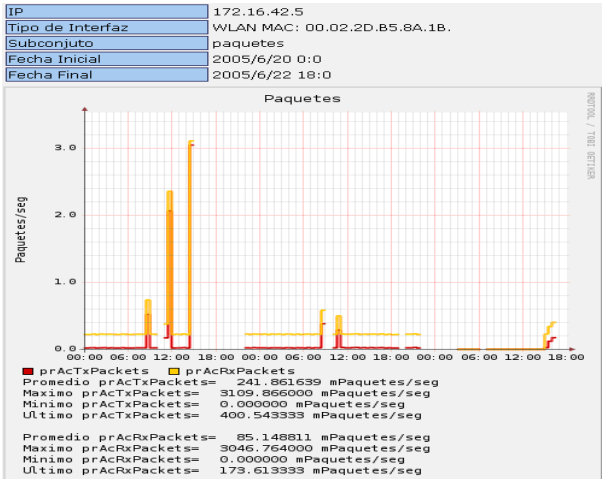

Figura2. Prototipo con RRDTool y SNMP

Para la gestión de estaciones *VHF* y *HF* se pensó en utilizar comandos del sistema y logs para obtener la información de gestión necesaria, y se realizó una implementación para obtener estos datos. Además se compararon herramientas de gestión como *Nagios* y *Zabbix* que son herramientas de monitoreo de redes de fuente abierta, muy potentes y utilizadas.

Al comparar *Nagios* y *Zabbix* se puede decir que *Nagios* es una herramienta más desarrollada que *Zabbix*, pero esto se debe a que *Nagios* es una herramienta que se creó un poco antes que *Zabbix*, pero también se puede afirmar que *Zabbix* día tras día tiene más personas y empresas interesadas en ella que facilitan su rápida evolución. *Zabbix* es una herramienta que está creciendo rápidamente gracias al trabajo constante de su grupo desarrollador y a sugerencias y contribuciones realizadas por sus usuarios, incluso por usuarios de *Nagios* que ven en *Zabbix* una posibilidad muy interesante para el monitereo de sus redes. Una muestra del rápido avance de *Zabbix* son las notables diferencias y mejoras de la versión 1.1 con respecto a la versión 1.0. *Nagios* tiene una característica particular y es que las funcionalidades adicionales se encuentran en módulos independientes, por un lado esto hace que *Nagios* sea una aplicación extensible, pero por otro implica que se tiene que tratar con aplicaciones diferentes para *conseguir* la funcionalidad deseada. *Zabbix* por el contrario contiene toda su funcionalidad, no requiere de módulos adicionales y tiene una arquitectura bastante flexible y modular que permite adicionar casi cualquier funcionalidad requerida.

Teniendo en cuenta esto se realizó un prototipo [3] que sigue la arquitectura del sistema de gestión planteada y que consistió en la adición de funcionalidad a la versión 1.0 de *Zabbix*. Este prototipo permitió obtener los adjuntos de los correos con información de gestión, adicionar equipos a la base de datos de *Zabbix*, procesar la información de los adjuntos para obtener los datos de gestión de las estaciones e insertar esos datos, y para algunos su "timestamp", a la base de datos de *Zabbix*, y además desplegar vía Web la información del sistema, los equipos adicionados y ejecutar comandos en forma remota segura sobre un equipo.

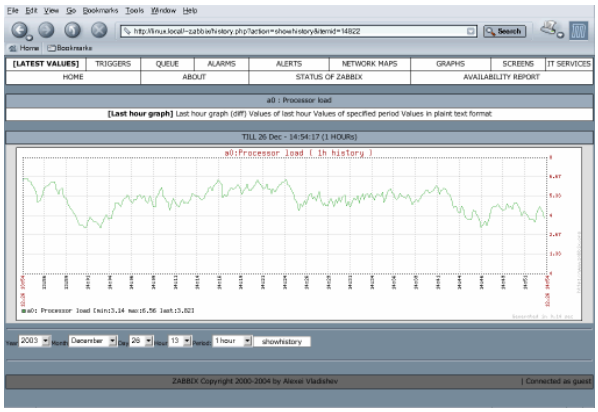

Figura 3. Zabbix 1.0

Una vez analizadas las posibilidades para la gestión de enrutadores y estaciones *VHF* y *HF* se decidió conformar un módulo para la recolección de información de las estaciones a partir de la implementación realizada para este fin, además de la realizada para monitorear interfaces de red, e incluir este módulo en otro que permite la configuración de una estación. Una vez conformado este módulo de configuración de las estaciones se decidió crear un módulo equivalente para los enrutadores, un módulo que permite su configuración y gestión. También se decidió utilizar *Zabbix* versión 1.1 porque esta versión tiene muchas mejoras con respecto a la versión 1.0. *Zabbix* 1.1 tiene diferencias conceptuales

