CAPÍTULO 4: Implementación del sistema en el Escenario Real

En este capítulo se va a analizar el escenario real de la empresa, los requisitos necesarios tanto software como hardware para la instalación de la herramienta Zabbix y la instalación y configuración de dicha herramienta de la manera más adecuada, para satisfacer las necesidades descritas por los operadores de red de RTVA (*Radio y Televisión de Andalucía*). Asimismo, al final se comprobará, si las características hardware del sistema y de la red, admiten la configuración elegida. Se van a distinguir cuatro partes: estudio, instalación, configuración y análisis de resultados.

4.1 - ESTUDIO

Por petición del personal de la propia empresa, argumentando cuestiones de seguridad, no se van a especificar cada una de las subredes y tampoco las direcciones IP (*Internet Protocol*) de los distintos dispositivos. A modo de ejemplo y para entender mejor la topología, se mostrarán algunas de las subredes de RTVA y se estudiarán las posibles configuraciones de los distintos dispositivos de la red. Puesto que lo que más interesa, es cómo monitorizar un tipo de dispositivo, a lo largo del capítulo, se mencionarán todos los tipos existentes y la configuración elegida para cada uno. En los dispositivos de igual tipo, se tomará la misma configuración y por tanto, no será necesario volver a describirla. En primer lugar, se describirá de forma muy general, la topología y las características de la red interna de la empresa.

4.1.1 - TOPOLOGÍA DE LA RED DE RTVA

En realidad, se trata de una subred perteneciente a la Red Corporativa de la Junta de Andalucía. Su estructura es en estrella y coincide con la topología de cada una de las delegaciones, que siempre es más o menos la misma, aunque varían en número de dispositivos. Por ejemplo, la de San Juan de Aznalfarache reúne a la mayor parte de los servidores.

La nube azul de la figura 4.1, representa la Red Corporativa de la Junta de Andalucía, de cuya gestión y control se encarga la empresa Telefónica. Las distintas delegaciones de RTVA se conectan a esta, mediante unos *routers* CISCO.

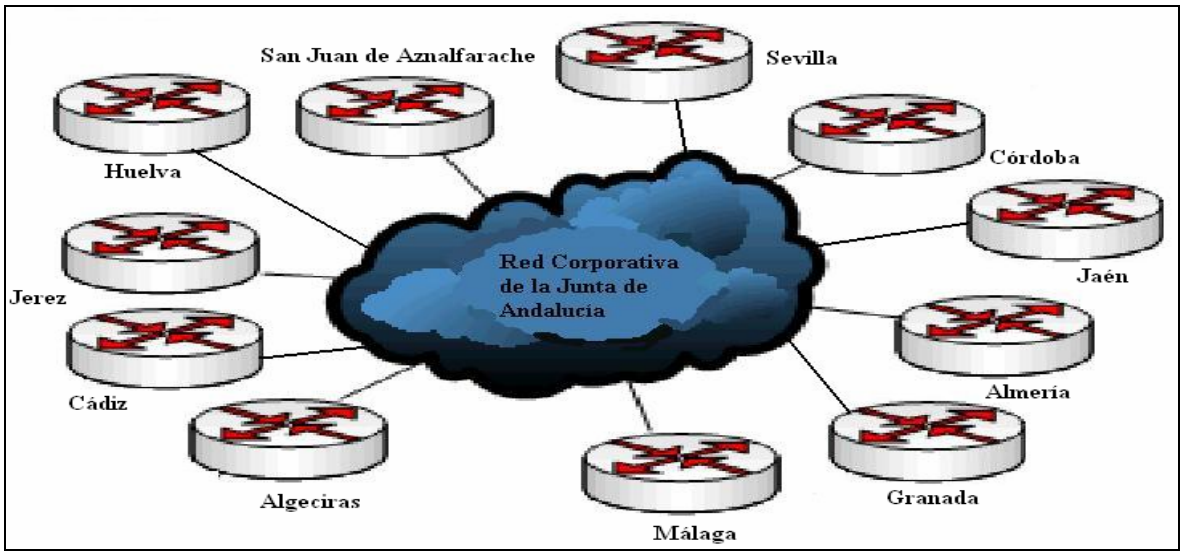


Figura 4.1 - Topología de la red de RTVA

Para hacerse una idea de las topologías de las distintas delegaciones, en la figura 4.2 se muestran la de Sevilla y la de Cádiz. Tal y como se observa, se trata de estructuras bastante sencillas, basadas en topologías en estrella, en la que se repiten distintos elementos:

- Dos *routers* CISCO que las unen a la Red Corporativa de la Junta de Andalucía. Uno de los *routers* es el principal y el que se utiliza normalmente, el otro es una copia exacta. Si existe algún fallo en el primero, no caerá la red de la delegación gracias al segundo.
- Una pila de *switches* centrales conectados entre sí. Suelen ser de la marca 3COM y para entender mejor la forma en la que se conectan, se muestra el esquema del aparato en la figura 4.2. Existen dos formas, mediante los puertos *matrix* o mediante los puertos *10BASE-T*.

Cabe resaltar, que la conexión de los *switches* mediante puertos *matrix*, no sólo permite utilizar al bloque como a un sólo dispositivo, sino que en general, permiten velocidades más altas que los 10BASE-T [9]. Sin embargo, algunos *switches* disponen de uno o dos puertos 10BASE-T, que ofrecen las mismas prestaciones que los primeros. Estos se utilizan para la conexión de distintas pilas o de *switches* con diferentes IP. Todo esto se puede observar en la figura 4.3. Las velocidades máximas soportadas por cada tipo de enlace, varían en función del modelo. A modelos más modernos, como es lógico, mayores serán estas.

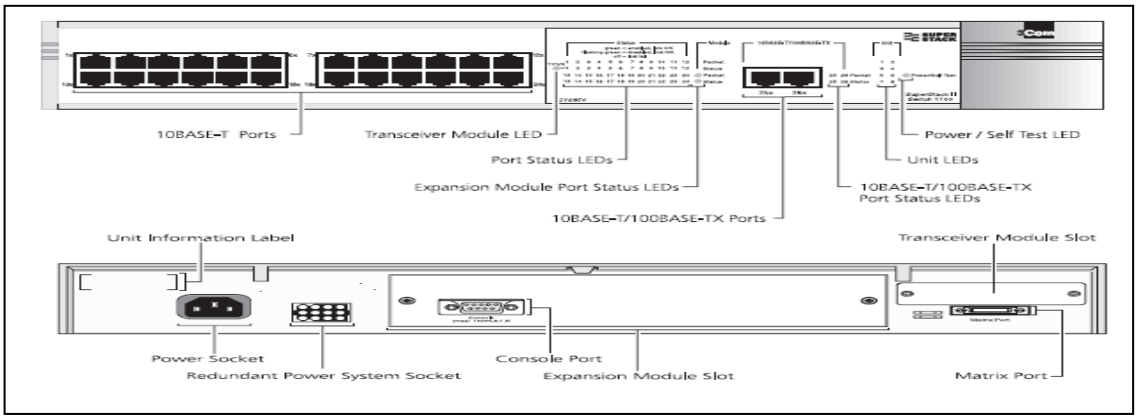


Figura 4.2 - Switch de 3COM.

Con la finalidad de una mejor comprensión de lo explicado, se pasa a comentar un par de ejemplos de delegaciones de RTVA. Uno muy sencillo es el de la delegación de Cádiz, mostrada en la figura 4.4. Se observan tres *switches* centrales, dos conectados entre sí en una sola pila mediante puertos *matrix* y estos conectados a otro, mediante un puerto 10BASE-T que soporta hasta 100Mbps. También pueden observarse las conexiones a la Red Corporativa mediante los *routers* de Telefónica, Rs 72 y Rs 73.

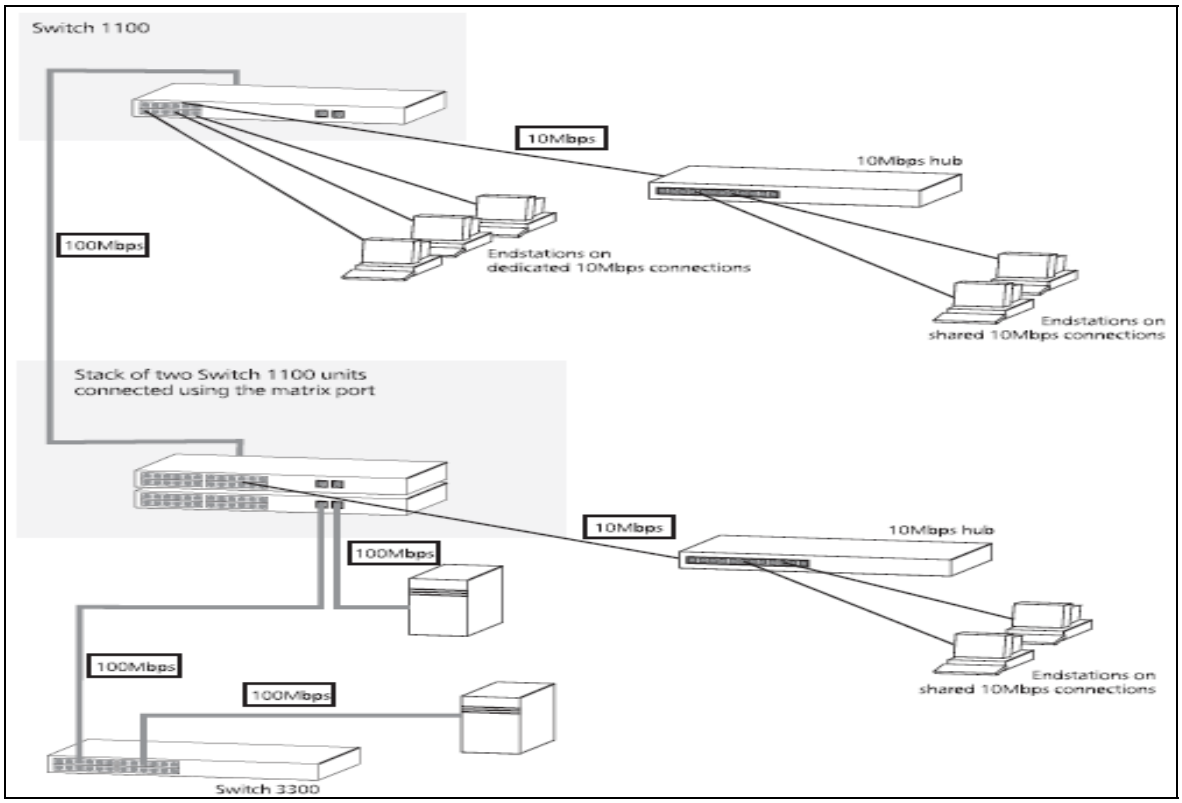


Figura 4.3 - Distintas conexiones que admite el switch.

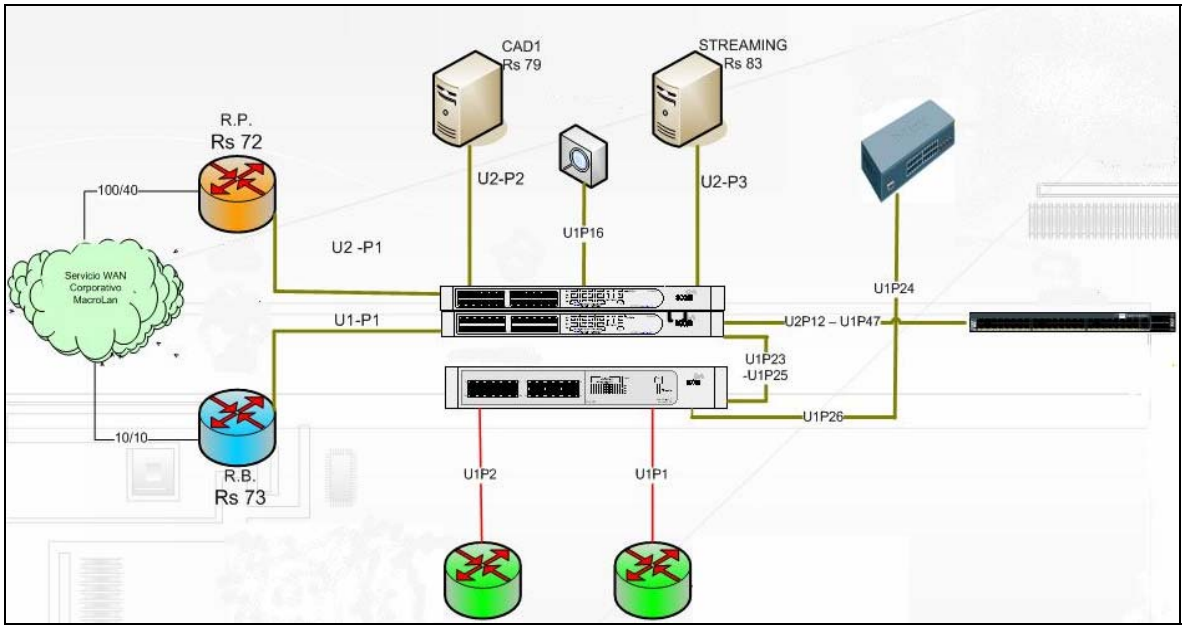


Figura 4.4 - Topología de Cádiz.

Una delegación más grande, se puede observar en la figura 4.5. Se trata de la delegación de Sevilla, pero a pesar de ser mayor en tamaño, la topología continúa siendo muy sencilla. También de usan los dos tipos de conexiones. Los *switches* que aparecen juntos, forman una pila, ya que están conectados mediante puertos *matrix*. Estas pilas están unidas a otras, mediante los puertos especiales 10BASE-T que soportan mayores velocidades. Además, también se observa la conexión de los *switches* a los *routers* de Telefónica RB y RP.

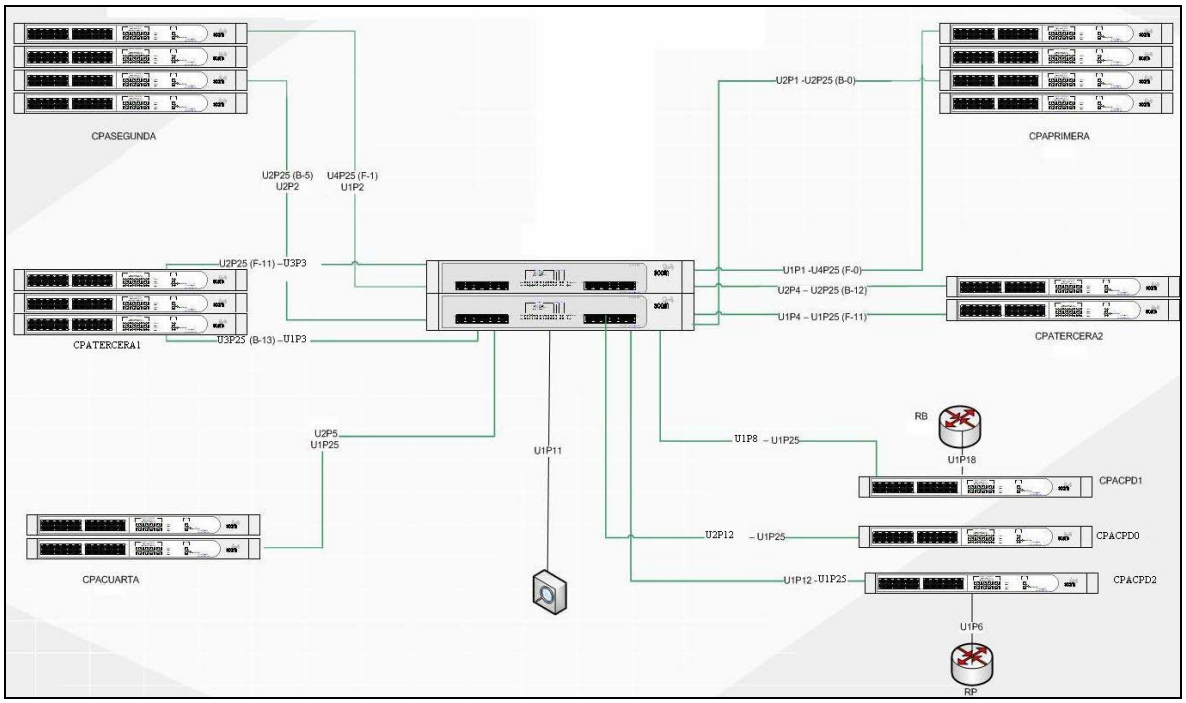


Figura 4.5 - Topología de Sevilla.

La elección de una conexión u otra, está relacionada con el número de puntos de red necesario, en función del número de terminales de la delegación y de la división que se quiera realizar del ancho de banda. También del número de IP disponible, ya que la conexión mediante puertos *matrix*, permite añadir puertos a una misma IP. En los ejemplos se puede observar que como en Sevilla hay más trabajadores, se necesita mayor ancho de banda y más puntos de red que en Cádiz y por tanto, más *switches*. Para disminuir el número de IP necesarias, se apilan los *switches* mediante puertos *matrix*. Bajo esta conexión, todos los *switches* de un mismo bloque tendrán la misma IP y se diferenciarán por el número de unidad. Por tanto, la pila de *switches* trabaja como un único dispositivo de muchos puertos. La unidad principal es la 1 y pertenece al switch que contiene los puertos *matrix*, a los que se conectan los demás. Para la enumeración de los puertos 10-BASE-T, se utilizan tres dígitos: el primero es el número de unidad y los dos últimos el de puerto. Por ejemplo, el puerto 15 de la unidad 1, será el 115 de todo el bloque. El 20 de la unidad 2, es decir, el switch conectado al puerto *matrix* 2, será el 220 de todo el bloque. Esto es bastante importante, porque al monitorizar el bloque con una sola IP, debemos saber como acceder a todos los puertos de los distintos *switches*.

Sea como sea la conexión, en todas las delegaciones existen dos *switches* distintos, cada uno conectado a un router de los de Telefónica. Estarán unidos entre ellos de alguna de las dos formas explicadas anteriormente.

El resto de elementos dependen de la delegación, pero van a ir conectados a los *switches* centrales, tal y como se puede observar en la topología de la delegación de Cádiz, figura 4.4. En caso de que exista un número elevado de ordenadores personales de los distintos trabajadores de la delegación, irán conectados a los *switches* que a su vez, van conectados al central. Por ejemplo, en la delegación de Sevilla, los *switches* conectados a los *routers* de Telefónica, son los de la planta sótano. Estos se comunican con la pila central mediante puertos 10-BASE-T. A los puertos de estos, se conectan los bloques de *switches* de las distintas plantas del pabellón. A los puertos 10-BASE-T mayoritarios de estos *switches*, van conectados los ordenadores de los trabajadores. Si se necesitan más puntos de red, se aumenta el número de *switches* en la pila. También cabe mencionar, que en esta delegación, los enlaces que unen a los bloques de *switches* de las distintas plantas con el central, están duplicados. Para evitar redundancia, se activa el *spaning tree*.

Por último, destacar que los *switches* se conectan entre sí mediante fibra óptica, cuando lo hacen a través de los puertos 10-BASE-T. Gracias a su color naranja, se puede afirmar

que se trata de una fibra multimodo. Y para la conexión de los distintos terminales a los *switches*, se utiliza cable de red RJ45.

4.1.2 - TIPOS DE DISPOSITIVOS DE LA RED Y FORMAS DE MONITORIZARLOS.

En un principio, lo que más le interesaba a la empresa, era la monitorización de los *switches* con sus correspondientes enlaces. Pues existe otro sistema de monitorización, basado en *OpenNMS*, que se encarga de otros dispositivos. Pero al descubrir la visualización de los distintos dispositivos, con sencillas pantallas mediante Zabbix, se decidió añadir algunos elementos más.

4.1.2.1 - Formas de recopilar información en Zabbix.

Zabbix permite la recopilar datos de los distintos dispositivos utilizando distintas opciones: mediante agentes o mediante chequeos [10].

4.1.2.1.1 - Mediante agentes

Se pueden utilizar dos tipos de agentes:

- Agente Zabbix**.** Necesita ser instalado en la máquina a monitorizar. Soporta las mismas plataformas que el gestor. Responde a las peticiones del gestor, enviándole los datos requeridos por este.
- Agente SNMP (*Simple Network Management Protocol)***.** Necesita estar instalado y activado en la máquina a monitorizar. Es un protocolo soportado por multitud de dispositivos.

Para descubrir las diferencias derivadas de utilizar un agente SNMP o uno Zabbix, se procedió a realizar un pequeño estudio. En primer lugar, se realizaron una serie de pruebas, que permitieron observar el comportamiento de los recursos del sistema, cuando sobre este funcionaba uno u otro. La recogida de datos se realizó mediante el propio gestor Zabbix, pues permite obtener información variada de los distintos recursos y servicios. Se experimentó con ambos agentes variando el sistema operativo, *Windows* y *Linux*, y el número de variables a monitorizar. Tras obtener las primeras conclusiones, se optó por realizar una breve captura de tramas, para comprobar la causa de la variación de algunos datos, como el número de operaciones de entrada y salida o el tráfico de la red.

Para la recogida de variables relacionadas con las características de la propia máquina, se utilizaron las variables de Zabbix referidas a las siguientes características:

- Memoria física utilizada por cada servicio.
- Memoria virtual utilizada por cada servicio.
- Número de operaciones de entrada y salida generadas por cada proceso.
- Tiempo medio de utilización de la CPU (*Central Processing Unit*)
- por cada proceso.
- Carga media de la CPU.
- Tráfico de red de entrada y salida.

Estudio en Windows	**Memoria física utilizada por SNMP**	**Memoria física utilizada por Zabbix**	**Memoria virtual utilizada por SNMP**	**Memoria virtual utilizada por Zabbix**	**Carga media de CPU**	**t de utilización de procesador por snmp**	**t de utilización de procesador por SNMP**
10 elementos	3.75MB	14.13MB	1.55MB	10.56MB	1.20	400ms	5026ms
20 elementos	3.75MB	14.13MB	1.55MB	10.56MB	1.20	500ms	5768ms

Tabla 4.1 - Consumo de recursos variando en número de elementos.

Tras la realización de las distintas pruebas, se concluyó que:

- El agente Zabbix ya dispone de multitud de variables definidas para monitorizar los recursos de un sistema. Para el agente SNMP, hay que buscar las adecuadas para un determinado dispositivo y además, añadirlas una a una. Aunque el número de estas en SNMP es mayor. Si se utiliza Zabbix se ahorrará tiempo.

- En general, el agente Zabbix consume mayores recursos que el agente SNMP, tal y como puede observarse en la tabla 4.1. Estos resultados hicieron sospechar, que ambos agentes no utilizaban el mismo protocolo de transporte y se optó por capturar unas cuantas tramas para comprobarlo. Efectivamente se comprobó, que SNMP trabaja sobre UDP (*User Datagram Protocol*) y en cambio Zabbix sobre TCP (*Transmission-Control-Protocol*). De ahí que el segundo consuma mayores recursos.

- El agente Zabbix no es soportado por todos los dispositivos, mientras que SNMP sí.

- Al duplicar el número de elementos a monitorizar, el consumo de memoria prácticamente no varía, pero se observa un aumento en: el tiempo de utilización de CPU

por los distintos servicios, el número de operaciones de entrada y salida y el tráfico de red. Este resultado parece obvio, pues cuanto más variables se monitoricen, más datos se intercambiarán entre agente y gestor.

- Sobre Linux se consumen menos recursos que sobre Windows, pero la diferencia es poco significativa.

- El consumo de recursos por ambos agentes es bastante bajo. En el caso del agente Zabbix, que es el que más consume, utiliza aproximadamente un 3% de la RAM (*Random Access Memory)* y del total de memoria virtual. En cuanto a la red, sólo utiliza aproximadamente 6 Kbytes del enlace descendente y 5.45 Kbytes del ascendente, para la monitorización de 30 variables cada 60 segundos. Como ejemplo en la figura 4.6, aparece una gráfica que muestra el tráfico de entrada de red, cuando se utiliza un agente Zabbix con 20, 10 y 30 elementos.

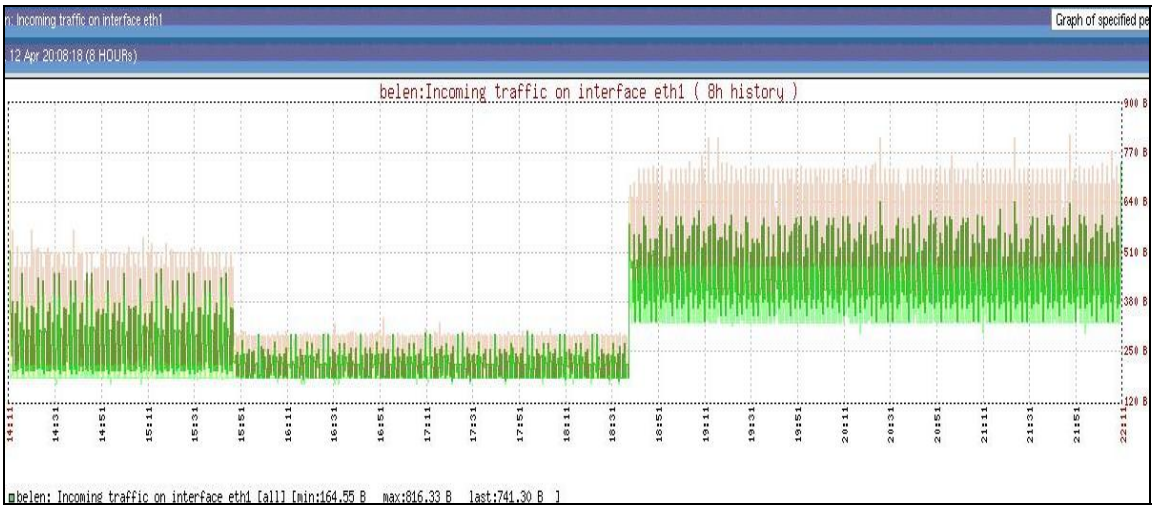


Figura 4.6 - Tráfico de red entrante para 20, 10 y 30 elementos.

Tras estas conclusiones, se optará por utilizar el agente Zabbix, en todos los casos que sea posible. Cuando el dispositivo no lo soporte, se utilizará SNMP con el fin de que el proyecto no se alargue demasiado. Además, existirán casos en los que no interesará monitorizar variables y para estos, podemos utilizar un tipo de chequeo de los permitidos por Zabbix.

4.1.2.1.2 - Mediante chequeo

Existen varios:

- Chequeo simple. Hace posibles las comprobaciones de determinados servicios, sin la necesidad de tener instalado ningún tipo de agente en la máquina. Se le pasa como parámetros, la IP y el puerto y devuelve 0,1 o 2 en función del estado del servicio.

El inconveniente de este tipo de chequeo, es que sólo puede acceder a un número limitado de servicios y no ofrece ningún otro tipo de información. Para saber los servicios que ofrece, puede consultarse el manual.

- Chequeo interno. Se puede utilizar únicamente en el gestor Zabbix. Sirve para monitorizar sus parámetros internos.

- Chequeo agregado. Realiza una serie de operaciones, sobre los datos recogidos de un grupo de dispositivos. El grupo debe estar definido y cada máquina debe poseer la variable que se utilizará, para realizar los cálculos estadísticos que permite este tipo de chequeo. Por ejemplo, supongamos que interesa conocer el espacio ocupado por el conjunto de servidores que conforman el grupo de *hosts* "MySQL Servers". Cada uno debe tener definido el *item* relacionado con este parámetro, que recoge la información de la máquina de forma individual. En este caso, se trataría de utilizar como *key* para el elemento,*"vfs.fs.size[/,total]"*. Este *item* sería monitorizado mediante un agente Zabbix. Supongamos que a este lo llamamos "espacio ocupado". Después, se definiría un *item* en el gestor Zabbix de tipo "chequeo agregado", que se añadiría a cada una de las máquinas pertenecientes al grupo "MySQL Servers". La sintaxis de su clave sería la siguiente:

  grpsum["MySQL Servers","vfs.fs.size[/,total]","last","0"]

  Esta función sumará todos los valores del espacio utilizado por cada uno de los miembros del grupo y mostrará el total.

- Chequeo externo. Se utiliza para que el gestor ejecute en una máquina un *script shell* o binario. La sintaxis de la clave del elemento será *script[parameters],* donde *script* se refiere al nombre y *parameters* a los parámetros en línea necesarios. Este tipo de chequeos puede llegar a colapsar el sistema, por eso, no es recomendable utilizarlo demasiado.

4.1.2.2 - Presentación de los distintos dispositivos a gestionar

Aunque la red de RTVA consta de multitud de dispositivos, este proyecto se dedicó a la gestión de los *switches* más importantes y de algún servidor. Aún así, en este apartado se propondrán los parámetros más importantes a monitorizar, en el caso de que en un futuro se decida gestionar también servidores Windows. Se pasan a comentar los parámetros que interesa conocer de cada uno y la solución escogida para recoger los valores de estos.

4.1.2.2.1 - *Switches* 3COM.

Son de gran importancia, pues el fallo de los centrales, puede hacer que falle la conexión en una planta de una delegación o en la delegación completa. Esta situación es bastante grave, teniendo en cuenta que RTVA es una empresa del sector de la información y por tanto, debe estar continuamente conectada con el exterior.

Los *switches* no permiten la instalación de un agente Zabbix y por tanto, se van a monitorizar mediante un agente SNMP. En principio, dichos agentes ya vienen instalados de fábrica. Se deben activar y para tener un acceso protegido, han de definirse las comunidades de SNMP. Para ello, se puede escribir en la línea de comandos:

system management snmp community

Después se continuará con las indicaciones de pantalla. Debido a que sólo soportan las versiones 1 y 2 de SNMP, no existen buenas herramientas que aseguren la fiabilidad en el uso de este protocolo.

En estos dispositivos, también debe activarse el envío de las *traps* a la IP del gestor Zabbix. Simplemente, después de acceder al switch, se escribe en la línea de comandos las siguientes órdenes:

system management snmp trap create

monitor

<ip_gestor_zabbix>

Se pueden mandar a más de una IP y para visualizar el conjunto de direcciones a las que se enviarán las traps, se escribe:

system management snmp trap summary

Para modificarlo:

system management snmp trap modify

Y por último, para borrar:

system management snmp trap delete

Un ejemplo de la interfaz de línea de comandos se muestra en la figura 4.7.

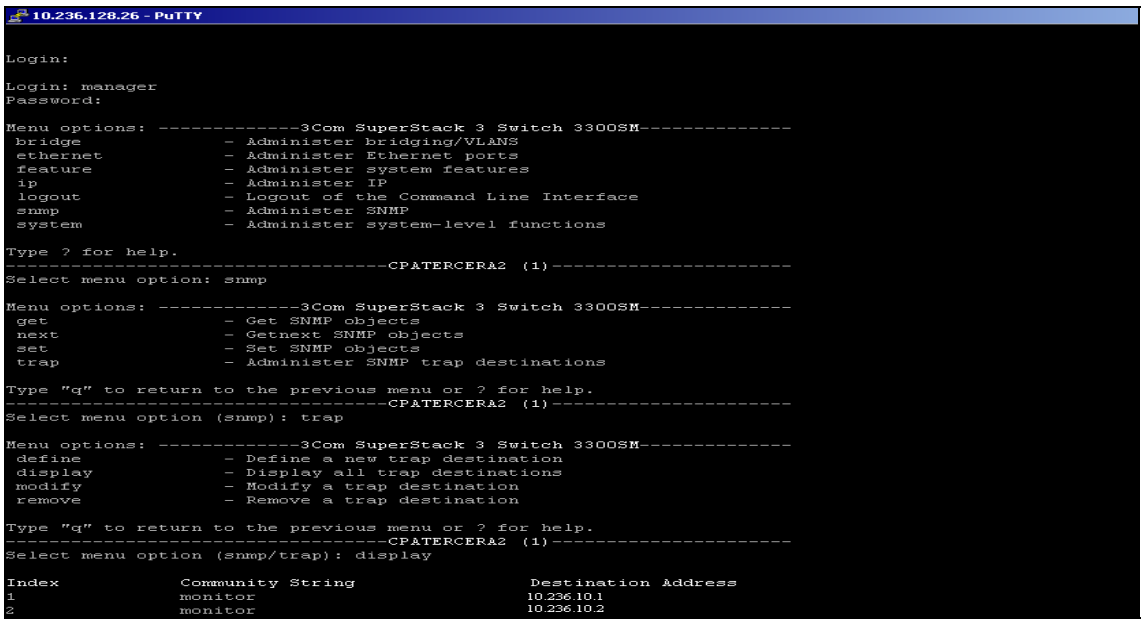


Figura 4.7 - Ejemplo de interfaz de comandos del *switch*.

Otro factor importante a tener en cuenta para la monitorización de estos dispositivos, es la necesidad de tener las MIB (*Management Information Base*) soportadas por los mismos. Esto fue una de las cosas que más tiempo duró. En el manual no vienen las MIB soportadas. Los *switches* son de la marca 3COM. Esta compañía ofrece distintas MIB para la gestión de sus dispositivos; sin embargo, los modelos existentes en RTVA no soportan ninguna y se tuvo que optar por utilizar MIB genéricas, definidas para la versión 1 de SNMP. Se usó como la "RFC1213.mib". Por otro lado, también cuentan con una serie de notificaciones, que enviarán automáticamente al gestor, cuando se de una determinada situación, definidas en RFC1215.

Para descubrir las MIB soportadas por el dispositivo, se utilizó MIB Browser. Se trata de un programa muy sencillo, que en su versión demo, permite cargar MIB y obtener los valores de las distintas variables soportadas. Simplemente, se deben configurar las comunidades y la IP en la que se encuentra el gestor. También permite descubrir los OID de los distintos elementos, lo cual como se verá más adelante, será crucial para configurar la monitorización de estos en el gestor Zabbix.

De cada switch, va a interesar monitorizar los puertos más importantes. Estos suelen ser los que interconectan distintos *switches*, el switch con los *routers* de Telefónica o el switch con algún servidor. Las variables SNMP, referidas a los parámetros que se han considerado más interesantes para un puerto son:

- ifOperStatus - Estado de servicio para cada una de las bocas.
- ifInDiscards - Número de tramas entrantes descartadas para cada una de las bocas.

- ifOutErrors - Número de octetos erróneos salientes para cada boca.
- ifInErrors - Número de tramas erróneas entrantes para cada una de las bocas.
- ifOutDiscards - Número de paquetes desechados.
- ifInOctets - Octetos entrantes para cada una de las bocas de los *switches*.
- ifOutOctets - Número de octetos salientes para cada boca.

Por otro lado, las variables referidas a las notificaciones que enviará el agente SNMP a Zabbix son:

- coldStart - Se envía cuando el dispositivo que contiene al agente SNMP se reinicia y su configuración puede haber sido alterada.
- authenticationFailure - Se envía cuando el agente recibe mensajes no identificados, como por ejemplo, mensajes SNMP con una comunidad errónea.
- warmStart - Envía esta trap cuando el dispositivo se reinicia sin que su configuración haya sido modificada.
- linkDown - Si se desactiva un enlace, el gestor recibirá una de este tipo.
- linkUp - Si se activa un enlace, el agente mandará esta trap al gestor.

En principio, no se van a añadir más variables, pues con el tiempo se irán ampliando en función de las necesidades surgidas.

4.1.2.2.2 - Servidores.

En realidad, sólo se monitorizaron dos servidores Linux, pero se realizaron una serie de pruebas en una máquina Windows. Se hizo así, por si en un futuro se deseara gestionar servidores de este tipo.

La propuesta es que todos se monitoricen mediante agentes Zabbix, a no ser que sólo se desee comprobar algunos servicios. En este caso, no sería necesario instalar nada.

No se debe olvidar, que los servidores no son más que computadores y por tanto, van a interesar los parámetros relacionados con su identificación y con las características principales hardware y software, que puedan interrumpir su correcto funcionamiento.

Los dispositivos poseen una serie de características, que no van a variar, si el administrador no las cambia. Es absurdo consumir recursos para obtener información que ya se conoce. Es el ejemplo del nombre del equipo, que si no lo cambia el administrador, siempre será el mismo. Por ello, se optó por recoger este tipo de información, directamente en el

inventario. Se consigue así, un ahorro de recursos. Estos datos están relacionados, sobre todo, con las características de la máquina. Son los identificados como información general del host. Más adelante se verá como introducirlos en el inventario. Mediante Zabbix no se encontraron todas las variables relacionadas con esta información. Por tanto, se optó por utilizar comandos de Linux [11], que permitían obtener las características directamente de la máquina remota. El resultado puede observarse en la figura 4.8. Luego se almacenó todo en el inventario. Toda esta información se considera permanente, ya que no va a variar durante un largo periodo de tiempo y si lo hace, será el administrador el que la cambiará.

Cuando las variables, referidas sobre todo al hardware, puedan recogerse mediante unidades naturales o mediante porcentajes, se optará por el segundo método. Se hace así, porque para definir los *triggers*, no se necesitará conocer los valores reales de los componentes hardware y por tanto, se podrán definir una vez en una plantilla y usarse para distintas máquinas.

Las variables se pasan a agrupar, en función del intervalo de actualización de las mismas. Estas serán comunes en todos los servidores, a excepción de los servicios levantados.

Figura 4.8 - Características del servidor Zabbix.

Información general del host.

Como se ha mencionado ya, será la recogida en el inventario:

- Nombre.
- Sistema operativo.
- Número de serie.
- Tipo de procesador.
- Memoria RAM.

Variables recogidas cada 30 minutos.

Se toma este intervalo para variables que modifican su valor de forma progresiva. Por tanto, los fallos que se puedan producir, serán predecibles y podrá optarse por una política proactiva. Por ejemplo, el espacio en disco normalmente, irá aumentando poco a poco, por tanto no hace falta que se esté comprobando cada minuto. A este conjunto pertenecen:

- Porcentaje de memoria swap libre.
- Bytes de memoria libre.
- Fecha del último reinicio del servidor.
- Tamaño de la memoria.
- Porcentaje de disco utilizado.

Variables recogidas cada minuto.

Se le asigna este intervalo, a aquellas variables cuyo valor pueda modificarse de forma abrupta y por tanto, los errores que se produzcan no serán predecibles.

- Fecha y hora local.
- Estado del servidor.
- Recogida de mensajes logs.
- Comprobación de procesos.
- Número de usuarios conectados al sistema.
- Carga de la CPU.
- Bytes de entrada en servidor.
- Bytes de salida en servidor.

Servidor Zabbix

En este servidor, también se utilizarán variables propias del chequeo interno, como son la cola del servidor y la recogida de mensajes logs. Todas serán recogidas cada minuto. Se pasan a mencionar:

- Cola de elementos en Zabbix.
- Recogida de mensajes logs de Zabbix.
- Variables del escenario web.
- Comprobación de procesos. Asegura que los siguientes procesos, esenciales para el correcto funcionamiento de Zabbix, se están ejecutando:

    - Servidor Zabbix.
    - Agente Zabbix.
    - inetd.
    - mysqld.
    - sshd.
    - syslogd.
    - snmptrapd.
    - snmpd.
    - http

- Disponibilidad de puertos 10050 y 10051.
- Última modificación del fichero /etc/passwd.

Servidor esafe

Este servidor contiene el software "eSafe Dual Engine Email Security" que es un sistema de seguridad de correos electrónicos. Permite proteger la red de virus, spam, códigos maliciosos y todo tipo de amenaza para la red, que pueda aparecer a través de un correo.

La empresa sólo estaba interesada en que se comprobara que algunos servicios estuvieran activos y que dicho servidor enviara las notificaciones al gestor Zabbix. Por este motivo, se utilizó el chequeo simple. Los servicios que deben estar activos son: SMTP (*Simple Mail Transfer Protocol)*, FTP (*File Transfer Protocol)*, ICMP (*Internet Control Message Protocol*), IMAP (*Internet Message Access Protocol*), NNTP (*Network News*

Transport Protocol), POP (*Post Office Protocol*), SSH (*Secure SHell*) Y HTTP (*HyperText Transfer Protocol*). Se comprobarán cada minuto.

Servidores Windows

Aunque en realidad no se monitorizó ninguna máquina con dicho sistema operativo, se realizaron pruebas por si en un futuro se decidiera gestionar algún equipo. Además este sistema operativo soporta mediante agente Zabbix un mayor número de variables, que el sistema operativo Linux.

Además de todos los parámetros comentados anteriormente, puede ser interesante conocer información específica del consumo de recursos por un determinado proceso. Esto se utilizó, por ejemplo, para el estudio de las diferencias existentes entre el agente Zabbix y el SNMP.

Los procesos que se comprobarán en este tipo de máquinas serán:

- El de agente Zabbix.
- El de servicio de red.
- El de servicio antivirus.
- El de servicio ofrecido por el servidor.

Seguramente existirán otros servicios importantes en la máquina a gestionar, que se deberán agregar en su momento.

4.2 - INSTALACIÓN

En este apartado, se comentarán las necesidades hardware y software de la plataforma Zabbix y a continuación, se explicarán los procesos de instalación de gestor y agentes. Además, se comentarán una serie de operaciones añadidas, como por ejemplo, el reinicio de los procesos con la máquina o la creación de distintos *scripts*.