con respecto a la versión 1.0 pero esas diferencias son las que hacen que esta nueva versión sea mucho más organizada y completa. Debido a esto fue necesario estudiar la nueva versión de *Zabbix* para poder realizar las adiciones y modificaciones necesarias a esta herramienta y conseguir el sistema de gestión de redes deseado, y además se tuvieron en cuenta los nuevos módulos de recolección de información de los equipos gestionados y la arquitectura planteada para el sistema de gestión. De los módulos que se encuentran en el equipo gestionado y en el gestor se va a hablar un más adelante.

4. ZABBIX

Zabbix [4] es una herramienta de monitoreo de redes que permite el *polling* y *trapping* de datos de los equipos gestionados, es decir que los datos pueden ser solicitados por el gestor, o los datos pueden ser proporcionados por el equipo gestionado sin petición del gestor, esta herramienta permite desplegar gráficos, datos y mapas, además realizar la configuración de la herramienta vía Web y notificar la ocurrencia de eventos predefinidos. *Zabbix* cuenta con documentación y con un foro y listas de correo que hacen posible intercambiar experiencias con usuarios de esta herramienta que permite solucionar problemas de una manera más rápida, realizar sugerencias, reportar fallos de la herramienta y publicar parches para algún caso en particular.

Zabbix solo corre sobre *Linux* y requiere *Apache*, *PHP* y *Mysql*, por ahora la versión 1.1 no trabaja con *Postgres*. *Zabbix* tienen diferentes utilidades, en este caso solo se utiliza *zabbix-server* y *zabbix-sender*. *zabbix-server* es el que se encarga de toda la funcionalidad de la herramienta, es el que lleva a cabo todas las tareas del gestor, y *zabbix-sender* es una utilidad de línea de comandos que permite la inserción del valor de una variable en la base de datos de una forma más fácil. Esta utilidad es importante en este caso ya que permite insertar los valores de los datos obtenidos del procesamiento de los logs de gestión que llegan como adjuntos en los correos de gestión que llegan al gestor.

Gracias a su base de datos, *Zabbix* permite el manejo de usuarios y grupos de usuarios de la herramienta, equipos y grupos de equipos gestionados, variables a monitorear, disparadores de eventos, alertas, alarmas, acciones cuando se producen eventos, datos históricos de las variables monitoreadas, mapas, gráficos, entre otros aspectos.

La interfaz Web de *Zabbix* es muy amigable y permite escoger el grupo de equipos, el equipo gestionado y diferentes opciones para visualizar los valores de variables en forma textual o en un gráfico, los eventos producidos, las alertas y alarmas, los mapas y gráficos entre otros, y además a los usuarios con derechos de administrador les permite realizar adiciones, actualizaciones y modificaciones de los usuarios de la herramienta, grupos de equipos, y equipos gestionados, disparadores de eventos, acciones cuando se producen eventos, gráficos y mapas, además para estos últimos permite la adición de íconos y de imágenes de fondo para hacer su despliegue más agradables. Los gráficos pueden tener dos o más variables y los mapas pueden tener varios equipos. Si los equipos se encuentran conectados se utilizan enlaces que se asocian con un evento con lo que se puede conocer rápidamente el estado de los elementos de la red.

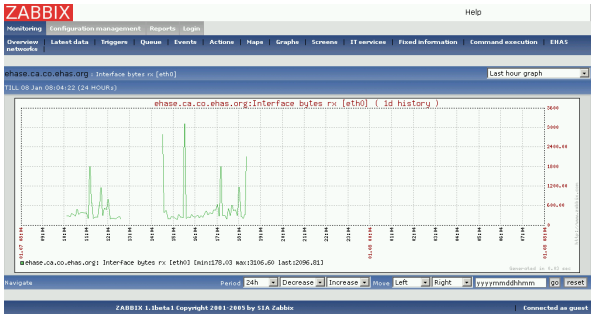

Figura 4. Zabbix 1.1

5. EQUIPO GESTIONADO

Como se mencionó anteriormente, un equipo gestionado puede ser una estación o un enrutador inalámbrico. Una estación debe tener instalado *ehas-station*, *ehas-netman* y *grunt-ehas* además de *Procmail*, un agente de entrega de correo, y *Postfix*, un agente de transferencia de correo. Un enrutador debe tener instalado *ehas-router*, *ehas-netman*, *grunt-ehas*, además *erouterboard*, *Procmail* y *Masqmail* que al igual que *Postfix* es un agente de transferencia de correo. A continuación se va a describir estos elementos [5].