4.2.1 - REQUISITOS PARA LA INSTALACIÓN DE ZABBIX

Será necesario disponer de un hardware con unas características determinadas y de un SO (*Sistema Operativo*) que lo soporte. A continuación, se realiza un análisis de los requisitos hardware y software.

4.2.1.1 - Requisitos Hardware

El hardware adecuado dependerá del número de variables y de equipos a monitorizar, así como del intervalo de actualización y del tiempo de permanencia de datos en memoria.

Como todavía no se ha mencionado nada relacionado con la configuración del gestor, se van a elegir unas determinadas características para la plataforma Zabbix y al final del proyecto, se determinará si son suficientes o necesitan ampliarse.

Para una monitorización centralizada, (también admite la opción de monitorización distribuida), y en base a las exigencias mencionadas en el manual de Zabbix, las características elegidas para nuestro sistema son:

- **CPU**

  Dependiendo del número de equipos y parámetros a monitorizar y de la base de datos elegida, se necesitará un procesador más o menos rápido. En este caso, la plataforma escogida cuenta con dos procesadores de tipo *Intel(R) Xeon(R)* a 2.4 GHz.

- **MEMORIA FÍSICA**

  La mínima exigida es de 128 MB y lo aconsejable es que sea mayor para conseguir mayor rapidez. Por tanto, la plataforma contará con una *RAM* de tamaño 1 Gbyte.

- **DISCO DURO**

  La elección de su tamaño depende básicamente del tamaño elegido para la base de datos. Por eso, se deberá tener en cuenta el número de parámetros a monitorizar y el tiempo que van a permanecer dichos datos almacenados. En este caso, el valor escogido es de 80 Gbytes.

4.2.1.2 - Requisitos software

Zabbix es soportado por las siguientes plataformas:

- AIX
- FreeBSD
- HP-UX
- Linux
- Mac OS/X
- OpenBSD
- SCO Open Server
- Solaris

El propio manual recomienda el uso de las distribuciones sobre Linux mencionadas a continuación, asegurando que ofrecen mejor soporte:

- Debian Linux

- RedHat Linux

- SuSE Linux

- Ubuntu Linux

La seleccionada para este proyecto, será Linux y más concretamente, la distribución Debian GNU versión 2.6.18.

Después de esta elección, se debe tener en cuenta que Zabbix, para su correcto funcionamiento, necesitará la instalación de Apache, una base de datos de entre las soportadas y el módulo PHP. Como base de datos, se puede escoger entre MySQL, Oracle, PostgreSQL o SQLite. Se opta por utilizar la primera, ya que ofrece un mayor número de funcionalidades sobre Zabbix que las demás.

Entorno	Distribución	Hardware	BBDD	Hosts
Small	Debian GNU/Linux	P2 350MHz 256MB	MySQL MyISAM	20
Medium	Debian GNU/Linux 64 bit	AMD Athlon 3200+ 2GB	MySQL InnoDB	500
Large	Debian GNU/Linux 64 bit	Intel Dual Core 6400 4GB RAID	MySQL InnoDB or PostgreSQL	>1000
Very Large	RedHat Enterprise	Intel Xeon 2 CPU 8GB RAID	MySQL InnoDB or PostgreSQL	>10000

Tabla 4.2 - Recursos en función del número de equipos.

Las versiones mínimas de los módulos mencionados que funcionan bien con el gestor Zabbix son:

- Apache Versión 1.3.12 o posterior.
- PHP Versión 4.3 o posterior con módulos php-gd y php-bcmath.
- MySQL Versión 3.22 o posterior con módulo php-mysql.

Por último, la plataforma también debe disponer de las utilidades make, make-dev, gcc, de la aplicación snmp y de las siguientes librerías:

- libc6-dev
- libmysql++-dev
- libcurl
- libpcap0.8
- libsnmp9
- libsnmp-perl
- libsnmp-dev

- libsnmp-base
- libapache2-mod-php5

Un ejemplo de requerimientos para un entorno en producción, en función de algunos parámetros, puede visualizarse en la tabla 4.2.

4.2.2 - INSTALACIÓN DEL GESTOR ZABBIX

Antes de proceder a la instalación de Zabbix, la plataforma debe contar con Apache, MySQL y PHP funcionando correctamente, tal y como se ha comentado. Para facilitar este paso a usuarios con conocimientos básicos de Linux, se aconseja utilizar el gestor de paquetes Synaptic.

Puede comprobarse, que tanto Apache como MySQL se ejecutan correctamente, asegurándose de que aparecen sus procesos demonios, *apache2* o *httpd* para Apache y *mysqld* para la base de datos. Para ello, se teclea la siguiente orden en la línea de comandos [12]:

netstat -tulpen

Entonces se procede a instalar el gestor. Los pasos necesarios para la correcta instalación de Zabbix, son los siguientes:

- Descarga de los ficheros con el código fuente de la página principal: *www.zabbix.com*. Este paquete incluye gestor y agente para la monitorización del propio Zabbix.
- Creación de cuenta "zabbix". Deben escribirse los siguientes comandos:

  useradd zabbix

  passwd zabbix

  Para continuar, se pasa al usuario "root":

  su - <*INTRO*> e introducimos contraseña.

- Descompresión del código fuente:

  tar xzvf zabbix-1.4.tar.gz

- Creación de la base de datos. Existen distintos *scripts*: uno para la creación de la base de datos, uno para cada tipo soportado (soporta MySQL, Oracle, PostgreSQL, SQLite); y otro, para insertar una configuración por defecto. Para realizar la creación de dicha base de datos, se debe acceder con un usuario que contenga los permisos adecuados. Se

siguen distintas pautas, según el tipo de base de datos. En este caso, se va a utilizar MySQL y los pasos a realizar son:

shell> mysql -u<username> -p<INTRO>

Deberá introducir la contraseña y después pulsar <INTRO>. Ya dentro de MySQL, se prosigue con la creación de la base de datos para Zabbix:

mysql> create database zabbix;

mysql> quit;

Dentro del directorio zabbix1.4 (es el número de versión instalada, por tanto puede variar):

shell> cd schema

shell> cat mysql.sql | mysql -u<username> -p zabbix

shell> cd ../data

shell> cat data.sql | mysql -u<username> -p zabbix

shell> cat images_mysql.sql | mysql -u<username> -p zabbix

Si hemos añadido contraseña, será solicitada cada vez que se acceda de esta forma a la base de datos de Zabbix.

- Configuración y compilación del código fuente.

  shell> ./configure --enable-server --with-mysql --with-net-snmp –-with-libcurl

- Instalación.

  shell> make install

  Los ficheros binarios se instalarán por defecto en */usr/local/bin* y las librerías en */usr/local/lib*. Si se desea otra ubicación se debe usar "-prefix".

- Configuración de /etc/services. Añado las siguientes líneas a dicho fichero:

  zabbix_agent 10050/tcp

  zabbix_trap 10051/tcp

- Copia de ficheros de configuración. Se crea el directorio *zabbix* y se almacenan en este, todos los ficheros de configuración, como por ejemplo, el del servidor *zabbix_server.conf*:

  mkdir /etc/zabbix

cp /zabbix-1.4/misc/conf/ /etc/zabbix/*

- Configuración del gestor Zabbix. Se abre el fichero de configuración y se modifica lo necesario:

vim /etc/zabbix/zabbix_server.conf

Se debe comprobar que "DBName=" contiene el nombre de la base de datos. En nuestro caso es zabbix.

DBName=zabbix

Se guardan los cambios y se sale. Si se desea modificar la contraseña para el acceso a Zabbix mediante línea de comandos, debe hacerse en este fichero.

- Interfaz web. Se crea un directorio de nombre *zabbix* en */var/www* y se copia en su interior el contenido de */usr/local/zabbix-1.4/frontends/php*. Se trata de los ficheros PHP que permiten el uso de la interfaz web de Zabbix.

mkdir /var/www/zabbix

cp –a /zabbix-1.4/frontends/php/ /var/www/zabbix/*

- Inicio del servidor. Simplemente debe posicionarse en el directorio que contiene al ejecutable y teclear:

zabbix_server

- Acceso a Zabbix por la interfaz web. Se abre el navegador y se escribe "http://localhost/zabbix". Aparecerá algo parecido a lo mostrado en la figura 4.9.

Se prosigue con la instalación del *frontend*, pulsando sobre *Next*. Entonces se deberá aceptar la licencia GPL y continuará chequeando los prerrequisitos, los ficheros de configuración y la conexión con la base de datos. Se puede visualizar en la figura 4.10, los fallos que surgieron en esta instalación. Para arreglarlos, debe localizarse el fichero “php.ini” y cambiar los parámetros que sean erróneos. El tiempo máximo de ejecución se cambió a 300 y la zona horaria a la de Madrid. Ambos parámetros se modifican en php.ini:

max_execution_time = 300

date.timezone = Europe/Madrid

Tras estos pasos, si todo se ha realizado de forma adecuada, al pulsar *Retry*, aparecerán todos los prerrequisitos correctos. Se proseguirá la instalación pulsando *Next* y para finalizar se pulsa el botón "Finish".

A partir de este momento, ya se puede acceder a Zabbix a través del interfaz web. Simplemente se abre un navegador y en la barra de navegación, se pone la dirección IP de la máquina donde se encuentra Zabbix*:*

http://10.236.137.58/zabbix.

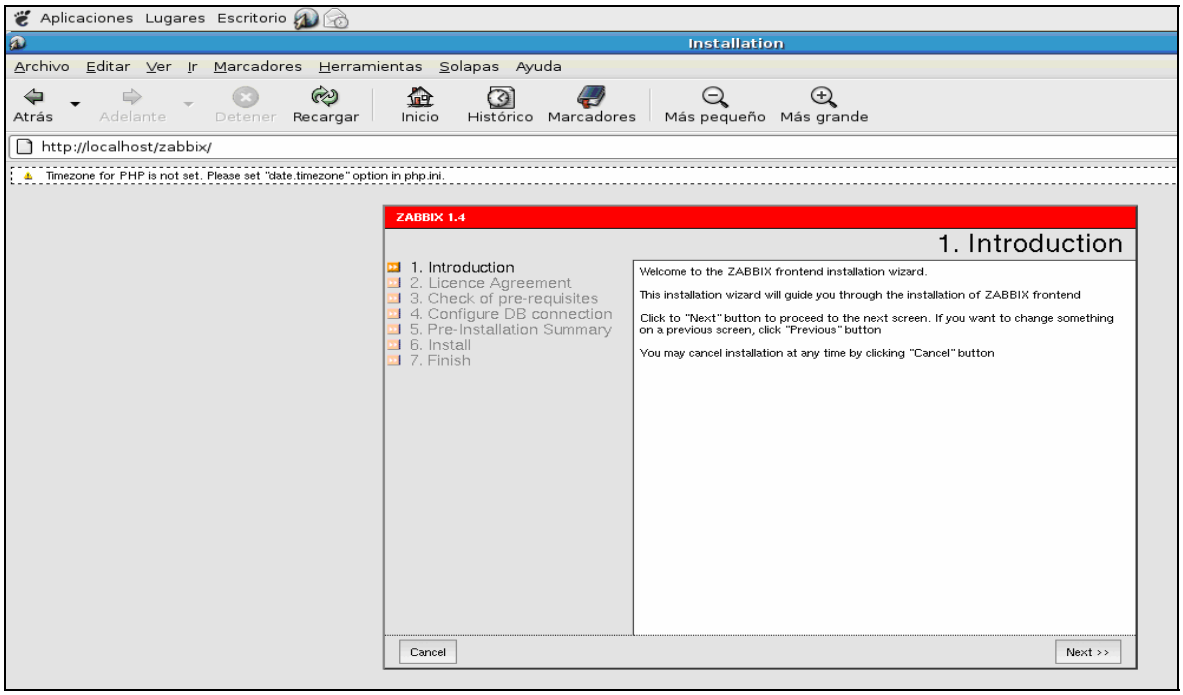


Figura 4.9 - Pantalla de inicio de Zabbix tras instalación.

Aparecerá una pantalla como la mostrada en la figura 4.11. Inicialmente el usuario es "admin" y no tiene contraseña.

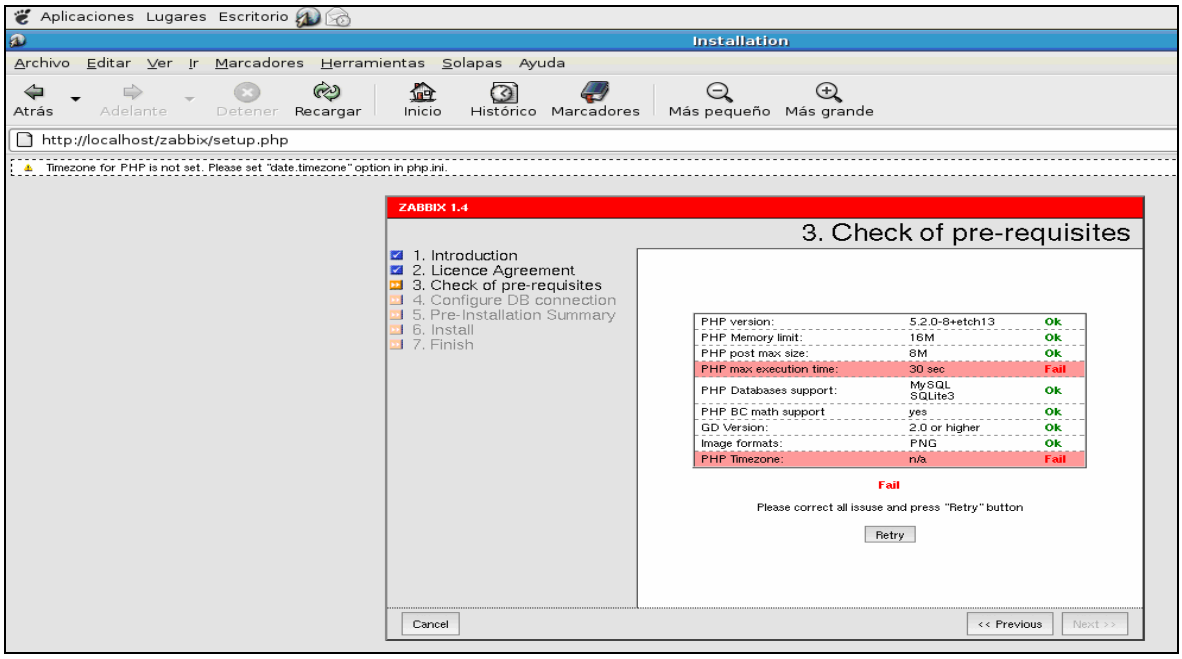


Figura 4.10 - Chequeo de requisitos en instalación de frontend.

Para cambiar la contraseña, se pulsa “Profile” en la parte superior derecha de la ventana. Cabe recordar que esta contraseña sólo servirá para el interfaz web.

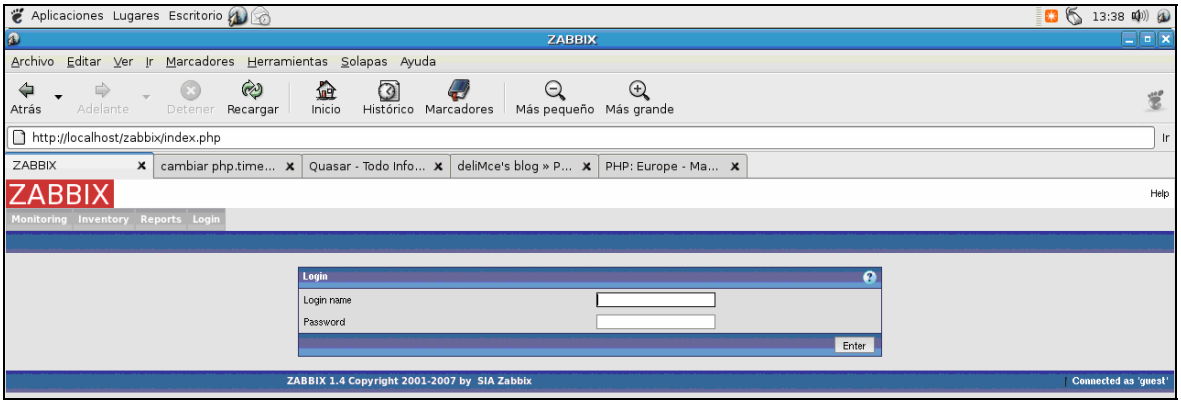


Figura 4.11 - Pantalla de inicio de la interfaz web de Zabbix.

4.2.3 - INSTALACIÓN DE PAQUETE PARA LA RECOGIDA DE TRAPS

Para la recogida de traps, notificaciones SNMP, se debe instalar el paquete "snmptrapd" de "net-snmp". Una vez realizada la instalación, se modifica el fichero de configuración *snmptrapd.conf*. Para encontrar su ubicación, se puede utilizar el comando "*find*". Entonces se copia en el directorio */etc/snmp/* y se edita añadiéndole la siguiente línea:

traphandle default /bin/bash /usr/local/zabbix-1.4/bin/snmptrap.sh

```
ZABBIX_SERVER="localhost";
ZABBIX_PORT="10051";

ZABBIX_SENDER="/usr/local/bin/zabbix_sender";

KEY="snmptraps";
HOST="traps";

# END OF CONFIGURATION
read hostname
read ip
read uptime
read oid
read address
read community
read enterprise

oid=`echo $oid|cut -f2 -d' '`
address=`echo $address|cut -f2 -d' '`
community=`echo $community|cut -f2 -d' '`
enterprise=`echo $enterprise|cut -f2 -d' '`

oid=`echo $oid|cut -f11 -d'.'`
community=`echo $community|cut -f2 -d""`

str="$hostname $address $community $enterprise $oid"

$ZABBIX_SENDER $ZABBIX_SERVER $ZABBIX_PORT $HOST $KEY
"$str"

echo $ZABBIX_SENDER $ZABBIX_SERVER $ZABBIX_PORT
$HOST $KEY \"$str\" > /tmp/testtrap.txt
```

Tabla 4.3 – *Script* para recogida de traps en testtrap.txt.

Tras esto, se modifica el *script* con los parámetros adecuados, tal y como se muestra en la tabla 4.3 y luego, se copia en el directorio expresado en su fichero de configuración.

Para el correcto funcionamiento de la recogida de *traps*, el fichero testtrap.txt, que almacenará la información recogida de los *traps*, debe existir en el directorio *tmp*. Si no existe, debe crearse.

Aún no se puede poner al gestor Zabbix a recoger las traps, pues es necesaria la definición de un *host* especial en dicho gestor. Esto se realizará mediante la interfaz web y se verá más adelante en el apartado de *Configuración*.

4.2.4 - INSTALACIÓN DEL AGENTE EN LA MÁQUINA GESTOR

La finalidad de este agente es comprobar el correcto funcionamiento de la máquina que contiene al gestor Zabbix. Para su instalación, se procede tal y como expresa el manual. Una vez situados en el directorio */usr/local/zabbix-1.4*, se teclea:

./configure --enable-agent

Chequeará el sistema para comprobar que existen todas las librerías y aplicaciones necesarias y compilará el agente a instalar. Para construirlo, simplemente se escribe:

make

Como anteriormente ya se copiaron todos los ficheros de configuración en */etc/zabbix*, se abre el del agente y se modifican los parámetros necesarios para su correcto funcionamiento.

vim /etc/zabbix/zabbix_agentd.conf

En este caso, sólo se van a activar las líneas que permiten la monitorización de la base de datos, tabla 4.4. Para los demás parámetros, se mantendrán los valores que vienen por defecto.

En el manual, se aconseja utilizar el agente Zabbix mediante el ejecutable *zabbix_agentd*. Para ello, deben realizarse los siguientes pasos:

- Configurar */etc/inetd.conf*. A este fichero se le añade la siguiente línea:

  zabbix_agent stream tcp nowait.3600 zabbix /usr/local/bin/zabbix_agent

```
####### USER-DEFINED MONITORED PARAMETERS #######
# Format: UserParameter=<key>,<shell command>
# Note that shell command must not return empty string or EOL only
#UserParameter=system.test,who|wc -l
### Set of parameter for monitoring MySQL server (v3.23.42 and
later)
### Change -u<username> and add -p<password> if required
UserParameter=mysql.ping,mysqladmin -uroot ping|grep alive|wc -l
UserParameter=mysql.uptime,mysqladmin -uroot status|cut -f2 -
d":"|cut -f1 -d"T"
UserParameter=mysql.threads,mysqladmin -uroot status|cut -f3 -
d":"|cut -f1 -d"Q"
UserParameter=mysql.questions,mysqladmin -uroot status|cut -f4 -
d":"|cut -f1 -d"S"
UserParameter=mysql.slowqueries,mysqladmin -uroot status|cut -f5 -
d":"|cut -f1 -d"O"
```

Tabla 4.4 - Líneas correspondientes a la monitorización de la base de datos.