ehas-station contiene a *config-ehas*, una aplicación que permite configurar una estación a través de una interfaz gráfica. El equivalente de *ehas-station* para los enrutadores inalámbricos se llama *ehas-router* y contiene a *config-router* que permite configurar un enrutador a través de una interfaz gráfica. *config-ehas* y *config router* permiten configurar si el envío de logs está activo, el correo electrónico al que se envían los logs, el modelo de la estación o del enrutador, el tiempo en el que se envía el correo que indica que la estación está "alive", que puede ser cada 2 horas, 8 horas, 2 días, 6 días o 15 días, además permiten establecer las interfaces de red que se desean monitorear: *loopback*, *ethernet*, *wi-fi*. *config-ehas* también permite determinar si el monitoreo diario de correo electrónico, placa de interfaz, conexiones de radio, conexiones por módem y telefónicas está activo mientras que *config-router* si el

monitoreo diario de correo electrónico, placa de interfaz, y conexiones telefónicas está activo, y además permiten enviar la información del sistema. A continuación se muestra una imagen de *config-ehas* y *config-router*.

Figura 5. config-ehas

Figura 6. config-router

La configuración que se realiza a través de *config-ehas* o *config-router* se almacena en un archivo en el sistema, así mismo la configuración de gestión de la estación o el enrutador se almacena en un archivo que contiene el correo electrónico al que se van a enviar los logs, los diferentes períodos en los que se va a obtener la información que se desea monitorear: 5 minutos, período de "alive" escogido por el usuario y 1 día, además las acciones que van a permitir obtener la información de cada uno de los logs, el log de información del sistema, el log con datos de cada 5 minutos, el log de "alive" y el log diario.

ehas-netman se encarga de que la información de cada uno de los logs se obtenga en los momentos adecuados gracias a la utilización del *cron* o programador de tareas del sistema. Para obtener la información de gestión se ejecutan las acciones especificadas en el archivo que tiene la configuración de gestión. Esas acciones se encuentran implementadas en *ehas-station*, *ehas-router* y *ehas-netman*, y permiten obtener la información de gestión deseada a través de la ejecución de un comando o de un log del sistema. *ehas-netman* realiza diferentes operaciones sobre los logs del sistema y si es el caso envía el correo electrónico con el log como adjunto comprimido en formato .gz. Los logs contienen la siguiente información:

Log con información del sistema contiene:

- Nombre del equipo
- Paquetes instalados
- Tabla de particiones
- Sistema de archivos
- Memoria
- Hardware *PCI*
- Hardware *USB*
- Hardware *PCMCIA*
- Información de configuración
- Información de configuración de gestión

Log con información de cada 5 minutos

- Temperatura y estado de la *CPU*. Se utiliza *ACPI*. Se registra la temperatura y el estado de la *CPU*
- Tiempo de encendido. Se registran el intervalo de tiempo en el que la estación estuvo encendida.
- Interfaces. De esto se hablará un poco más adelante.

Log "alive"

- Si el período de "alive" es inferior a 1 día la información de gestión de las interfaces monitoreadas se envían como adjunto en este correo, pero si el período de "alive" es superior a 1 día esta información se envían junto con la información de gestión diaria.

Log diario

- Estado de discos duros. Se utiliza *Smartools* que proporciona información de los discos duros, permite saber si hubo algún problema o si va a haber alguno en las próximas horas, etc.
- "Crashes" del sistema
- Espacio total, usado y disponible en bytes e inodos
- Estadísiticas de la *CPU*. Se analizan los datos registrados en el log con información de cada 5 minutos y se determina la temperatura mínima, el estado de la *CPU* y su "timestamp", la temperatura máxima, el estado de la CPU y su "timestamp", y se obtiene el promedio de las temperaturas, el "timestamp" inicial y final de los datos registrados, y a partir de la temperatura promedio se determina el estado de la *CPU* para lo que se tienen en cuenta los estados especificados por *ACPI* según la temperatura.
- Estadísticas de encendido. Se leen los datos almacenados en el log con información de cada 5 minutos y se determinan los intervalos de tiempo en los que el computador estuvo encendido.

- Correo electrónico, placa de interfaz y conexiones por módem. Se obtiene la información registrada en el sistema para cada uno de ellos. La información de correo electrónico contiene correos entrantes y salientes, etc. La información de la placa de interfaz contiene el nivel de batería, y *SWR* y temperatura de cada una de las conexiones de radio.
- Telefonía. Se utiliza un log de *Asterisk*, que es la herramienta que permite integrar la *PSTN* e Internet. Ese log contiene varios datos, entre ellos los tipos de llamadas, extensiones desde las que se han realizado llamadas y de las cuales se ha recibido llamadas, etc.
- Cola de correo. Se determina el número de correos electrónicos a espera de ser enviados, se determina el tamaño en bytes de esos mensajes y el tiempo más largo que un mensaje ha estado en la cola.
- Conexiones por radio. Se utiliza la información registrada en el sistema para cada conexión tanto *VHF* como *HF* y se calcula el tiempo de uso del proxy, además se determina la *BER VHF* y *HF*, y el "timestamp" inicial y final de los datos medidos, velocidad *VHF* de transmisión y de recepción teniendo en cuenta que la velocidad *VHF* solo puede tener ciertos valores, la velocidad *HF* de transmisión y de recepción así como también la *SNR* máximas, mínimas y los "timestamp" en los que se producen esos valores, y se calculan los promedios de cada una de ellas y se determina el intervalo de tiempo en el que se realizaron las medidas.

Para el monitoreo de las interfaces en un principio se pensó utilizar *SNMP* como se comentó anteriormente, pero se encontró que el agente *SNMP* de *Avantcom* depende de la versión del driver de la tarjeta para obtener toda la información de gestión de las *MIBs* inalámbricas, entonces se estudiaron algunas herramientas que permiten el monitoreo de interfaces, e incluso se vió la forma en la que trabaja el agente *SNMP*, y se encontraron comandos y archivos del sistema que permiten obtener la información de gestión que se necesita.

Teniendo en cuenta se realizó una nueva implementación que contiene las variables a monitorear de las posibles interfaces de una estación o enrutador inalámbrico, el tipo de dato de cada variable, esto fue necesario ya que se identificaron dos tipos de datos, el primero para los datos que se utilizan tal como son leídos y otro tipo para los datos que se obtienen como la diferencia entre el valor actual y el valor inmediatamente anterior dividido por la diferencia entre los tiempos en que se realizaron sus lecturas, y además contiene información para poder realizar el filtrado de la información de gestión y obtener el dato. Esta implantación utiliza comandos y archivos del sistema para obtener información de las interfaces, como por ejemplo, para todas las interfaces. paquetes, bytes y errores tx y rx, para las interfaces inalámbricas: punto de acceso, tasa de bits, señal, ruido, para las tarjetas *Prism*: octetos unicast y multicast tx y rx, para las tarjetas *Atheros*: reintentos fallidos en tx, fallas por CRC malo en rx, para los clientes de las tarjetas *Prism:* señal, ruido, tasa de bits, paquetes a 1, 2, 5 y 11 Mbps de tx y rx, y para los clientes de las tarjetas *Atheros*: señal, ruido, entre otros.

Hasta ahora se ha hablado de monitorear estaciones o enrutadores inalámbricos pero también es posible controlarlos, para eso se utiliza *grunt* que permite la ejecución remota segura de comandos. La seguridad de *grunt* se debe a que trabaja junto con *gpg* que permite firmar digitalmente la información. *grunt* contiene *gruntrun* y *gruntreceive*. *gruntrun* permite enviar un correo con un comando, y *gruntreceive* ejecuta el comando una vez llega el correo. En cuanto a los comandos es necesario ejecutar el comando que permite enviar la información del sistema para la adición de los equipos en el gestor y además ejecutar cualquier comando proporcionando un *password*. Debido a esto fue necesario crear un *alias* que ejecuta el comando que permite el envío de información fija, además de un par de claves privada (en el gestor) y pública (en el gestionado), y otro *alias* que permite la ejecución de cualquier comando, además de otro un par de claves privada (en el gestor) y pública (en el gestionado).

En el enrutador se escogió *Masqmail* debido a que es más simple y por lo tanto más liviano que *Postfix*, algo que se debe tener en cuenta en los enrutadores inalámbricos. *Masqmail* al igual que *Postfix* también maneja *aliases* y trabaja con *Procmail*.