El ejecutable debe estar en esa ruta de acceso.

- Se sale del fichero y se reinicia el servicio inetd:

  killall -HUP inetd

Si la instalación ha sido correcta, en el interfaz web de Zabbix, debe aparecer una máquina llamada *zabbix server* con una serie de parámetros monitorizados.

4.2.5 - INSTALACIÓN DEL AGENTE ZABBIX EN LINUX

Se deben realizar los siguientes pasos:

1. Descompresión del fichero. Tras haber bajado el fichero fuente correspondiente al agente, zabbix*_agents_1.4.4.linux2_4.i386.tar.gz*, se descomprime. Debe tenerse en cuenta que la versión del agente debe ser igual o anterior a la del gestor, pues puede no funcionar en otro caso.

   tar xzvf zabbix_agents_1.4.4.linux2_4.i386.tar.gz

2. Creación de usuario zabbix. Para crearlo, se accede al sistema como usuario root y se añade:

   adduser zabbix

Si además de desea que contenga una clave, por ejemplo *zabbix*, se escribe:

   passwd zabbix

3. Creación del directorio de trabajo del usuario en */home*:

   mkdir /home/zabbix

4. Copia de los archivos de inicio. Se realiza la copia desde */etc/skel*, que contiene principalmente archivos de configuración por defecto:

cp /etc/skel/. /home/zabbix*

5. Se hace que tome posesión de su carpeta:

chown zabbix.zabbix -R /home/zabbix

6. Copia del ejecutable. Se cambia al usuario *zabbix*:

su – zabbix

Se crea la carpeta */home/zabbix/bin* y se copia el ejecutable que se ha extraído anteriormente:

mkdir /home/zabbix/bin

cp /home/usuario/Desktop/zabbix_agentd /home/zabbix/bin

7. Configuración. Hay que modificar la IP del servidor y el nombre de la máquina local. Se crea el directorio */etc/zabbix* y se copia en este el fichero creado:

mkdir /etc/zabbix

cp zabbix_agentd.conf /etc/zabbix

8. Ejecución. Debe accederse como usuario Zabbix:

cd /home/zabbix/bin

zabbix_agentd

Comentar que el usuario *zabbix* debe poseer los permisos necesarios para poder ejecutar al agente.

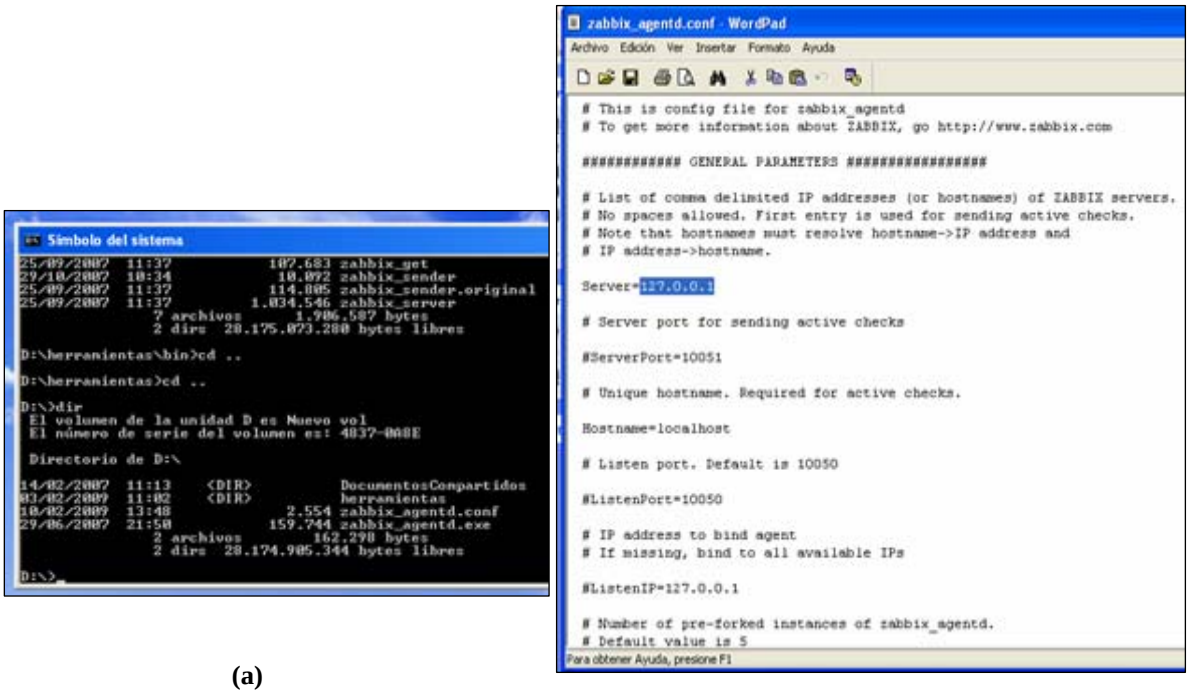
Si se utiliza el comando “netstat -tulpen” se puede comprobar que el servicio está levantado.

Tras realizar estos pasos, se debe definir la máquina que contiene al agente en el gestor Zabbix. Se puede realizar mediante el DNS o simplemente con la IP de la máquina del agente.

4.2.6 - INSTALACIÓN DEL AGENTE ZABBIX EN WINDOWS

Consiste en seguir los siguientes pasos:

1. Abrir una ventana de comandos MS-DOS: Inicio | Ejecutar... y teclear: *cmd*



(a) **(b)**

Figura 4.12 - Pasos de la instalación del agente: (a) Copia de ficheros en D; (b) Configuración.

2. Ubicar el *prompt* de la ventana de comandos, en el directorio donde están los ejecutables:

cd d:

3. Editar los parámetros necesarios en el fichero *zabbix_agentd.conf* tales como el nombre del equipo, que será registrado en el gestor Zabbix y la IP correspondiente.

Hostname=bacosta

Server=192.168.137.58

Estos pasos pueden observarse en las figura 4.12.

4. Instalar el agente como un servicio de Windows.

zabbix_agentd.exe -i -c zabbix_agentd.conf

Si todo marcha correctamente, aparecerá el siguiente párrafo:

zabbix_agentd.exe [3044]: ZABBIX Agent service created successfully.
zabbix_agentd.exe [3044]: Event source "ZABBIX Agent" installed successfully.

5. Arrancar el servicio *ZABBIX Agent* creado en el paso anterior.

zabbix_agentd.exe -s -c zabbix_agentd.conf

Si todo transcurre con normalidad, el sistema indicará que se ha arrancado el servicio:

zabbix_agentd.exe [2324]: ZABBIX Agent service started successfully.

Aún así, se debe comprobar que el servicio está "Iniciado" y que el visor de sucesos del sistema indica "El servicio ZABBIX Agent entró en estado Activo". Cada vez que reinicie la máquina, reiniciará el proceso correspondiente al agente Zabbix.

Si en un futuro se desea desinstalar el agente, bastará con teclear:

zabbix_agentd.exe -d -c zabbix_agentd.conf

4.2.7 - INSTALACIÓN DEL AGENTE SNMP EN LINUX

Para este proyecto la plataforma utilizada será Debian y los pasos a seguir para la correcta instalación de este tipo de agente son:

1.- Actualización de repositorios.

apt-get update

2.-Instalación de snmpd.

apt-get install snmpd

3.- Configuración. Tras completar la instalación, debe configurarse SNMP en el directorio */etc/snmp*.

vim /etc/snmp/snmpd.conf

Lamentablemente, la facilidad de instalación del paquete SNMP, choca de bruces con la configuración, ya que el fichero generado es escaso y exige una inclusión de muchas variables especializadas. Por tanto, se va a realizar una configuración muy sencilla, que incluye a las variables más importantes de la configuración SNMP para una máquina Debian.

El fichero contendrá como mínimo las líneas citadas en la tabla 4.5. Tras esta modificación, se guarda el fichero en el directorio indicado y se reinicia el servicio snmp:

/etc/init.d/snmpd restart

Si SNMP funciona correctamente, al teclear la siguiente línea, aparecerá en pantalla una lista de variables:

snmpwalk -v 1 -c public localhost system

```
####
# First, map the community name (COMMUNITY) into a security name
# (local and mynetwork, depending on where the request is coming
# from):
# sec.name source community
#com2sec paranoid default public
com2sec readonly default public
#com2sec readwrite default private
####
# Second, map the security names into group names:
# sec.model sec.name
group MyROSystem v1 paranoid
group MyROSystem v2c paranoid
group MyROSystem usm paranoid
group MyROGroup v1 readonly
group MyROGroup v2c readonly
group MyROGroup usm readonly
group MyRWGroup v1 readwrite
group MyRWGroup v2c readwrite
group MyRWGroup usm readwrite
####
# Third, create a view for us to let the groups have rights to:
####
# Third, create a view for us to let the groups have rights to:
# incl/excl subtree mask
view all included .1 80
view system included .iso.org.dod.internet.mgmt.mib-2.system
####
# Finally, grant the 2 groups access to the 1 view with different
# write permissions:
# context sec.model sec.level match read write notif
access MyROSystem "" any noauth exact system none none
access MyROGroup "" any noauth exact all none none
access MyRWGroup "" any noauth exact all all none
# -----------------------------------------------------------------------
###############################################################################
#
# System contact information
#
# It is also possible to set the sysContact and sysLocation system
# variables through the snmpd.conf file. **PLEASE NOTE** that setting
# the value of these objects here makes these objects READ-ONLY
# (regardless of any access control settings). Any attempt to set the
# value of an object whose value is given here will fail with an error
# status of notWritable.
syslocation Unknown (configure /etc/snmp/snmpd.local.conf)
syscontact Root <root@localhost> (configure /etc/snmp/snmpd.local.conf)
```

Tabla 4.5 - Fichero de configuración de agente SNMP.

4.2.8 - INSTALACIÓN DE AGENTE SNMP EN WINDOWS

El citado aquí es para una plataforma con Windows XP y aunque variarán algunas opciones según la versión, el procedimiento siempre es más o menos el mismo:

1. Instalación de las herramientas de monitorización de Windows. Para ello, se sigue la siguiente ruta:

   Inicio-->Panel de control-->Agregar o quitar programas->Agregar o quitar componentes de Windows.

Se selecciona la opción de *Managemet and monitoring Tools*. Se pueden comprobar las herramientas que añade dicha opción pulsando en *Detalles.* Aparecen dos

componentes, *Simple Network Management Protocol* y *WMI SNMP Provides.* Se acepta y se instalan siguiendo las indicaciones de pantalla.

2. Configuración del agente SNMP. Se procede de la siguiente forma:

Panel de Control--> Herramientas administrativas-->Servicios.

Dentro de este, buscamos *SNMP Service* y hacemos click. Aparecen las distintas opciones para SNMP. Se pincha en la pestaña del agente, *Agent*, y se modifican los campos necesarios. De igual forma se hará para las pestañas *Traps* y *Security*. Pueden observarse las pantallas que aparecen para cada opción en la figura 4.13. Se pasa a explicar las distintas opciones de cada una:

1. Pestaña *Agent*. Ofrece distintas opciones:

- Contacto.
- Ubicación.
- Servicios. Se distinguen distintas opciones a seleccionar:

  - 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.

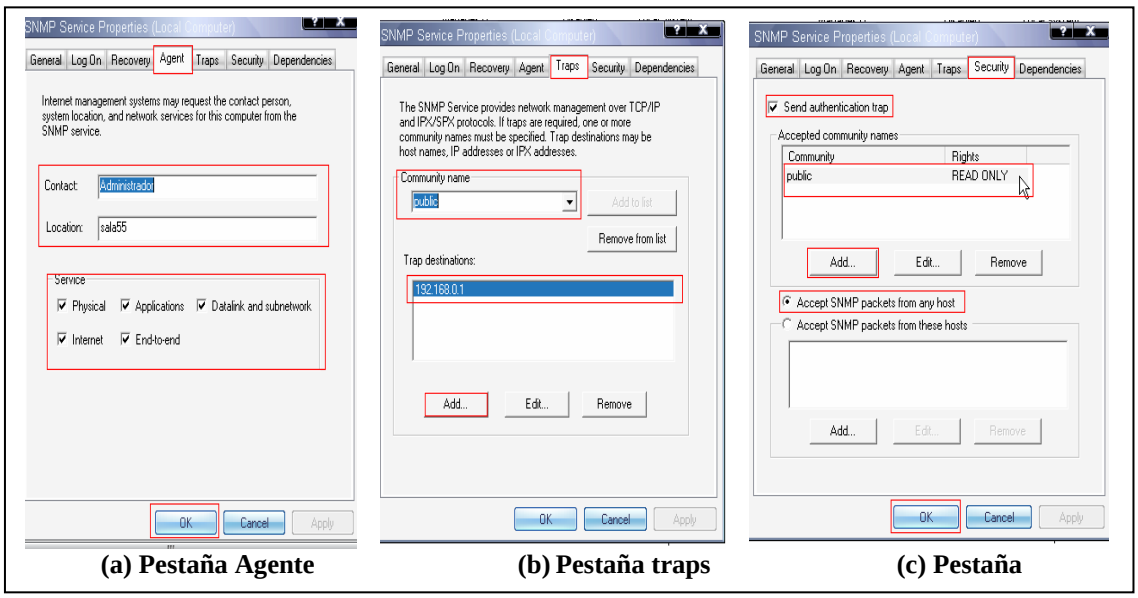
  - Internet: especifica si este equipo actúa como una puerta de enlace IP.

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

2. Pestaña *Traps*. Simplemente se introduce la comunidad SNMP y la IP del gestor que recibirá las *traps*.

3. Pestaña *Security*. Se activa el envío de *trap* de autenticación. Después se agrega el nombre de la comunidad aceptada y se añade el host desde el que se recibirán paquetes SNMP. En este caso será el que contiene al gestor Zabbix.

Una vez realizados estos pasos, se reinicia el servicio SNMP con el botón derecho del ratón. Tras esto, el agente SNMP en este equipo ya estará listo y ejecutándose. Para comprobarlo, basta con buscar el proceso utilizando el administrador del sistema.



(a) Pestaña Agente **(b) Pestaña traps** **(c) Pestaña**

Figura 4.13 - Opciones para las configuraciones de agente, traps y seguridad.

4.2.9 - REINICIO DE SERVICIOS

Es importante que cuando la máquina que contiene a Zabbix reinicie, se levanten automáticamente todos los servicios para su correcto funcionamiento. Tras la instalación, se comprobó que los que no lo hacían eran los relacionados con el agente y el gestor Zabbix. Asimismo, se programó la realización de una copia de la base de datos diariamente.

La carpeta de instalación, en este caso situada en */usr/local/*, contiene una carpeta con los *scripts* que inicializan estos servicios, *zabbix-1.4.1/misc/init.d/debian*.

Se copian dichos ficheros en */etc/init.d*. Se modifican las rutas relacionadas con los ejecutables y los ficheros de configuración.

Después se teclea *rcconf*. Aparecerá una pantalla como la de la figura 4.14.

Se escogen los servicios a reiniciar y se sale.

La próxima vez que se reinicie, también lo harán los servicios de Zabbix.

Para el servicio de recogida de traps, no viene el *script* de reinicio del servicio. Así que se realizó y se copió en */etc/init.d*.

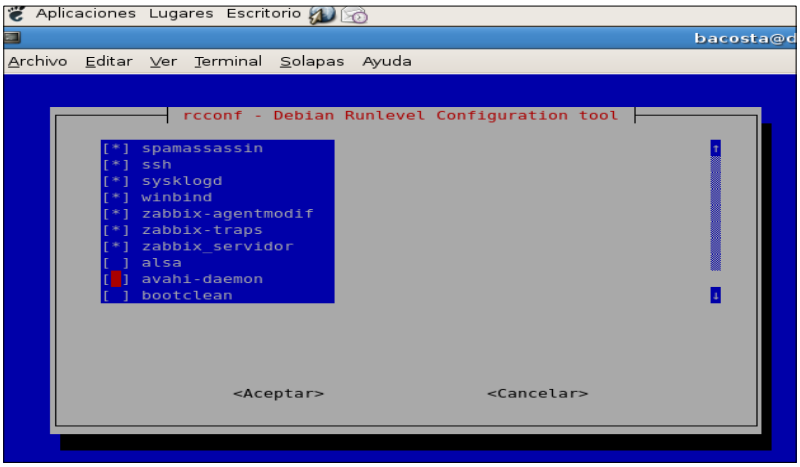


Figura 4.14 - Utilización de rcconf.

4.2.10 - EJECUCIÓN DE COPIA DE SEGURIDAD MEDIANTE CRON

En primer lugar, se realizará un sencillo *script*, como el mostrado en la tabla 4.6, que servirá para crear una copia de seguridad de la base de datos a diario, borrando la del día anterior. Con el fin de que se ejecute todos los días, se utiliza el cron del sistema. Existen dos métodos: el primero, consiste en copiar el *script* directamente en */etc/cron.daily* y el segundo, en copiarlo en */etc/init.d* y añadir al fichero */etc/crontab* la siguiente línea, que hará que la copia de seguridad se realice a las 23:55 de la noche:

*55 23 * * * root /etc/init.d/copia-seguridad.sh*

```
#####################################
### Backup base de datos + bzip ###
#####################################
#! /bin/bash
date=`date -I`
### Copia de la base de datos de zabbix
mysqldump --opt -u root --password="zabbix" --add-drop-table zabbix | bzip2 -c
> /root/zabbix__`date +%Y-%m-%d_%H-%M`.sql.bz2
rm /root/zabbix__`date +%Y-%m-%d_* -d '1 day ago'`.sql.bz2
```

Tabla 4.6 – *Script* para la copia de la base de datos.

4.2.11 - PROCESO DE ACTUALIZACIÓN

Se realizan los siguientes pasos:

- Interrupción de servicios - Todos los comandos mostrados a continuación se ejecutan desde el directorio raíz para parar los siguientes servicios:

  - Servidor: ./etc/init.d/zabbix_agentd stop

  - Agente: ./etc/init.d/zabbix_server stop

  - Recogida de traps: ./etc/init.d/snmpd stop

Antes de continuar, se ha de comprobar que los servicios se han parado correctamente:

netstat –tulpen

Si no se han parado correctamente, directamente utilizo el comando *kill* y termino los números de procesos correspondientes a dichos servicios.

- Copia de la base de datos - Como la copia se realiza a diario, simplemente se comprueba si existe la del día actual en el directorio */root*. De no ser así, sólo debería ejecutarse el *script* correspondiente.

- Descompresión del fichero de instalación - Se crea un directorio con el nombre del fichero a descomprimir y se descomprime el de la actualización:

mv zabbix-1.4.2.tar.tar /usr/local

cd /usr/local/

mkdir zabbix-1.4.2

tar xzvf zabbix-1.4.2.tar.tar

cd zabbix-1.4.2

Con estas órdenes se consigue la descompresión del fichero en el directorio creado y posteriormente se pasa a la compilación e instalación:

./configure --enable-server --enable-agent --with-mysql --with-net-snmp

make

make install

- Instalación del *frontend* -

Se ejecuta:

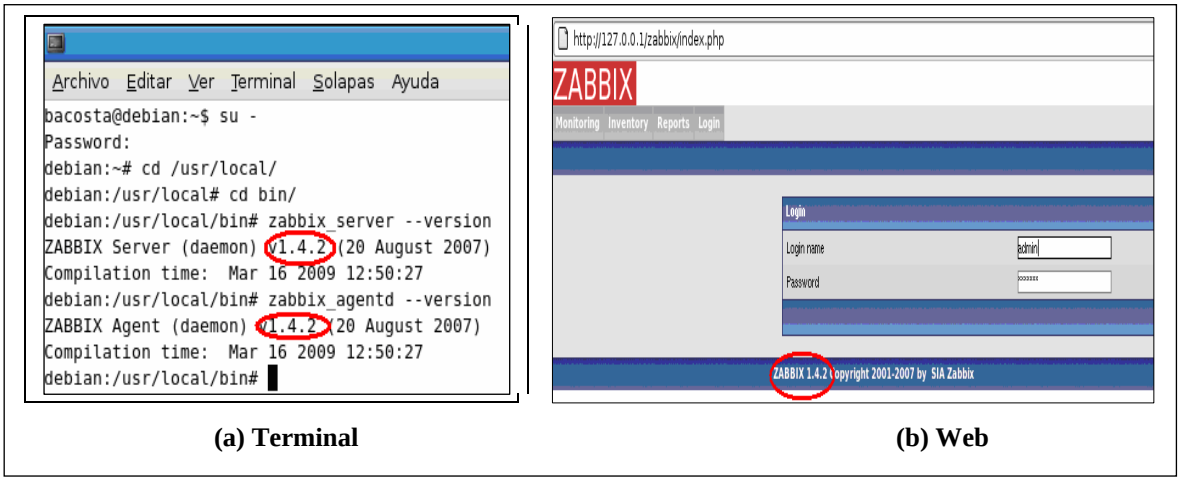
cp -R frontends/php/ /var/www/zabbix*

Con esto finaliza la instalación. Sólo queda reiniciar la máquina y después comprobar si la versión coincide con la que se acaba de instalar:

/usr/local/bin/zabbix_agentd --version

Como salida, habiéndose actualizado a la versión 1.4.2, debe obtenerse alfo parecido a lo mostrado en la figura 4.15.