6. GESTOR

6.1. Procesamiento de correo

Es una estación con sistema operativo *Linux* ya que *Zabbix*, la herramienta de gestión utilizada, no funciona sobre *Windows*, en esa estación además se debe instalar *Mysql*, *PHP* y *Apache*. *Mysql* porque es el motor de base de datos que ya utilizan las versiones más actuales de Zabbix, *PHP* porque *Zabbix* y los componentes desarrollados para el gestor se encuentran en este lenguaje, y *Apache* como servidor Web. Además *Procmail* y *Postfix*, y si se desea también se pueden instalar todos los paquetes necesarios en un equipo gestionado para gestionar el gestor.

En el gestor se crea un usuario, que es a quien se van a enviar los correos electrónicos con los logs, y se utiliza *Procmail* que permite la entrega de correos a sus destinatarios y además analizarlos. En este caso *Procmail* analiza el asunto de cada correo que llega al usuario creado y determina si el correo contiene un log

de gestión que puede ser un log con información del sistema, un log de "alive" o un log diario como se mencionó anteriormente. En cualquiera de estos casos se obtiene el nombre y el dominio del equipo gestionado a partir del remitente, además del adjunto, que viene comprimido en formato *.gz*, utilizando *munpack*, una utilidad que permite desempaquetar mensajes en formato *MIME* o en *UUencode*, en este caso los correos se encuentran en *UUenconde*, y los adjuntos obtenidos se almacenan en un directorio temporal. A partir del nombre y el dominio se obtiene el directorio del equipo, directorio en el que se van a almacenar los logs obtenidos.

Si el log que llega es un log con información del sistema se determina si existe el directorio del equipo, si no existe se crea y se realiza la adición del equipo en la base de datos de *Zabbix*, esto se va a explicar en detalle un poco más adelante, además, se obtienen las diferentes secciones del log que contiene la información del sistema y se almacenan en diferentes archivos en un directorio dentro del directorio del equipo. Estas secciones son: datos generales como el nombre del equipo, paquetes instalados, configuración, particiones, sistema de archivos, información de la memoria, hardware *PCI* y *USB*, y se utilizan para el despliegue de la información del sistema a través de la interfaz Web. Si el directorio existe se realiza la actualización del equipo, de la que se va a hablar en detalle un poco más adelante, y se obtienen nuevamente las secciones del log que contiene la información del sistema.

Si el log que llega es un log de "alive" o un log diario se verifica si existe el directorio del equipo, si no existe se le pide al equipo gestionado que envíe el log con información del sistema que como se comentó anteriormente permite la adición del equipo en el sistema de gestión. Para lograr esto se utiliza *gruntrun* con una de sus opciones y una vez llega el log con la información del sistema se realiza lo descrito anteriormente. Si el directorio del equipo existe se realiza el procesamiento del log de "alive" o diario del que se hablará más adelante. Además, si el log que llega es un log de "alive", se debe insertar un 1 en la variable "alive" del equipo para lo que se utiliza la utilidad *zabbix-sender*.

6.2. Adición y actualización de equipos

En cuanto a la adición y actualización de un equipo en la base de datos de *Zabbix*, primero que todo *Zabbix* utiliza plantillas, es decir un equipo que pertenece al grupo plantillas de la base de datos de *Zabbix*. Una plantilla tiene variables, disparadores de eventos, acciones en eventos y gráficos. Se puede tener una plantilla que está enlazada con otra u otras plantillas, esta es una de las características más interesantes de las nuevas versiones de *Zabbix* que significa que esa plantilla contiene todo lo de las otras plantillas y es la plantilla que se puede utilizar para realizar la adición de un equipo. El enlace entre una plantillas y sus plantillas hace que se pueda adicionar, modificar o eliminar variables, disparadores de eventos, acciones en eventos y gráficos en cada una de ellas y eso se refleje en la plantilla que la contiene y por tanto en los equipos que se han adicionado con base en esa plantilla. En este caso se tiene una sola plantilla tanto para estaciones como enrutadores que se encuentra enlazada con otra plantilla que contiene todas las variables, disparadores de eventos, acciones en eventos y gráficos para los equipos gestionados.

Para realizar la adición y actualización de un equipo sin duplicar la funcionalidad existente en *Zabbix* se estudió su funcionamiento y se encontró que *Zabbix* separa muy bien la parte Web, de la parte de control y de la parte de la base de datos lo que permite lograr la independencia de cada una de estas partes. En la parte Web tiene páginas que permiten ver los diferentes aspectos que cubre *Zabbix*: variables, eventos, gráficos etc., en la parte de control tiene un módulo de control que incluye todos los otros módulos de control y contiene funciones generales y comunes. Existe un módulo de control para variables, disparadores de eventos, acciones en eventos, gráficos, etc., además de un módulo de control que se encarga de todo lo relacionado con la base de datos. Este módulo permite la conexión con la base de datos de Zabbix, realizar consultas y obtener los resultados de esas consultas a la base de datos que se puede encontrar en *Postgres* o *Mysql*. Este script es el que permite la independencia entre el control y la base de datos.

Para la adición de un equipo, al igual que para las otras funcionalidades que proporciona *Zabbix*, esta herramienta involucra una parte Web, una parte de control y una parte relacionada con la base de datos, entonces se identificaron los módulos que intervienen en este proceso y se estudiaron las funciones que se llevan a cabo para lograrlo. Se encontró que la parte de control y la parte relacionada con la base de datos de *Zabbix* se pueden reutilizar para realizar la adición de equipos pero que era necesario un módulo que cumpliera el mismo papel que la página Web que permite la adición de un equipo con base en una plantilla, es decir que consiguiera las variables necesarias, que para el caso de la página Web son proporcionadas por el usuario, y utilizara la funcionalidad de adición que se encuentra en la parte de control. Además se identificó que en cada una de las funciones que se llevan a cabo para adicionar un equipo se realiza un chequeo de permisos del usuario que inicia la sesión Web, algo que se debe tener presente si se van a ejecutar esas funciones pero ya no desde una página Web.

El módulo que se adicionó determina el país al que pertenece el equipo a gestionar y los diferentes

subdominios que tiene el equipo que son los grupos a los que se va a adicionar el equipo para lograr un mejor despliegue en la interfaz Web. También analiza la parte del log que contiene la información del sistema, exactamente la configuración de gestión, y determina las variables que se desea monitorear para solamente adicionar esas variables. Del log que contiene la información del sistema también se obtienen los discos duros, los puntos de montaje y las conexiones de radio *VHF* y *HF* ya que es necesario personalizar las variables para cada uno de estos elementos con su valor correspondiente. Finalmente, este módulo se encarga de realizar las llamadas necesarias a la funcionalidad de *Zabbix* que se encuentra en la parte de control para realizar la adición de un equipo.

Para la adición de un equipo, *Zabbix* determina si el equipo ya existe en la base de datos, si no existe se inserta en la base de datos, además se adicionan las plantillas al equipo, es decir las plantillas a las que está enlazada la plantilla que se escogió para realizar la adición, y en seguida se sincroniza el equipo con las plantillas, en nuestro caso como se mencionó anteriormente, con una sola plantilla, es decir, se adicionan al equipo las variables, disparadores de eventos, acciones en eventos y gráficos de la plantilla a la que se enlazó el equipo.

Para lograr la adición de las variables deseadas y además personalizadas se realizaron algunos cambios en *Zabbix* sin afectar su funcionamiento normal. Los cambios realizados se basan en la identificación de un patrón en la variable a adicionar, esa variable puede ser una variable sencilla como “alive”, una variable sencilla que solo se adiciona si está determinado por la configuración de gestión, una variable que se debe personalizar con un solo valor como un disco duro, un punto de montaje, una partición o una interfaz de red, o una variable que se debe personalizar con dos valores, una interfaz inalámbrica y uno de sus clientes. Antes de adicionar una variable *Zabbix* determina si la variable ya existe o no en el equipo. Además de esto también fueron necesarias algunas modificaciones para lograr la adición de disparadores de eventos y acciones en eventos.

Los disparadores de eventos tienen expresiones que contienen una variable, una función que se ejecuta sobre esa variable y un valor que se compara con el resultado de la función ejecutada sobre la variable. Teniendo en cuenta esto, dependiendo de la variable, puede ser necesario personalizar el valor del disparador del evento, como es el caso de la variable “alive” para la que el disparador de evento debe tener como valor el establecido por el usuario en la opción de “stillalive” de *config-ehas* y que se encuentra en el log que contiene la información del sistema, o puede ser necesario adicionar el disparador de evento para cada una de las variables personalizadas, como es el caso de la variable de espacio disponible para cada punto de montaje que también se obtuvieron del log que contiene la información del sistema.