Si todo se ha realizado correctamente y aun así, no se obtiene la versión deseada, es seguramente porque se deben copiar los ejecutables de la actualización en la carpeta */usr/local/bin.*



(a) Terminal **(b) Web**

Figura 4.15 - Comprobación de que la actualización se ha hecho correctamente.

4.3 - CONFIGURACIÓN

En primer lugar se realizará una breve explicación del manejo de las distintas opciones de la interfaz web de Zabbix. Tras esto, se comentarán los distintos pasos realizados para configurar al gestor Zabbix de RTVA.

4.3.1 - INTRODUCCIÓN A LA INTERFAZ WEB DE ZABBIX

Gracias a su sencillez, la configuración del gestor Zabbix se hará a través de su interfaz web. A continuación, se realiza una descripción de sus principales opciones.

Para acceder a la Web de la plataforma desde cualquier ordenador de RTVA, se debe abrir el navegador y poner en la barra de navegación la IP de la máquina donde se encuentra instalado el gestor Zabbix. Un ejemplo podría ser: *http://192.168.1.34/zabbix.* Entonces aparece una pantalla que es la correspondiente al grupo de usuarios *guest* y *que* no permite realizar cambios en la configuración. Para ello, se deberá acceder como usuario *admin*. Si es la primera vez que se accede, no existirá contraseña para este usuario. Por tanto, una vez registrado como administrador, se debe establecer una contraseña que impida al resto de usuarios modificar la configuración del sistema. Para este propósito, se pincha con el ratón sobre la opción *Profile*, que aparece en la esquina superior derecha. Tras realizar el cambio de contraseña, aparecerá una pantalla como la mostrada en la figura 4.16, donde se observan distintas pestañas que recogen las distintas opciones disponibles en el gestor. Aunque se puede cambiar el idioma de la interfaz web al español, se va a

mantener el inglés, pues facilita la comprensión de las distintas opciones explicadas en el manual de Zabbix. Además en español no traduce todas las opciones.

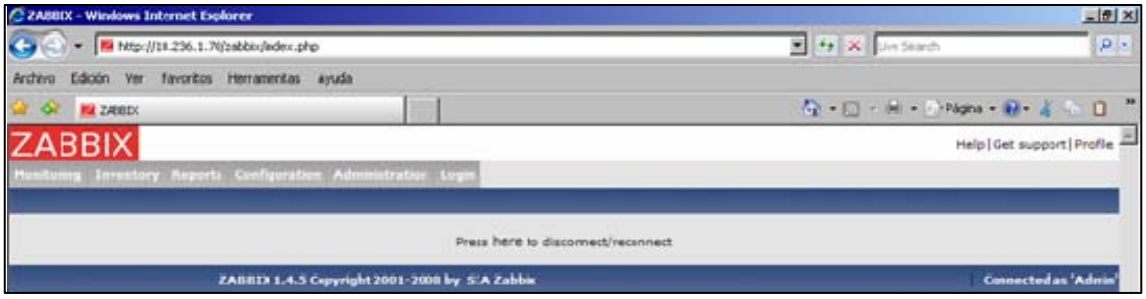


Figura 4.16 - Pantalla de inicio como administrador de Zabbix.

4.3.1.1.- Pestaña *Monitoring*

Es la pestaña que recoge las opciones relacionadas con la supervisión de los equipos. Al pinchar sobre esta opción, aparecen las siguientes:

- **Overview -** Mostrará si han saltado las alertas asociadas a los distintos parámetros de uno o varios equipos.

- **Web -** El módulo de monitorización WEB permite la supervisión fácil y flexible de la disponibilidad y el funcionamiento de sitios y aplicaciones WEB. Utiliza el intercambio de GET y POST y soporta tanto HTTP como HTTPS. La información recogida será el tiempo de respuesta, la velocidad de bajada y el código de respuesta de la aplicación. Este módulo sólo se utilizó en el proyecto, para la interfaz Web del servidor Zabbix. El resultado se visualiza en la figura 4.17.

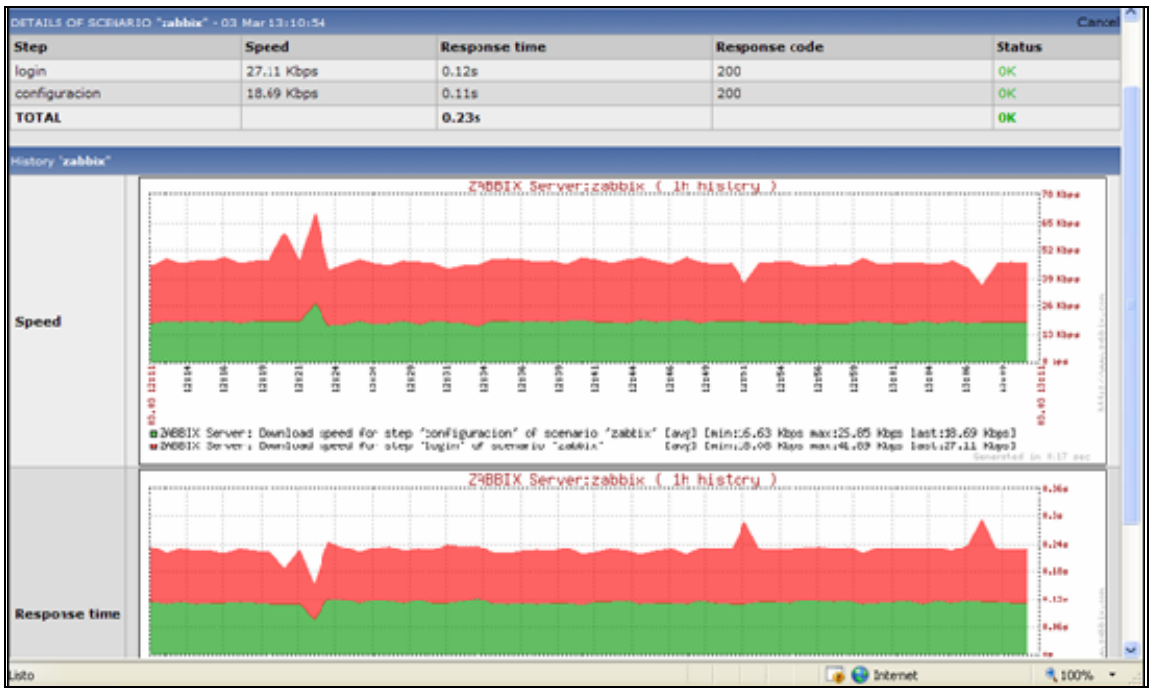


Figura 4.17 - Visualización del escenario web.

- **Latest data -** Permite visualizar de varias formas, el estado de las alertas de los equipos seleccionados. Pueden observarse mediante gráficos si son valores numéricos o mediante un histórico para todos los tipos. Esta opción se encuentra en la última columna.
- **Triggers -** Muestra el estado de las distintas alertas, asociadas a un dispositivo o a un conjunto de estos.
- **Queue -** Muestra el número de parámetros que aún no se han actualizado en la base de datos y cuanto llevan esperando. Vienen clasificados según la forma de recogerse: mediante agente Zabbix, agente SNMPv1, chequeo simple,...
- **Events -** Se visualizan los cambios producidos en las distintas alertas.
- **Actions** - Se visualizan las acciones producidas por motivo de una alerta.
- **Maps -** Muestra los mapas creados. Mediante distintos iconos para los dispositivos y diferentes colores para los enlaces, se sabrá si algo está fallando o no.
- **Graphs** - Sirve para visualizar los gráficos creados previamente, para determinados datos.
- **Screens** - Muestra las pantallas creadas previamente. Pueden estar formadas por mapas, gráficos y otras pantallas.
- **Discovery** - Es la acción que hace que Zabbix, busque en la red un equipo que cumpla unas determinadas reglas. Por ejemplo, que contenga a un agente Zabbix o a uno SNMP.
- **IT services** - Sirve para visualizar parámetros como el nivel de disponibilidad, el rendimiento u otros atributos definidos, para comprobar la calidad de los distintos servicios. No se va a utilizar.

4.3.1.2 - Pestaña *Inventory*

Muestra la *información general* de los equipos definidos en el gestor Zabbix. Se marca la casilla *use profile* cuando se crea el equipo y se rellenan los campos. Esta es la opción que se utilizará para los datos que se consideraron permanentes anteriormente. Por ejemplo, para el servidor Zabbix se rellenó tal y como aparece en la figura 4.18.

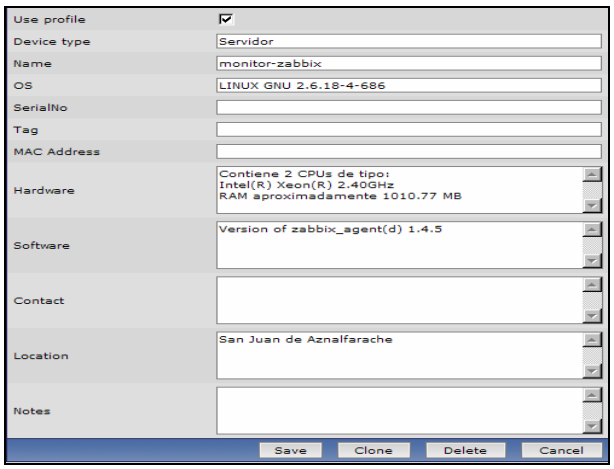


Figura 4.18 - Insertando datos del equipo Zabbix en el inventario.

4.3.1.3 - Pestaña *Reports*

Muestra la información relacionada con la plataforma Zabbix.

- **Status of Zabbix -** La información que ofrece se muestra en la figura 4.19. Se trata del estado del servidor Zabbix y el número de dispositivos, elementos, condiciones, eventos y alertas que se han definido.

- **Availability report** - Muestra el porcentaje de tiempo que las alertas han permanecido en un determinado estado, dando una idea de la disponibilidad de cada equipo, en función de los aspectos monitorizados.

- **Most busy triggers top 100 -** Se visualizan las alertas que más veces han cambiado de estado, durante un determinado periodo de tiempo.

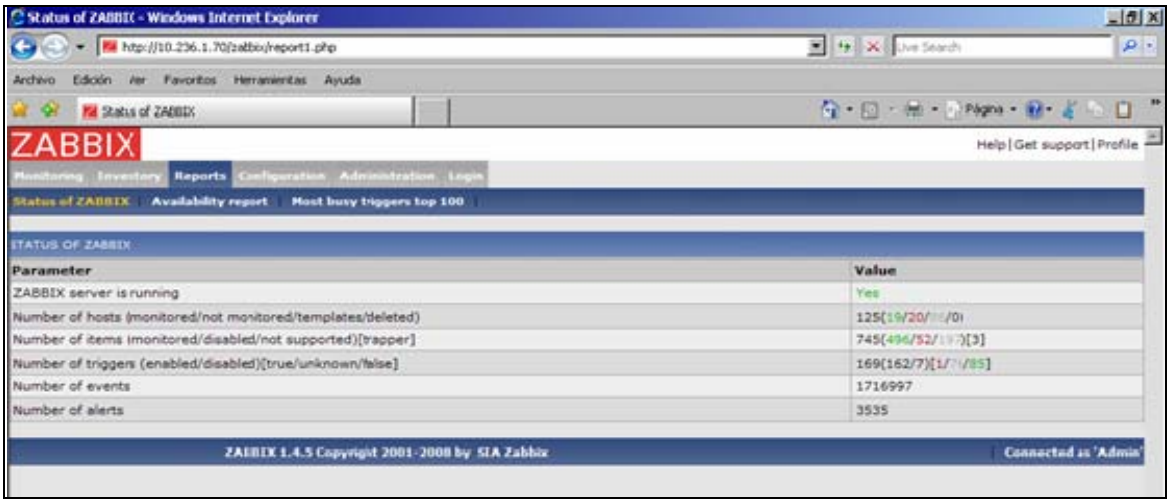


Figura 4.19 - Estado de Zabbix.

4.3.1.4 - Pestaña *Configuration*

Tal y como su nombre indica, es el apartado donde se realiza la configuración de la plataforma. Sus pestañas son:

- **General -** Permite distintas opciones, que se van seleccionando a través de la pestaña que aparece en la parte superior derecha. Se podrá realizar entre otras, la configuración del tiempo de almacenamiento de eventos y acciones, la importación de imágenes de fondo o de dispositivos, el mapeado entre elementos numéricos y texto, etc.
- **Web -** Para la configuración de la página web a monitorizar.
- **Hosts -** Es la opción que permite añadir equipos, grupos de equipos y plantillas de elementos a la plataforma. Simplemente se selecciona el tipo a añadir y luego se pulsa *create.* Más adelante se especificará como se han añadido los distintos componentes.
- **Items -** Permite la creación, modificación, activación, desactivación o borrado de las variables a monitorizar de los distintos equipos.
- **Triggers -** Se pueden crear, modificar, activar, desactivar y borrar distintas alertas asociadas a las variables anteriores.
- **Actions -** Se pueden crear procesos que se ejecutan al activarse una alerta. Pueden ser: el envío de un email, de un sms, la ejecución de comandos remotos en un dispositivo, etc.
- **Maps -** Permite crear mapas para visualizar fácilmente, el comportamiento de los distintos componentes.
- **Graphs -** Es donde se pueden crear los gráficos simples o múltiples de los valores recogidos por una variable numérica.
- **Screens -** Además de pantallas formadas por distintos componentes ya comentados, también permite la sucesión en el tiempo de estas.
- **IT services -** Aquí se configuran los parámetros relacionados con la SLA (*Service Level Agreement*).
- **Discovery -** En este apartado es donde se crean los descubrimientos, imponiendo las reglas a seguir.
- **Export/ Import -** Permite guardar la configuración existente para poder traspasarla a otras plataformas. También se pueden importar configuraciones de otros sistemas. Los datos quedan almacenados en un fichero con extensión *.xml*.

4.3.1.5 - Pestaña *Administration*

Existen distintas opciones:

- **Users -** Ofrece la opción de crear usuarios y grupo de estos y configurar los accesos. Además también permite visualizar qué usuarios están conectados a la plataforma.
- **Media Types -** En esta opción se añaden los medios disponibles para comunicar a la plataforma con los administradores. Estos pueden ser sms, email, Jabber,...
- **Audit -** Refleja los cambios realizados en la configuración de la plataforma a través de la interfaz Web, así como los inicios de sesión. Estos aparecen como en la figura 4.20.
- **Notifications -** Muestra las notificaciones enviadas a los administradores.
- **Instalation -** Permite testear el correcto funcionamiento de los componentes de la plataforma, tales como PHP y MySQL.

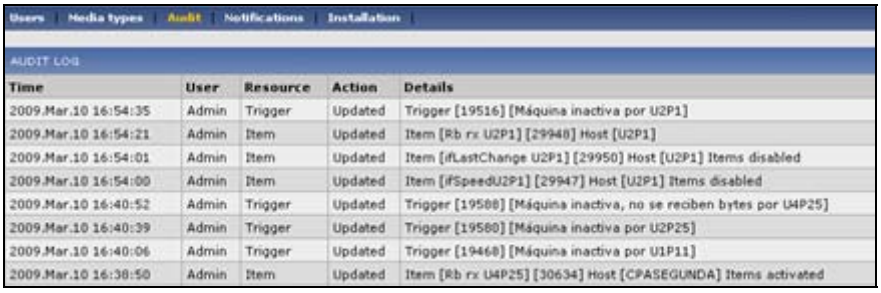


Figura 4.20 - Registro de las operaciones realizadas en la configuración.

4.3.1.6 - Pestaña *Login*

Permite abrir o cerrar una sesión. Si se accede a la plataforma web de Zabbix, por defecto aparecerán todas las pestañas del menú excepto la de la configuración. Esta aparece cuando accedes como *administrador*. Más adelante, se comentará como configurar las opciones que aparecen al iniciar con usuario *guest* (es con el que se accede por defecto).

4.3.2 - CÓMO UTILIZAR LA INTERFAZ WEB DE ZABBIX

El procedimiento para añadir cualquier tipo de dispositivo, será siempre el mismo: primero, la creación de una plantilla que contendrá a todos los elementos que se desean monitorizar de un mismo dispositivo, después la creación de estos y por último, la creación del dispositivo con la respectiva vinculación a su plantilla. Tras estos pasos, se añadirán las alarmas y acciones asociadas y para finalizar, se creará un grupo y un mapa que contendrá a todos los elementos de una delegación. Se pasan a explicar la realización de estos pasos

mediante la interfaz Web. Se utilizarán las opciones de la pestaña *Configuration*. Como ya se ha mencionado, se deberá acceder como administrador, para conseguir los permisos adecuados.

4.3.2.1 - Creación de *templates*

Es la opción que permite añadir plantillas de elementos. Hay que dirigirse a la pestaña *hosts*, elegir la opción *templates* y pulsar sobre *Create Template*. En este paso, simplemente se le asignará un nombre.

Zabbix por defecto trae plantillas ya definidas para Windows, Linux, SNMPv1,... se usarán algunas de estas como ya se verá más adelante.

4.3.2.2 - Creación de *items*

Los *items* no son más que los elementos que permitirán monitorizar las distintas variables. Para su creación, se pulsa sobre las pestaña *items* y después sobre *Create item*. En primer lugar, se pone la descripción de la variable y el tipo, en función de con que opción se va a monitorizar. Después se debe rellenar la *key*, que será una especie de identificador único para distinguir a las variables asignadas a un mismo *host*. Las opciones *keep history* y *keep trends* establecen el número de días, que permanecerán almacenados todos los datos y estadísticas. Para este campo en principio, se utilizará el mismo valor para todos los dispositivos. La última variable a comentar es la del estado, que debe ser *active*. Las demás opciones son específicas para cada dispositivo y por tanto, las comentaremos más adelante.

Tras la creación de todos los elementos de una misma plantilla, se vinculan con esta a través de su modificación en el menú correspondiente.

4.3.2.3 - Creación de *host group*

Para cada delegación de RTVA, se creará un grupo que contenga a todos los dispositivos. Simplemente se pulsa sobre *hosts*, se selecciona *host groups* y luego *create group*. Por último, se elige un nombre.

4.3.2.4 - Creación de *hosts*

Se creará un *host* por cada dispositivo o bloque de estos (los bloques de switches con una única IP, se tratarán como un único *host*). Este paso es tan sencillo como pulsar sobre la pestaña *hosts* y a continuación, en *create host*. Se introduce el nombre y se selecciona la pestaña del grupo creado anteriormente, para hacerle pertenecer a este. Se selecciona la

opción *use IP address*, que permitirá al gestor conectarse con este dispositivo mediante su IP. En la opción *Link with Template*, se selecciona la plantilla de elementos creada para monitorizar los parámetros del equipo, con el fin de vincularla a este. Por último, se marca la opción *use profile* y se rellenan los datos del dispositivo, para conseguir tener un inventario de la red monitorizada. Debe asegurarse de que el estado del dispositivo sea *Monitored.*

Si estos pasos se han realizado correctamente, al pulsar la pestaña *hosts* y seleccionar el grupo creado, deben aparecer todos los dispositivos de esa delegación en estado monitorizado y habilitado. Si no aparece habilitado, será debido a que el gestor no es capaz de comunicarse con el agente y se tendrá que descubrir el porqué.

4.3.2.5 - Creación de *triggers*

Para crear los *triggers*, primero de selecciona el grupo y la máquina para la que se crean, se pulsa la pestaña de igual nombre y después sobre *Create Trigger*. Se rellena el campo del nombre y a continuación, en el campo *Expression*, debe escribirse la expresión lógica, que realizará el cambio de estado de la alerta. Para las distintas expresiones que soporta, se puede consultar el manual. Después se seleccionará la severidad con la opción *severity* y en *comments*, se explicará el problema, la posible solución y la persona de contacto.

4.3.2.6 - Creación de *actions*

De nuevo, se pulsa la pestaña de igual nombre y después, sobre la opción *Create action*. En *Event Source*, se elige si la acción va a estar relacionada con una alarma o con un descubrimiento. Se pueden introducir varias condiciones para una misma acción, pulsando *New* en la opción *Conditions*. Aparecerán unas pestañas, que permiten introducir las reglas para el cálculo de las condiciones, que se deben cumplir para que se realice la acción. Después pulsando la tecla *new* referente a las *operations*, se encontrarán las opciones para enviar el mensaje a uno o a un grupo de usuarios, el tema del mensaje, el texto del mismo, la opción y el número de veces que se debe enviar si no cambia el estado de la condición, el tiempo entre envíos y por último el estado de la acción. Todas estas opciones, pueden observarse en la figura 4.21.

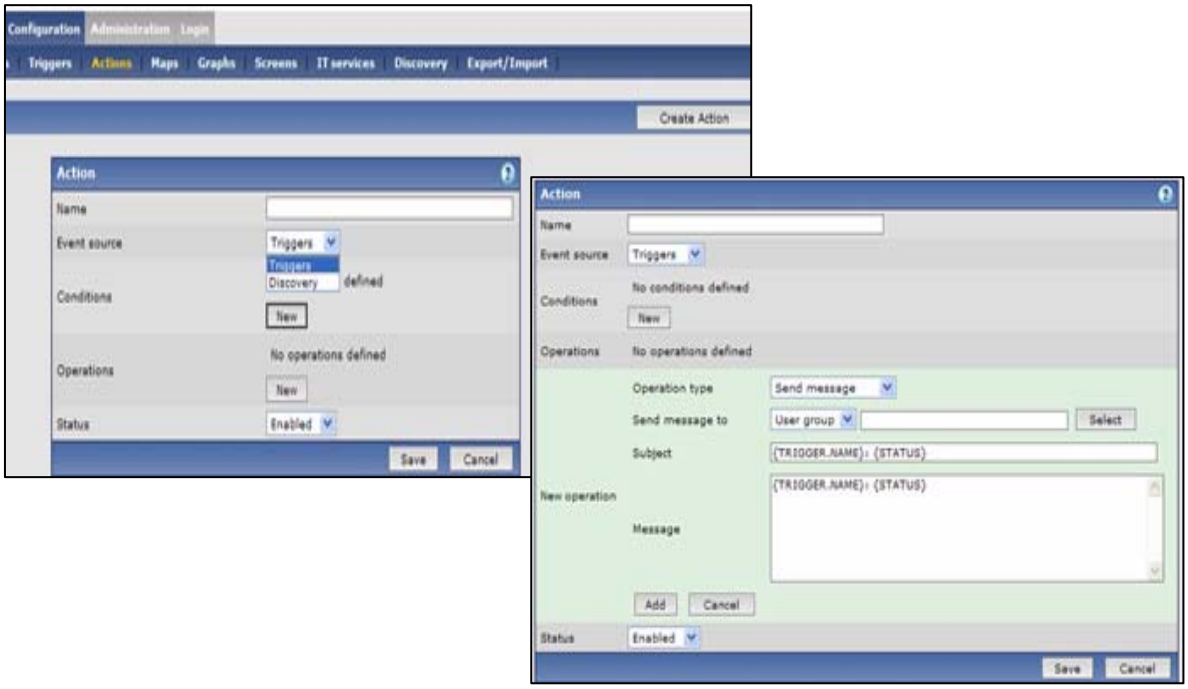


Figura 4.21 - Opciones que aparecen en la creación de una acción.

4.3.2.7- Creación de *maps*

Se creará un mapa por delegación y uno de todas estas, para asegurarse de que en todo momento su funcionamiento es el correcto.

Para cada dispositivo se deben introducir dos iconos como mínimo, uno para que se muestre si ninguna alarma asociada a este ha cambiado y otro que se muestre cuando cambie, para alertar a la persona encargada de la supervisión de la red.

También se introducirán fondos para mostrar la topología real de la subred y para poner fotos de las distintas delegaciones.

Los pasos para crear un mapa son siempre los mismos: Pestaña *Maps-> Create Map* y a continuación se rellenan los campos mostrados en la figura 4.22.

Para poder disponer de los fondos e iconos deseados, previamente deben haberse introducido en el gestor, utilizando la opción correspondiente en la pestaña *General*.

Tras esta breve introducción a las distintas opciones de configuración, se pasa a aplicarlas a cada dispositivo en particular.

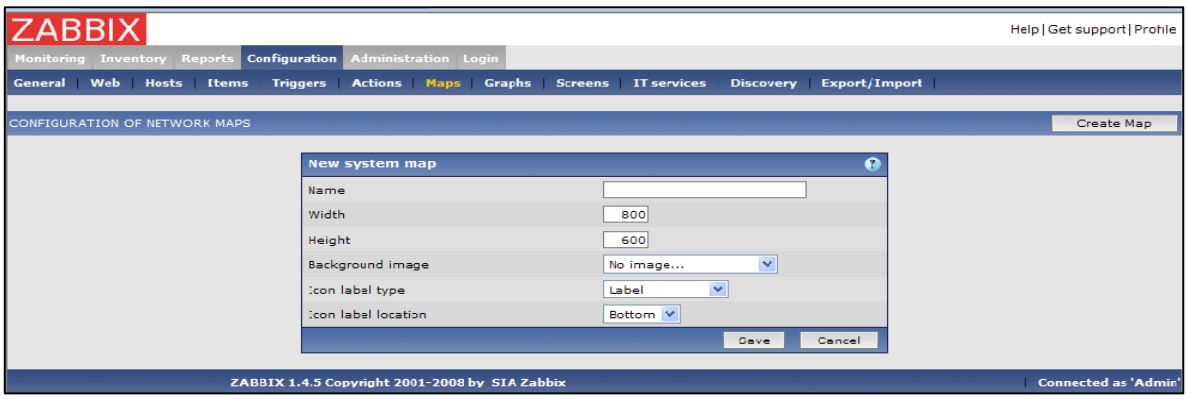


Figura 4.22 - Opciones para la creación de mapas.

4.3.3 - ADICIÓN DE DATOS DESDE LA INTERFAZ WEB DE ZABBIX

Se pasan a describir los pasos realizados para la configuración de los dispositivos, que se pretenden gestionar en este proyecto.

4.3.3.1 - Configuración de Switches monitorizados mediante SNMP

Como ya se eligieron las variables a monitorizar y se averiguaron los OID de las mismas, el siguiente paso será la definición de estas variables en el gestor Zabbix.

Como ya se comentó, los parámetros a monitorizar están relacionados con los puertos del *switch*. Por tanto, en primer lugar, se creará una plantilla de nombre U*i*P*j*, donde *i* representará el número de unidad y *j* el número de puerto, con i= 1,...,4 y j= 1,...,26. Por ejemplo, se puede observar en la figura 4.23, la plantilla creada para monitorizar el puerto 2 de la unidad 1 del bloque de *switches*. Será la U2P1.

Para la creación de los distintos *items*, se debe tener en cuenta que Zabbix no permite cargar directamente una MIB y por ello, hay que introducir uno a uno, los OID de los elementos que se desean monitorizar. Existirán una serie de opciones, que se rellenarán con el mismo valor para todos ellos:

- Serán de tipo SNMPv1.

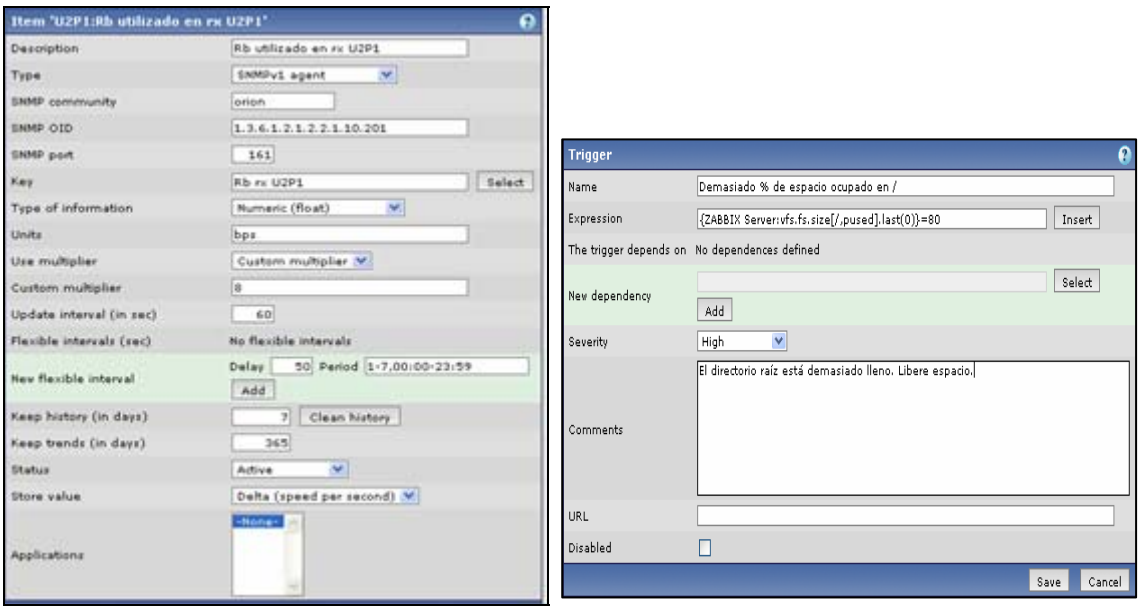
- El historial de todos los datos se mantendrá durante 7 días.

- El historial de las estadísticas de los datos, se mantendrá durante 365 días.

- El estado será activo.

- El intervalo de actualización será de 60 segundos.

Las opciones específicas para cada una, se representan en la tabla 4.7.



(a) Pantalla de creación de elemento **(b) Pantalla de creación de alarma**

Figura 4.23 - Ejemplos de pantallas de configuración de switch.

Se puede observar que para los octetos entrantes y salientes, la información se almacenará y visualizará en RB (*Régimen Binario*). Por eso, se utiliza un multiplicador 8, pues un octeto son 8 bits, como unidades, *Units,* "bps" y como valor almacenado, campo *Store value,* aparecerá el valor *Delta(speed per second),* que será lo que calcule el RB en bps. La pantalla de la creación del *item*, quedará tras completar los campos como la mostrada en la figura 4.23-a. Poniendo como unidades "bps", bit/s, Zabbix se encargará de calcular y mostrar los prefijos correspondientes: K,M,G,etc.

Estas variables se irán repitiendo para cada uno de los puertos del switch, pero debido a que cambian los puertos, no se podrá utilizar la misma plantilla para todos. En SNMP cada variable se identifica por un único OID y dentro de un switch, el mismo parámetro para distintos puertos, representa a dos variables y por tanto, contienen OID distintos. Por ejemplo, para monitorizar el puerto 1 de la unidad 1, se creó la plantilla U1P1. Esta servirá para todos los puertos 1 de todos los switches que hacen de unidad 1, ya que coinciden sus OID. Sin embargo, si cambia la unidad a la 2 o el puerto a cualquier otro, como por ejemplo el 15, ya existirá un cambio de OID que obligará a definir otras dos plantillas, la U2P1 para el primer caso o la U1P15 para el segundo. Por tanto, se realiza una plantilla para cada interfaz que se quiere monitorizar de cada unidad.

Una vez creados todos los elementos, se vinculan a la plantilla correspondiente. Para ahorrar tiempo, se pueden copiar las plantillas y después bastará con editarlas. Los elementos de la plantilla U2P1 se visualizan en la figura 4.24.

El siguiente paso será la creación de las alarmas. Se activarán cuando ocurran las siguientes situaciones:

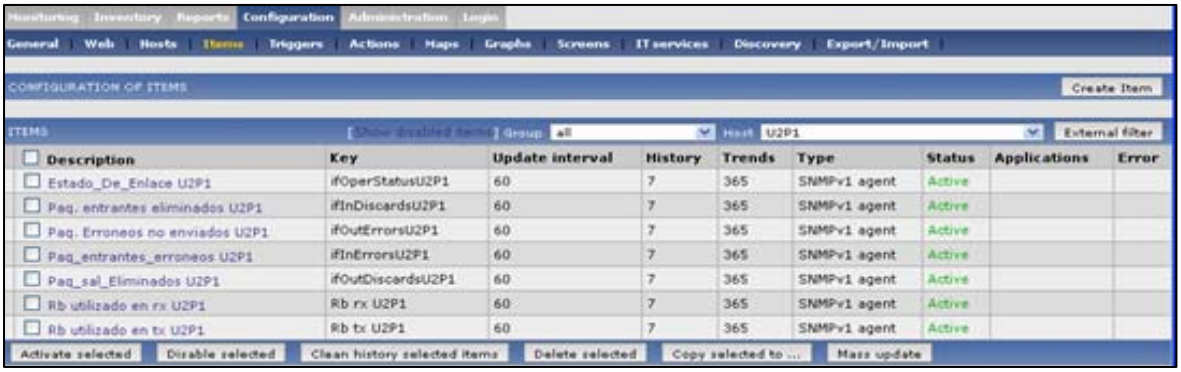
- El estado del enlace sea *off*. Esta alarma ocurrirá cuando un puerto del *switch* se desactive por alguna circunstancia. Para enlaces que no están duplicados, la severidad de la alarma será de nivel desastre. Si por el contrario, el enlace se encuentra duplicado, la alarma individual tendrá nivel alto y cuando ambos se encuentren inactivos, entonces saltará una alarma con nivel desastre.

Variable	ifOperStatus	ifInDiscards	ifOutErrors	ifInErrors	ifOutDiscards	ifInOctets	ifOutOctets
Descripción	Estado	Paquetes entrantes eliminados	Paquetes erróneos no enviados	Paquetes entrantes erróneos	Paquetes salientes eliminados	Número de octetos recibidos	Número de octetos transmitidos
Key	ifOperStatus U1P2	ifInDiscards U1P2	ifOutErrors U1P2	ifInErrors U1P2	ifOutDiscards U1P2	ifInOctets U1P2	ifOutOctets U1P2
Tipo de información	Character	Numeric (float)	Numeric (float)	Numeric (float)	Numeric(float)	Numeric (float)	Numeric(float)
Unidades	No aparece	--------	--------	--------	--------	bps	bps
Uso de multiplicador	No aparece	Do not use	--------	--------	--------	Custom multiplier	Custom multiplier
Multiplicador	No aparece	--------	--------	--------	--------	8	8
Intervalo de actualización	60	60	60	60	60	60	60
Valor almacenado	--------	--------	--------	--------	--------	Delta(speed per second)	Delta(speed per second)
Aplicaciones	--------	--------	--------	--------	--------	--------	--------

Tabla 4.7- Ejemplo de configuración del puerto 2 de la unidad 1 (U1P2).

- Alto número de paquetes entrantes erróneos. En un principio, se pensó utilizar alarmas de distintos niveles de criticidad, según el % de paquetes erróneos del total que llegan. Pero entonces surgió un problema, debido a que Zabbix no permite realizar alarmas que comparen variables. Para entenderlo mejor, veamos el siguiente ejemplo: se desearía poner una alarma, cuando el 20 % de paquetes entrantes fuesen erróneos. La variable correspondiente al número de paquetes entrantes erróneos contiene la *key*

"ifInErrors" y la de los paquetes entrantes en total, "ipInReceives". La condición para la alarma sería la siguiente expresión:



{ifInErrors.last(0)} > {(ifInReceives.last(0))*0.2}

Figura 4.24 - Elementos de la plantilla U2P1.

Sin embargo, Zabbix sólo permite que el valor de la condición sea una constante. Así que como solución, se optó por crear una alarma que se activara, cuando los paquetes entrantes se incrementaran de forma considerable durante varios intervalos de tiempo. Se observó de los datos recogidos, que cuando la red funciona correctamente, el número de octetos erróneos vale en media 0, así que consideraremos que si se incrementa hasta un valor de 50 o más, cada vez que se obtiene el valor y sucede durante 5 minutos, saltará una alarma de *warning*. Si continúa durante 15, la alarma será de severidad alta, *high*, y si permanece hasta los 30, la severidad cambiará a desastre.

Para definir todos estos triggers, se utilizarán las siguientes expresiones:

{U1P1: ifInErrors.count(300,50,"ge")}=5;

{U1P1: ifInErrors.count(900,50,"ge")}=15;

{U1P1: ifInErrors.count(1800,50,"ge")}=30;

La primera condición para que se active la alarma, es si la variable reúne 5 valores mayores o iguales a 50 durante 300 segundos. Para la segunda, son 10 valores durante 900 segundos y para la tercera, 30 durante 1800 segundos. En el manual, se pueden observar otras funciones permitidas para la expresión de los *triggers*.

Para las siguientes variables citadas, se continuarán definiendo las alarmas de esta misma forma. Por supuesto, variará la *key*, ya que se trata de una variable distinta, y el umbral con el que se compara el dato.

Este *trigger* se llamará "Alto número de errores de entrada".

- Alto número de paquetes salientes erróneos. En esta la *key* es "ifOutErrors" y el valor es de 50, pues se observó que a veces podían existir 25 paquetes no enviados y no se detectaban fallos en la red. El triggers se llamará "Alto número de errores de salida".

Estas dos últimas variables, detectan que se está dando un error en algún sitio, bien en la red, en un switch, en un computador, etc,...pues no será normal que lleguen tantas tramas erróneas.

- Alto número de paquetes entrantes descartados. La *key* de estas variables es "ifInDiscards" y su valor para el que salta la alarma es de nuevo 50. La media que se obtuvo para esta variable fue de 0 paquetes eliminados. Este *trigger* tendrá como nombre "Buffer de entrada demasiado lleno".

- Alto número de paquetes salientes descartados. Su *key* correspondiente es "ifOutDiscards". El valor a partir del cual salta la alarma es 50. El valor medio que se obtuvo, fue de 23 y todo funcionaba con normalidad. Este *trigger* se llamará "Buffer de salida demasiado lleno".

Estos dos últimos *triggers* se activarán cuando el *buffer* esté saturado, por eso descarta tramas.

Aun habiendo elegido valores para que salten dichas alarmas, se deberá continuar observando los datos de estas variables, por si en algún momento de produce un mal funcionamiento del dispositivo para valores menores de 50. En ese caso, se cambiaría la alerta para que se activara a ese valor.

- Régimen binario de entrada nulo - Para este dato se utilizó la variable "Rb rx" referida al número de octetos de entrada en la interfaz. Ahora en la expresión de la alarma, en vez de utilizar "ge", se utilizó "eq" que significa "igual" y como valor, se optó por el "0". Así, cuando no se reciba por el enlace durante 5,15 o 30 minutos, se activarán las alarmas correspondientes con las distintas severidades.

- Régimen binario de salida nulo. Se define de la misma forma que el anterior, variando únicamente la *key*, que ahora será "Rb tx".

  Los nombres de estos dos últimos *triggers*, serán "No se rx bits" y "No se tx bits". Se debe tener cuidado con estas alarmas, cuando existen enlaces duplicados con el Spanning Tree activado. En este caso, se realizará una alarma que se activará cuando ninguno de los enlaces reciba o transmita tráfico.

Cabe destacar que las *keys* de todos los *items* definidos, terminarán con la unidad y número de puerto como se muestra en la de la figura 4.24. De igual forma se hará en la expresión de los *triggers*.

Tras la definición de todos los elementos, se crea un *host group* por cada delegación de RTVA. Después, por cada bloque de switch o por cada uno de estos con una determinada IP, se define un *host* y se le vinculan las plantillas correspondientes a los puertos que se deseen monitorizar de este.