Las acciones a realizar en el caso de que se presente un evento consisten en el envío de un correo electrónico a los responsables del equipo. Para lograr esto se obtienen los responsables de la red del país al que pertenece el equipo y se establecen para las diferentes acciones de los disparadores de eventos. La adición de acciones a un disparador de evento se realiza teniendo en cuenta su descripción por lo que la actualización de la descripción de los disparadores de eventos solo se hace cuando se han adicionado las acciones, esas descripciones son importantes para la configuración de mapas.

Para realizar la actualización de un equipo también se llama a la funcionalidad de la parte de control que se encarga de la adición de un equipo ya que antes de la adición de una variable se determina si esa variable existe o no para el equipo en la base de datos, entonces no habría variables duplicadas y se agregarían las nuevas variables, pero para los disparadores de eventos y acciones en eventos no se realiza esa validación, entonces se decidió eliminar de la base de datos los disparadores de eventos y acciones en eventos con toda su información relacionada y se encontró que *Zabbix* realiza esta función entonces se decidió reutilizarla sin necesidad de realizar ninguna modificación. Una vez que se borran los disparadores de eventos y acciones en eventos, y toda la información relacionada con cada disparador de evento del equipo a actualizar se llama a la funcionalidad de la parte de control de *Zabbix* que se encarga de la adición de un equipo.

Para lograr esto además de la funcionalidad de *Zabbix* que se utilizó tal como está y las modificaciones realizadas para lograr el resultado comentado anteriormente, se agregó otra funcionalidad teniendo en cuenta la forma en la que trabaja *Zabbix* con la que se logra la independencia del control y la base de datos. Por esto se estudió la forma en la que *Zabbix* realiza consultas y obtiene los resultados de las consultas sin tratar directamente todo el tiempo con funciones de un motor específico de base de datos.

6.3. Procesamiento de logs

En cuanto al procesamiento de los logs de “alive”, cuando éstos contienen la información de gestión de las interfaces, y de los logs diarios que contienen la información descrita en la sección de equipo gestionado, cada una de las líneas de los logs tienen el valor del “timestamp” en el que se realizó la lectura de los datos y en seguida una palabra que identifica el tipo de información que se envía, y la información de gestión, teniendo en cuenta esto se separan las diferentes

secciones que contienen los logs y se realiza su procesamiento para obtener los datos de cada una de ellas y su correspondiente "timestamp".

En cada sección, para cada una de las variables se define su tipo: numérico o caracter, que permite realizar la actualización correcta de la base de datos, y el nombre de la variable del equipo a actualizar, además se especifica el nombre de la variable de la plantilla y el valor o los dos valores necesarios para personalizar esa variable en caso de que se deba adicionar si no existe en la base de datos para el equipo.

Con esa información lo primero que se hace es determinar si ya existe algún dato para la variable del equipo con ese "timestamp" en la base de datos, si es así no se debe insertar el dato, en caso contrario se inserta el dato utilizando *zabbix-sender* proporcionando, entre otros parámetros, el nombre de la variable del equipo y su valor. Si *zabbix-sender* da un resultado negativo no se insertó el dato lo que significa que esa variable no está en la base de datos para el equipo, caso en el que se utiliza nuevamente la funcionalidad de *Zabbix* que permite la adición de variables, y que se modificó para permitir la adición de variables simples o que requieren uno o dos valores, desde el módulo que se encarga del procesamiento de los logs. Una vez adicionada la variable se ejecuta nuevamente *zabbix-sender* y si su resultado es positivo se actualizan los valores de "timestamp" en la base de datos ya que *zabbix-sender* inserta el "timestamp" en el que se realiza la operación y para esto se debe conocer el tipo de dato.

6.4. Web

En cuanto a la parte Web, se adicionaron cuatro pestañas a la sección de Monitorización. La primera pestaña permite el despliegue de un árbol con los subdominios y sus equipos, que permite la fácil visualización de la localización de un equipo en la red. La segunda pestaña permite el despliegue de la información del sistema de un equipo, es decir de los paquetes instalados, el hardware y la configuración, además permite guardar localmente esa información, la tercera pestaña permite ejecutar comandos en forma remota y segura sobre una máquina o un grupo de máquinas. A través de esta página se puede pedir el envío del log que contiene la información del sistema o ejecutar cualquier comando proporcionando además un password y una dirección de correo electrónico a la que va a llegar el resultado de la ejecución del comando, y la última pestaña permite el despliegue de logs diarios a partir de una fecha.