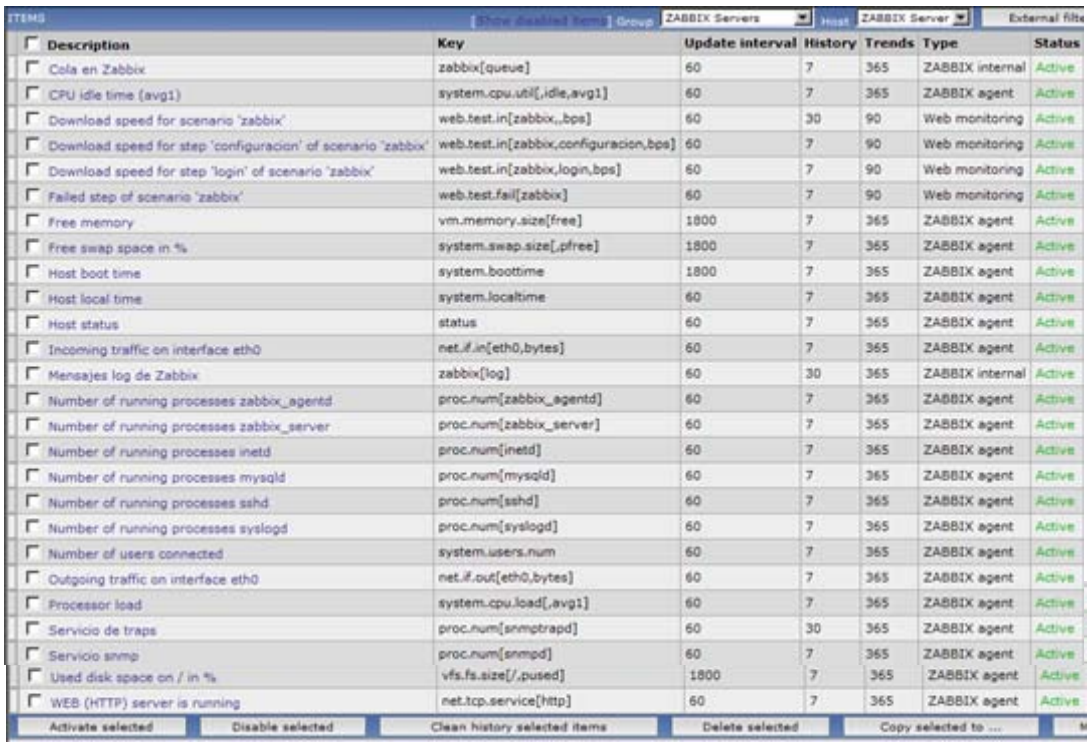
Este *host* se añade al *host group* correspondiente, según su delegación. El siguiente paso, será la creación del mapa para dicha delegación, donde aparecerá un determinado dibujo, cuando el switch funcione correctamente, que cambiará al saltar una de las alarmas creadas para este dispositivo. Si las alarmas que saltan son las asociadas al enlace, este cambiará de color, siendo el rojo el que expresa el error del mismo.

4.3.3.2 - Configuración de Servidores con agente Zabbix instalado

Los primeros pasos van a ser siempre los mismos. Creación o descubrimiento del equipo, adición de este a la delegación a la que pertenezca y vinculación con la plantilla correspondiente. En este caso, las plantillas ya vienen definidas en el gestor Zabbix. Se trata de la *Template_Linux* y *Template_Windows*, para servidores Linux la primera y Windows la segunda, donde sólo se dejarán activas las variables que se corresponden con los elementos elegidos en el apartado de Estudio. Los elementos que interesen y que no existan en esta plantilla, se deberán añadir de la forma ya explicada. Para las expresiones se estos, puede consultarse el manual de Zabbix. Como ejemplo en la figura 4.24, aparecen los *items* escogidos para el servidor Zabbix.

Quedarían por crear las distintas alarmas. Se definieron para todas las máquinas que eran gestionadas por agente Zabbix, fuese cual fuese su sistema operativo. Estas se activarán para las siguientes condiciones:

- Alto porcentaje de espacio ocupado en directorio raíz para Linux o de disco duro para Windows. Se crearán dos alarmas: la primera se activará cuando el espacio ocupado en % alcance el valor de 80 y será de severidad *high* y la segunda, cuando alcance el valor de 90 y será de severidad *disaster*.
- Alto porcentaje de memoria swap utilizada. Se definirán dos exactamente iguales a las anteriores.



Description	Key	Update interval	History	Trends	Type	Status
Cola en Zabbix	zabbix[queue]	60	7	365	ZABBIX internal	Active
CPU idle time (avg1)	system.cpu.util[,idle,avg1]	60	7	365	ZABBIX agent	Active
Download speed for scenario 'zabbix'	web.test.in[zabbix,,bps]	60	30	90	Web monitoring	Active
Download speed for step 'configuracion' of scenario 'zabbix'	web.test.in[zabbix,configuracion,bps]	60	7	90	Web monitoring	Active
Download speed for step 'login' of scenario 'zabbix'	web.test.in[zabbix,login,bps]	60	7	90	Web monitoring	Active
Failed step of scenario 'zabbix'	web.test.fail[zabbix]	60	7	90	Web monitoring	Active
Free memory	vm.memory.size[free]	1800	7	365	ZABBIX agent	Active
Free swap space in %	system.swap.size[,pfree]	1800	7	365	ZABBIX agent	Active
Host boot time	system.boottime	1800	7	365	ZABBIX agent	Active
Host local time	system.localtime	60	7	365	ZABBIX agent	Active
Host status	status	60	7	365	ZABBIX agent	Active
Incoming traffic on interface eth0	net.if.in[eth0,bytes]	60	7	365	ZABBIX agent	Active
Mensajes log de Zabbix	zabbix[log]	60	30	365	ZABBIX internal	Active
Number of running processes zabbix_agentd	proc.num[zabbix_agentd]	60	7	365	ZABBIX agent	Active
Number of running processes zabbix_server	proc.num[zabbix_server]	60	7	365	ZABBIX agent	Active
Number of running processes inetd	proc.num[inetd]	60	7	365	ZABBIX agent	Active
Number of running processes mysqld	proc.num[mysqld]	60	7	365	ZABBIX agent	Active
Number of running processes sshd	proc.num[sshd]	60	7	365	ZABBIX agent	Active
Number of running processes syslogd	proc.num[syslogd]	60	7	365	ZABBIX agent	Active
Number of users connected	system.users.num	60	7	365	ZABBIX agent	Active
Outgoing traffic on interface eth0	net.if.out[eth0,bytes]	60	7	365	ZABBIX agent	Active
Processor load	system.cpu.load[,avg1]	60	7	365	ZABBIX agent	Active
Servicio de traps	proc.num[snmptrapd]	60	30	365	ZABBIX agent	Active
Servicio snmp	proc.num[snmpd]	60	7	365	ZABBIX agent	Active
Used disk space on / in %	vfs.fs.size[/,pused]	1800	7	365	ZABBIX agent	Active
WEB (HTTP) server is running	net.tcp.service[http]	60	7	365	ZABBIX agent	Active

Figura 4.25 - Elementos monitorizados de Zabbix.

- Poca memoria libre. Se refiere a la RAM del sistema. El inconveniente de esta alarma es que los datos no se recogen en % y por tanto, se debe conocer el tamaño total de la memoria para cada máquina. Las condiciones para que las dos alarmas se activen, dependerán del total de memoria del que disponga la máquina. Por ejemplo, Zabbix dispone de una RAM de 1 Gbyte. Por tanto, la primera alarma saltará cuando tome el valor de $15 \cdot 10^6$ bytes, para lo cual sólo saltará una alerta de severidad alta, y la segunda con un valor igual o menor de $10 \cdot 10^6$ bytes con severidad desastre.

- Fallo del puerto por el que el servidor escucha las peticiones. Se comprobará si el puerto está accesible y si no lo está, saltará una alarma de severidad *high*.

- Comprobación de los servicios levantados. Si no lo están, se activará una alerta cuyo nivel de criticidad dependerá del tipo de servicio. Por ejemplo, si en Zabbix cae el servicio del servidor, la criticidad será de nivel *disaster*, en cambio si lo que cae es el servidor web, este será *high*. Esto se eligió así, porque Zabbix puede seguir recogiendo datos, que es su funcionalidad principal, sin ofrecer interfaz web, pero no si su servicio *Zabbix Server* falla.

- El servidor no recibe o no transmite. Si ocurre que el tráfico de entrada es nulo, se activa una alarma de severidad alta. Se creará otra igual para el de salida. Es indispensable que exista tráfico de entrada y salida, si no los servidores no se estarían comunicando con las máquinas a las que tienen que ofrecer su servicio.

En el caso de Zabbix, se consideró que si el último valor de alguna de estas variables valía 0, saltaría la alarma correspondiente. Se eligió así, porque Zabbix recoge muchos valores en intervalos de un minuto y si en algún momento la red falla, significa que estos se han dejado de recoger.

Todos los triggers creados para el servidor Zabbix, pueden observarse en la figura 4.26.

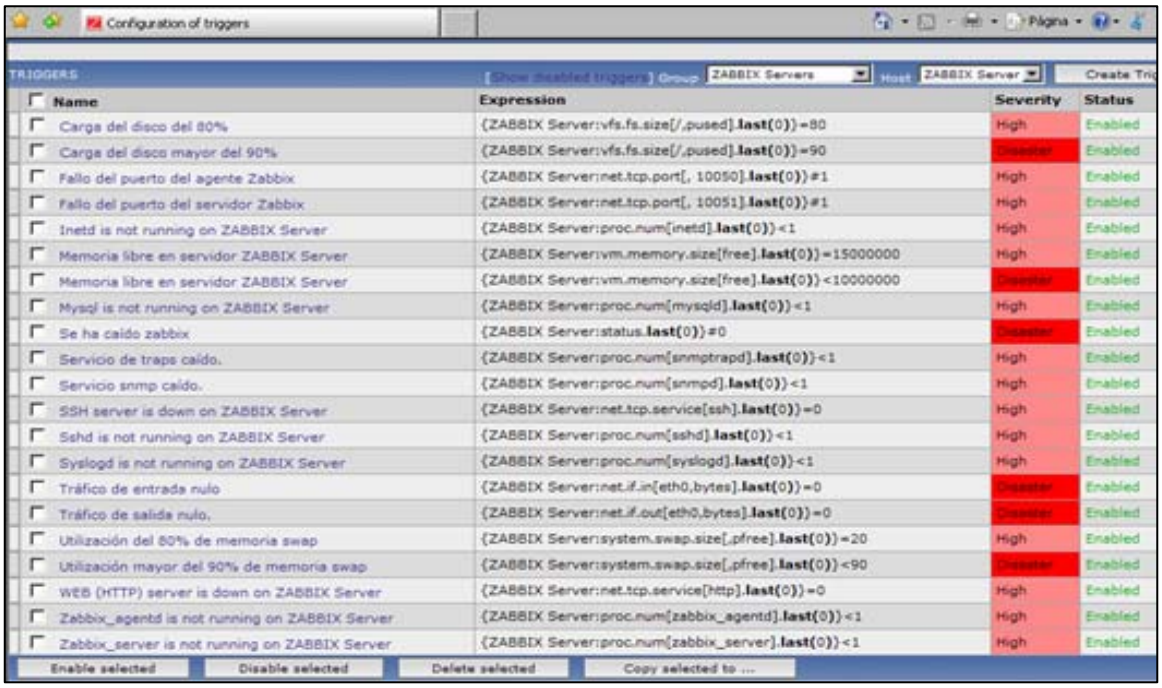


Figura 4.26 - Alertas del servidor Zabbix.

4.3.3.3 - Configuración de Servidores monitorizados mediante chequeos

Es el caso del servidor Esafe para el que se utilizó el chequeo simple. La configuración es bastante sencilla, sólo consiste en vincular la plantilla Template_Standalone, mostrada en la figura 4.27 y que ya viene definida por Zabbix, al *host* que se definió para este servidor. En el manual Zabbix aparece algún elemento más que se puede añadir.

Los triggers para este servidor se activarán si alguna de estas variables da un valor distinto de "1" (significa servicio funcionando correctamente).

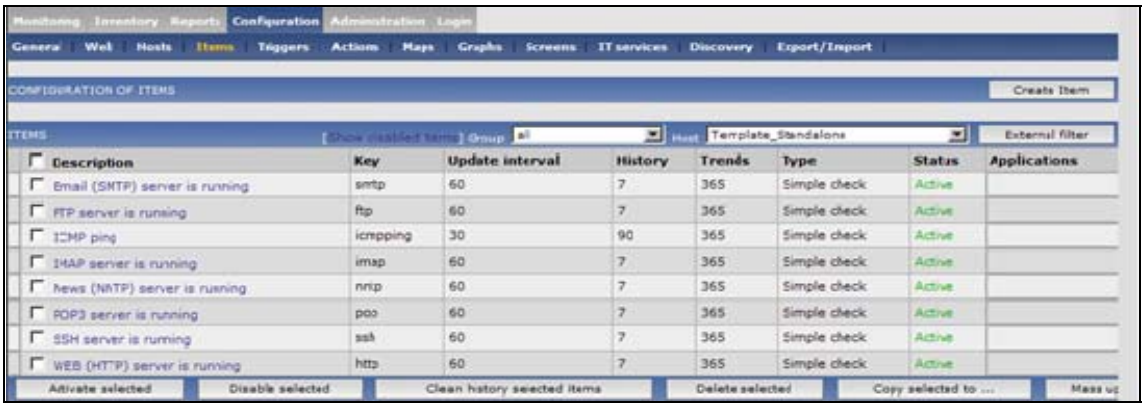


Figura 4.27 - Plantilla para los chequeos simples.

Otro tipo de chequeo que sólo puede ser utilizado por el servidor Zabbix, es el conocido como "Chequeo interno" que da información sobre la base de datos creada. Para este también existe una plantilla definida, *Template_App_MySQL*. Esta se vinculó al servidor Zabbix y se deshabilitaron los elementos que no interesaban. No se definieron alarmas para estos *items*.

4.3.3.4 - Creación de acciones: Envío de alertas vía email o sms

Las acciones serán siempre las mismas sea cual se la alerta, ya que sólo dependerá de la criticidad del *trigger*. Para las alertas con severidad desastre, se enviará un email y un sms al administrador, mientras que para las de severidad alta, solamente se enviará un email.

Para hacer posible el envío correcto del email o del sms, previamente se deben definir y configurar los parámetros necesarios. Para ello, hay que dirigirse a *media types* y luego pulsar sobre *Create Media type*. A continuación se rellenan los campos. Para este proyecto se definieron las herramientas que aparecen en la figura 4.28.

Se puede observar que el envío de sms, se hará a través de la ejecución del script correspondiente en el servidor Zabbix.

Figura 4.28 - Acciones definidas en Zabbix.

Por otro lado, se deben crear dos acciones. Una para el envío de email y otra para el de sms. Hay que dirigirse a la pestaña *Actions* y después pulsar *Create Action.* Entonces se rellenan los parámetros tal y como muestra la figura 4.29. Al añadir una nueva operación, aparecerá un cuadro para el título del mensaje y otro para el contenido del mismo. Para rellenar ambos campos, utilizamos unas macros que Zabbix trae ya implementadas y que indican lo siguiente:

- {HOSTNAME} - Indica el nombre del equipo en el que ocurre el problema.
- {TRIGGER.SEVERITY} - Indica la severidad del *trigger.*
- {IPADDRESS} - Es la IP del equipo que presenta el problema.
- {{HOSTNAME}:{TRIGGER.KEY}.last(0)} - Expresará el valor del elemento para el que ha saltado la alarma.

- {TIME} - Expresará la hora a la que ocurrió el problema.
- {DATE} - Expresará la fecha en la que se ocurrió el problema.

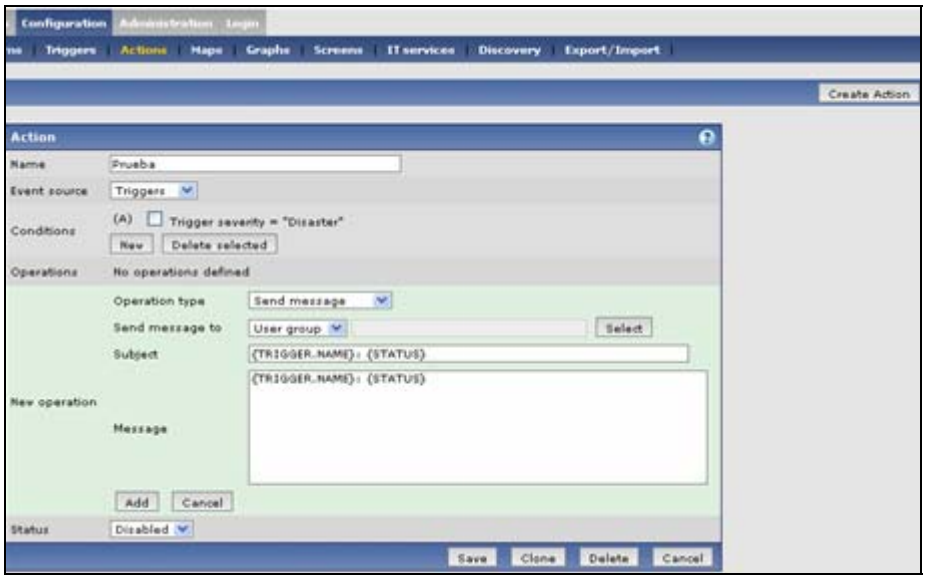


Figura 4.29 - Opciones para la creación de una acción.

4.3.3.5 - Configuración de Zabbix para la recogida de traps

Ya se ha comentado, que algunos dispositivos enviarán notificaciones ante una determinada situación. Ahora se pasa a configurar el gestor, para que sea posible la recogida de estas *traps SNMP*, a través del puerto 10051. Ya se dieron los pasos previos en el apartado de instalación. Los parámetros que aparecen en el script, deben coincidir con los que se van a definir a continuación en el gestor Zabbix.

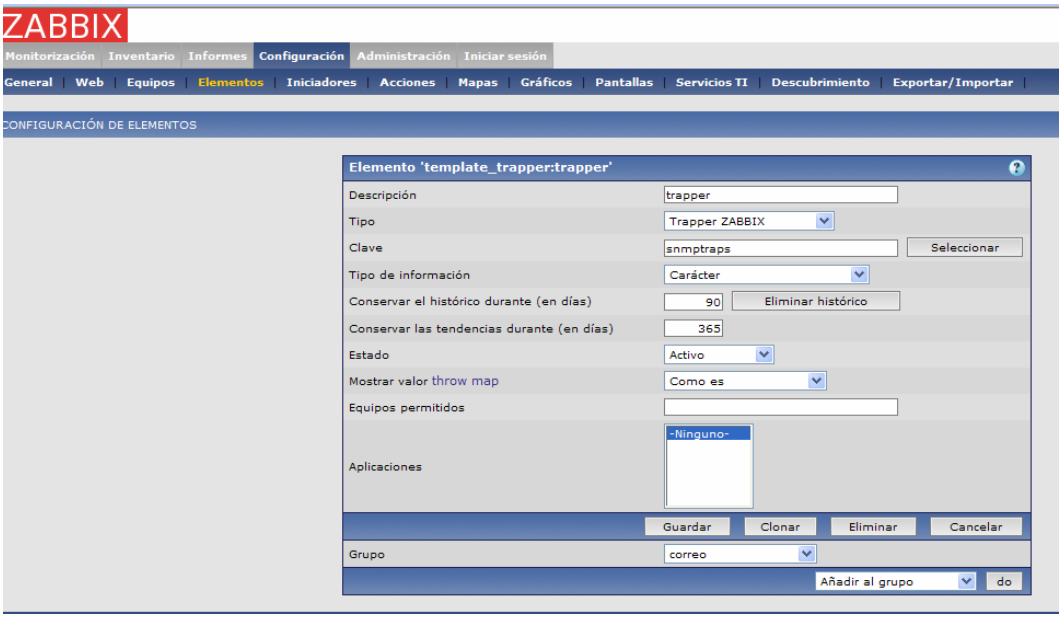
En primer lugar, se crea un elemento o *item* llamado “trapper”. Después una plantilla de nombre “template_trapper” y añadimos a esta el elemento anterior. Para finalizar, se crea un *host* de nombre “traps”. Todos los pasos a seguir se pueden visualizar en la figura 4.30, donde además se cambió el idioma y se comprobó que no traducía todas las opciones existentes. Tras registrar este *host*, se pasa a activar la recogida de *traps*, iniciando el servicio correspondiente en la plataforma Zabbix:

./usr/sbin/snmptrapd start

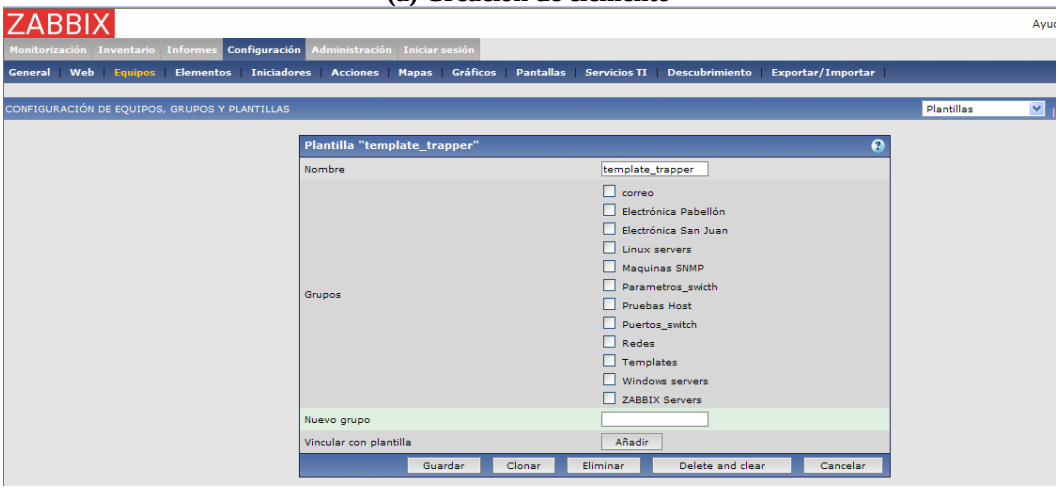
4.3.3.6 - Creación de grupos de usuarios

Zabbix permite la creación de varios tipos de usuario:

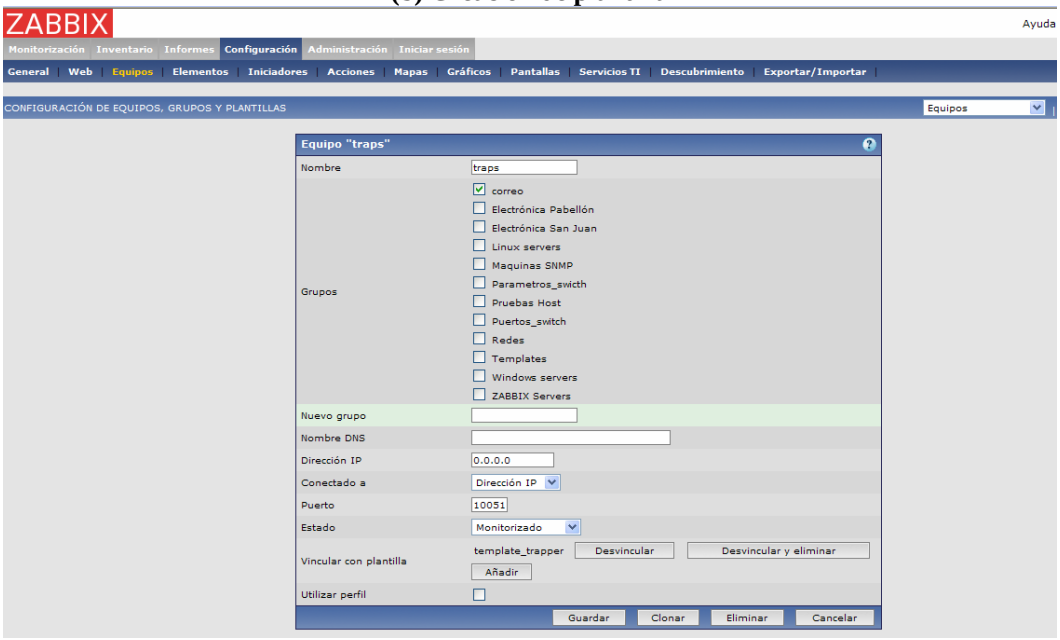
- Usuario *Super Admin*, será el que contenga todos los permisos, por tanto tendrá acceso para monitorizar, configurar y administrar.