En cada una de las páginas se reutilizó el encabezamiento y el pie de página fr *Zabbix* y se dividió la funcionalidad en diferentes páginas. Para el árbol de subdominios, las validaciones para la ejecución de comandos y el despliegue de logs diarios, y el calendario en el despliegue de logs diarios se utilizó *javascript*. En las páginas que requieren la selección de grupo, es decir de un subdominio en nuestro caso, y de un equipo se realizan consultas a la base de datos de *Zabbix* y se realiza el despliegue de los nombres de los equipos de tal forma se complemente con el del grupo. En la página que contiene la información de paquetes instalados, hardware y configuración, y en la página de logs diarios se adicionaron dos enlaces que permiten guardar los datos localmente ya sea en formato *.xls* o *.txt*, para esto el enlace se direcciona a una página que coloca un encabezamiento a las páginas que se encargan del despliegue de esa información y que establece el tipo de contenido: texto plano o aplicación, además del nombre del archivo, que hacen que se despliegue la ventana con las opciones de Guardar o Salvar. Teniendo en cuenta esto fue necesario realizar dos opciones de despliegue de la información, una para el formato *txt* y otra para el despliegue en la página y en formato *xls*.

Para el despliegue de la información del sistema se obtiene el equipo, se determina su directorio y se leen los archivos necesarios del directorio que contiene las secciones del log que contiene la información del sistema. Para ejecutar remotamente el comando que envía la información del sistema de un equipo o para ejecutar cualquier comando se utiliza *gruntrun* con una de sus dos opciones, y finalmente para el despliegue del log diario, se obtiene el equipo y la fecha, y se despliega el log correspondiente que se encuentra en el directorio que contiene los logs dentro del directorio del equipo.

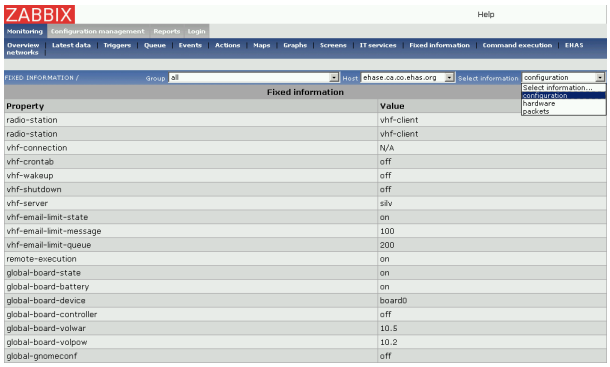

Figura 7. Zabbix 1.1 adaptado

7. PRUEBAS

Una vez terminada la implementación se realizaron pruebas con diferentes equipos: computadores, computadores en los que se configuraron conexiones *VHF* y *HF*, y en enrutadores inalámbricos, también, gracias a que se tienen equipos en el laboratorio que se van a instalar próximamente, se realizaron pruebas con M*ini-ITXs* y S*oekris*. El sistema parece funcionar bien, los equipos obtienen su propia información de gestión, la

envían por correo y el gestor la procesa, la almacena y despliega vía Web, además de permitir otras tareas de monitoreo y control.

8. CONCLUSIONES

- Una arquitectura en la que los equipos gestionados recolectan su información de gestión y la envían por correo electrónico al gestor permite realizar la gestión adecuada de equipos que no tienen conectividad permanente.
- Para gestionar una red se pueden utilizar estándares como *SNMP*, *WBEM*, etc., pero también se pueden utilizar comandos y logs del sistema para realizar las tareas de monitoreo y control.
- El software libre es muy importante porque permite ahorrar esfuerzos y tiempo en el desarrollo de aplicaciones, y además cuenta con el respaldo permanente de una gran comunidad.
- La modularización de una aplicación es sumamente importante para permitir que se le hagan adaptaciones y modificaciones necesarias para satisfacer necesidades específicas de otros usuarios.
- *RRDTool* es una herramienta bastante potente para el manejo de bases de datos Round Robin, muy utilizadas en el campo de la gestión así como también en otros campos.

9. REFERENCIAS

[1] http://www.avantcom.com

[2] http://people.ee.ethz.ch/~oetiker/Webtools/rrdtool/

[3] Oscar Ramos, Sistema de gestión de red para el programa EHAS, Sistema de monitorización en diferido basado en correo electrónico, Escuela Técnica Superior de Telecomunicación de la Universidad Politécnica de Madrid, 1-134, 2005.

[4] http://www.zabbix.com

[5] Documentación de los paquetes utilizados