(a) Creación de elemento



(b) Creación de plantilla



(c) Definición de equipo

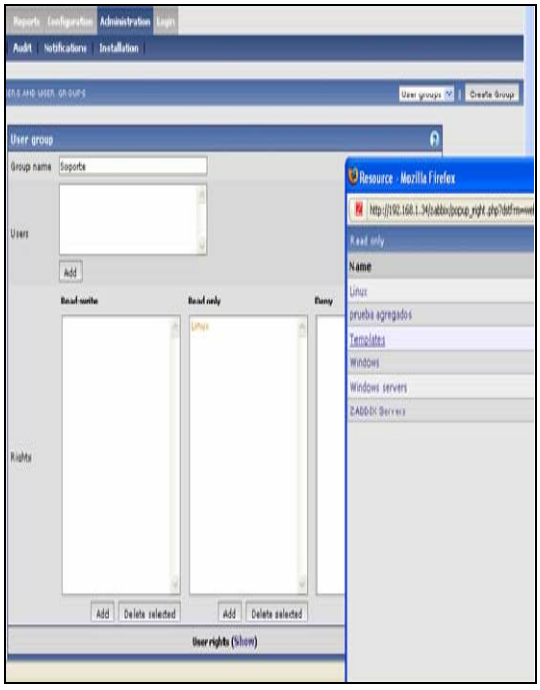
Figura 4.30 - Parámetros a crear para la recepción de traps.

- Usuario *Admin*, puede tener accesos de lectura y escritura en la configuración y acceder a la monitorización, pero el Super Admin será el que conceda estos permisos.
- Usuario *User*, que sólo podrá acceder a la pestaña de monitorización. Además el *Super Admin*, podrá limitar los datos visualizados por este usuario.

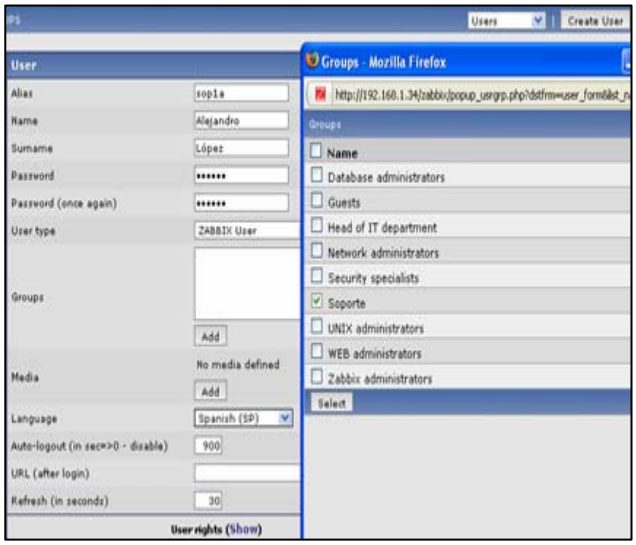
Se pueden crear grupos de usuarios con distintos permisos. Por defecto, aparecen varios grupos de usuarios y varios tipos de usuarios.

En este proyecto, sólo se pretenden utilizar dos grupos de usuarios:

- el de Super Admin, que viene por defecto. Permite realizar todos los cambios necesarios de la configuración y la administración. Será la cuenta asignada al operador de red que se encarga de gestionar el sistema.



(a) Creación del grupo Soporte



(b) Creación de usuario y vinculación con su grupo

Alias	Name	Surname	User type	Groups	Is online?
Admin	Zabbix	Administrator	ZABBIX Super Admin	Zabbix administrators	Yes (Tue, 21 Apr 2009 09:59:16 +0200)
guest	Default	User	ZABBIX User	Guests	No
sop1a	Alejandro	López	ZABBIX User	Soporte	No

(c)Grupos existentes en el proyecto.

Figura 4.30 - Pantalla de creación de usuarios y asignación de permisos.

- el de Soporte, que contendrá a todas las cuentas de los distintos operarios de este grupo de RTVA, para que puedan acceder con permiso de lectura a todos los datos recogidos. Cuando uno de los componentes de este grupo, encargado de controlar el sistema, visualice alguna alarma activa, avisará al operario anterior. Además este último, recibirá los emails o sms que enviará Zabbix ante las alarmas con severidad *disaster*. Para la creación de este grupo, primero de creará el *Group* Soporte con los permisos que se han comentado y luego, cada uno de los usuarios que lo conforman con sus respectivas *password* y vinculándolos a dicho grupo. Las distintas pantallas que se encontrarán en estos pasos, pueden visualizarse en la figura 4.30.

El primer acceso cuando se accede al interfaz es con usuario del grupo *guest*, por tanto, hay que asegurarse de que dicho usuario no tenga ningún tipo de permiso, si no queremos que los datos de nuestro sistema sean conocidos por cualquier persona que pueda acceder a este sistema por interfaz web.

4.3.3.7 - Creación de pantallas

Como ya se comentó al principio del proyecto, la principal finalidad de este es la visualización de forma sencilla de los posibles errores. Las pantallas de Zabbix son sin duda la mejor solución.

Primero se introdujeron en la plataforma las distintas imágenes que se corresponderían con los iconos y los fondos de las pantallas. Esto se realiza en la pestaña *General* de la opción *Configuration*. También desde aquí se importaron los correspondientes fondos, que serían una foto de la delegación y la topología real de esta. Además se añadió un mapa de Andalucía, a fin de que todas las delegaciones aparecieran en este.

En segundo lugar, se procedió a la creación de los mapas de las distintas delegaciones. Para ello, se accedió a la pestaña *Map* de *Configuration* y se creó un nuevo mapa al que se le fueron introduciendo uno por uno, los dispositivos pertenecientes a dicha delegación. También se le asignó como fondo, la foto de la misma. A cada dispositivo se le asignan tres iconos: uno para cuando funciona correctamente, es decir, no ha saltado ninguna de las alarmas que tiene definida; otro, para cuando salta alguna alarma y el último, cuando el estado de esta es desconocido. En el caso de la delegación de Cádiz, los iconos elegidos para cada una de las máquinas, pueden observarse en la figura 4.31.

Después se crean los enlaces entre las máquinas. A cada uno, hay que asignarle un *trigger* de alguna de las máquinas a las que une, que si no salta mostrará el color verde y si salta, el rojo.

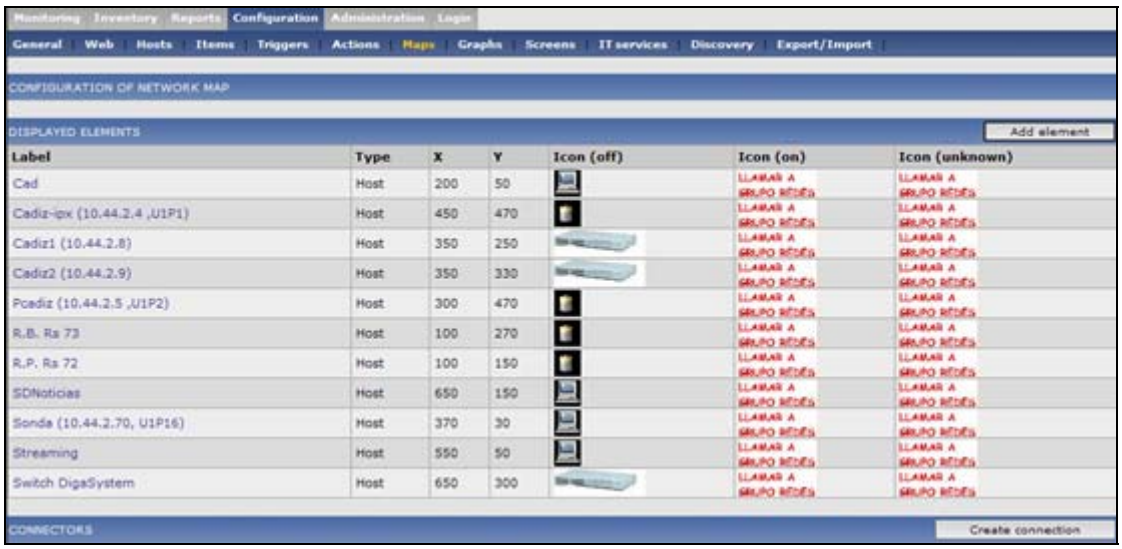


Figura 4.31 - Adición de *hosts* al mapa.

Al ir añadiendo los elementos, debemos asignarles las coordenadas para distribuirlos por el mapa. Los dispositivos a medida que se van añadiendo, van apareciendo en un dibujo cuadriculado que imita al mapa de salida. La figura 4.32 es ejemplo de este dibujo.

Tras tener el mapa listo, se pasa a crear una pantalla con dos columnas y una fila por cada delegación. Como en la figura 4.33 para la delegación de Cádiz, en la primera columna se pondrá el mapa creado y en la segunda, la topología introducida como imagen.

Además se creará otra pantalla con el mapa de Andalucía, donde un icono representará al conjunto de *hosts* de cada delegación, de tal forma que si ocurre un problema, el icono variará. En la figura 4.32, se puede observar que la delegación de Sevilla está teniendo problemas y la de Jaén, Cádiz y Algeciras no.

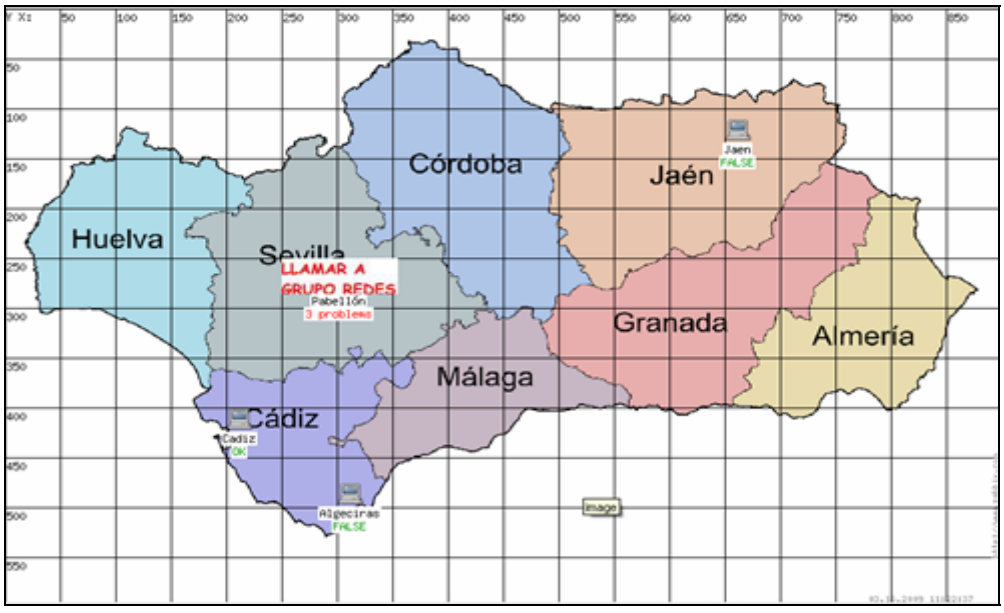


Figura 4.32 - Creación del mapa de Andalucía.

Por último, tras hacer una pantalla para cada una de las delegaciones, se crearán con estas, diapositivas con el fin de que se vayan mostrando cada 5 minutos en una pantalla situada en el CPD.

Figura 4.33 - Pantalla de la delegación de Cádiz.

4.4 - ANÁLISIS DE RESULTADOS

En este apartado, se va a comprobar si los recursos con los que se contaba inicialmente, son suficientes. Asimismo, se comentarán las ventajas e inconvenientes encontradas en el sistema y en el proyecto en general.

4.4.1 - COMPROBACIÓN DE REQUISITOS

Tal y como se comentó anteriormente, la plataforma de Zabbix debe contar con una serie de recursos para su correcto funcionamiento. Estas características dependerán de cómo se haya configurado el sistema. Como este paso ya se ha realizado, procedemos a comprobar si son suficientes los recursos iniciales o se necesitan ampliar.

El manual de Zabbix propone una serie de cálculos, tabla 4.8, para comprobar si el tamaño del disco es suficiente como para almacenar la base de datos del sistema.

Las variables de la tabla tienen los siguientes significados:

- D: Número de días que se guardan los datos. En la mayoría de los casos es 7.
- V: Número de variables monitorizadas. En la pestaña *reports*, opción *Status of Zabbix,* se hace referencia al número de variables monitorizadas. En este caso es de 982.
- I: Intervalo de refresco en segundos. Para las estadísticas, este intervalo siempre será de 30 minutos (ya está expresado en su fórmula). Para los datos es mayoritariamente de 60 segundos.

Parámetro	Fórmula de espacio requerido en disco (en bytes)
Configuración de Zabbix	Tamaño fijo. Aproximadamente 10 MB.
Historial de datos	F1=D*(V/I)*24*3600*B1
Historial de estadísticas	F2=D*(V/1800)*24*3600*B2
Eventos	F3=D*E*24*3600*B3
Espacio total necesario	Configuración + F1+F2+F3

Tabla 4.8 - Cálculo del tamaño de la base de datos.

Para B1, B2 y B3, se pondrá el peor de los casos, es decir, el mayor de los tamaños.

- B1: Número de bytes que ocupa el dato. Se tomará 50.
- B2: Número de bytes que ocupa la estadística. Será igual a 128.
- B3: Número de bytes que ocupa un evento. El peor de los casos, es de 130.
- E: Número de eventos por segundo. Se considera el peor de los casos, que es cuando vale 1.

Teniendo en cuenta todas estas variables, se obtiene que la base de datos de Zabbix ocupa un total de aproximadamente 600 MB. La plataforma cuenta con un disco duro de 80 GB, con lo cual se dispone de suficiente espacio, pues la base de datos ni siquiera ocupa el 2%.

En la tabla 4.2, se hace referencia a las necesidades que requiere el sistema en función del número de *host* a monitorizar. En este caso, el número de dispositivos monitorizados es de 130 aproximadamente y se cuenta con una CPU Intel(R) Xeon(R) 2.4 GHz. Se dispone de una capacidad de CPU muy superior a la necesitada. Pero para asegurarlo, se utilizaron los datos recogidos de la propia plataforma. En la figura 4.32, se hace referencia al % en media de CPU parada (*idle*), utilizada por proceso (*nice*) y utilizada por sistema (*system*). Se detecta que como máximo se utiliza el 40 % de la capacidad total, lo cual indica que satisface la demanda muy por encima.

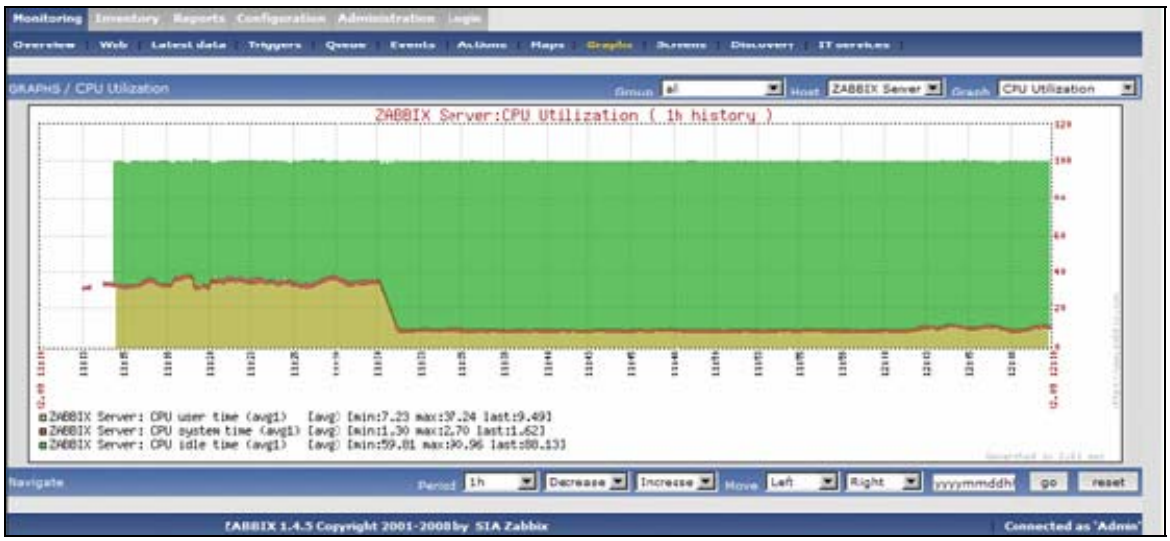


Figura 4.34 - Gráfica de las variables referidas a la CPU.

El único recurso que se observó que estaba colapsado, era la RAM del sistema. Se dispone de un total de 1 GB y la variable que hacía referencia a la memoria libre, rozaba un valor de 20 MB continuamente. Aunque se crearon *triggers* para que saltarán cuando el valor libre disminuyera a 15 MB, se debería realizar una ampliación de dicha memoria a fin de que quede libre aproximadamente un 20 o 30%, es decir, 200 o 300MB, pues eran numerosas las ocasiones en las que el sistema se colapsaba.

4.4.2 - PRIMERAS OBSERVACIONES

Al principio, se definieron un mayor número de variables, pero tras recopilar sus valores se consideró que no daban información interesante y se optó por quitarlas.

También se observó, que varios *triggers* se activaban continuamente sin motivo alguno. Eran los referidos a los bytes transmitidos y recibidos. Entonces, mirando en la topología de la red, se comprobó que esos enlaces estaban duplicados y accediendo a los switches, se detectó que tenían activado el *Spanning Tre*e. Por eso, para los switches que disponían de estos enlaces, se tuvo que definir una condición para que saltara la alarma de estas variables, sólo cuando no se transmitía ni recibía por ninguno de los dos.

Por último, comentar que se debe tener cuidado con el uso del agente Zabbix. Hay que comprobar que la versión de este es compatible con la del gestor, pues si no, no reconocerá a la máquina que lo contiene.