INGENIERÍA DE SERVIDORES (2019-20)
Grado en Ingeniería Informática
Universidad de Granada

Práctica 3: Monitorización y "profiling" de servidores

Alonso Bueno Herrero

8 de agosto de 2020

Índice

Referencias

[1] `https://es.wikipedia.org/wiki/Zabbix`

[2] `https://www.zabbix.com/`

[3] Página oficial de Ansible: `https://www.ansible.com/`

[4] Entrada de Wikipedia para Ansible en Español: `https://es.wikipedia.org/wiki/Ansible_(software)`

1. Introducción

El guión de prácticas asociado a este documento contiene una información interesante, que además es materia «de examen» sobre los monitores disponibles, su clasificación, además de algunos temas muy interesantes sobre Automatización y Profiling del sistema.

En el presente documento nos centraremos en resolver las tareas evaluables propuestas para esta práctica: monitorización con Zabbix y automatización básica de tareas con Ansible.

2. Sesión 1ª: Automatización básica de tareas con Ansible

2.1. Introducción a la orquestación con Ansible

En el guión oficial de esta práctica se indicaban varias herramientas para la automatización de tareas. Una de ellas, que actúa a nivel de plataforma es Ansible.

Ansible es un software que automatiza el **aprovisionamiento** de *software*, la gestión de configuraciones y el despliegue de aplicaciones. Está categorizado como una herramienta de **orquestación**, muy útil para los administradores de sistema y DevOps[1].

En otras palabras, Ansible permite a los DevOps gestionar sus servidores, configuraciones y aplicaciones de forma sencilla, robusta y paralela

Tal y como se indica en (4), «[Ansible] gestiona nodos a través de SSH y no requiere ningún software remoto adicional (excepto Python 2.4 o posterior para instalarlo)».

En definitiva, hablamos de una herramienta para:

- automatizar tareas;
- provisionar, y
- orquestar servicios (a esto nos dedicaremos a continuación).

2.2. Pruebas sobre nuestras máquinas virtuales

Para realizar las pruebas podemos usar las dos máquinas virtuales que tenemos: Ubuntu y centOS. Basta con ejecutar los comandos en una máquina apropiadamente para que se ejecute en la otra, logrando la comunicación entre ambos.

Para la prueba no hace falta realmente «logearse» en la máquina virtual que reciba las órdenes de ejecución, sólo donde vayamos a lanzar los comandos.

2.2.1. Instalar Ansible en Ubuntu

Basta con ejecutar el comando:

```
sudo apt install ansible
```

2.2.2. Creación de grupos *receptores* de tareas

En el fichero `/etc/ansible/hosts` se pueden definir los *receptores* y grupos de los mismos de las tareas que vayamos a lanzar con Ansible.

Para definir un grupo de destinos basta con indicar un nombre de referencia entre corchetes, y debajo, en cada línea, el formato `IP:PUERTO` del host destino. Un formato válido para un grupo básico sería:

[1] DevOps es una metodología de desarrollo de *software* basada en la integración entre desarrolladores *software* y administradores de sistemas

```
[UbuntuServer]
192.168.56.105:22022
[centOSServer]
192.168.56.110:22022
```

en el caso de que quisiéramos un grupo para cada servidor (un grupo para el servidor *Ubuntu* y otro grupo para el de *centOS*).

2.2.3. Lanzar tareas desde Linux (anfitrión o máquina virtual)

Mandar la orden de hacer `ping` a todos los grupos de servidores que hayamos definido en el fichero `/etc/ansible/hosts`:

```
ansible all -m ping  # all = mandar a todos los grupos
```

Es posible que nos dé un error, debido a problemas con la clave público-privada. Podemos solucionar este problema generando una clave con la orden

```
ssh-keygen
```

y se generan las claves (pública y privada)[2] en el fichero `.ssh/` (carpeta oculta en el directorio actual).

Ahora nos falta enviar la clave a los destinatarios uno a uno con

```
ssh-copy-id IP -p PUERTO
```

Si volvemos a ejecutar el comando de *Ansible*, puede que nos vuelva a dar un error relacionado con Python. Lo podemos solucionar instalándolo con:

```
sudo apt install python      # suponiendo que estamos en Ubuntu
apt update                   # actualizar repositorios
```

Ahora volvemos a ejecutar el comando (ver figura).

2.2.4. Lanzar tareas desde nuestra máquina nativa Windows

Basta con ejecutar el comando:

```
ansible all -a "ls /" -m command
```

si queremos enviar esa orden a **todos** los grupos, o

```
ansible NOMBRE_GRUPO "ls /" -m command
```

3. Sesión 2ª: Monitorización con Zabbix

3.1. Objetivos y presentación.

En esta sección se presenta un guión sobre la instalación y configuración del monitor Zabbix en Ubuntu y centOS, incluyendo un front-end para poder acceder a la monitorización a través de un navegador. El servidor Ubuntu será el protagonista de la coordinación de las tareas de monitorización de ambos sistemas operativos (a sí mismo y a centOS), de tal forma que:

[2] Refresca los conocimientos sobre criptografía necesarios para las prácticas en el documento asociado a la Práctica 2 (elaboración propia).

- En Ubuntu Server instalaremos el **agente** (tareas a "bajo nivel" de recopilación de datos), el *front-end* (para la monitorización) y el **servidor** (que controla la monitorización de ambos servidores).
- En centOS instalaremos únicamente el **agente**, para recopilar datos que serán luego usados y formateados por el servidor.

A lo largo del proceso veremos algunas capturas sobre los comandos ejecutados para la instalación de los diversos componentes.

3.1.1. Pero, ¿qué es exactamente Zabbix?

Zabbix es un Sistema de Monitorización de Redes creado por Alexei Vladishev. Está diseñado para monitorizar y registrar el estado de varios servicios de red, Servidores, y hardware de red.

Usa MySQL, PostgreSQL, SQLite, Oracle o IBM DB2 como base de datos. Su backend está escrito en C y el frontend web está escrito en PHP. Zabbix ofrece varias opciones de monitorización:

- Chequeos simples que pueden verificar la disponibilidad y el nivel de respuesta de servicios estándar como SMTP o HTTP sin necesidad de instalar ningún software sobre el host monitorizado.
- Un agente Zabbix puede también ser instalado sobre máquinas UNIX y Windows para monitorizar estadísticas como carga de CPU, utilización de red, espacio en disco, etc.
- Como alternativa a instalar el agente sobre los host, Zabbix incluye soporte para monitorizar vía protocolos SNMP, TCP y ICMP, como también sobre IPMI, JMX, SSH, telnet y usando parámetros de configuración personalizados. Zabbix soporta una variedad de mecanismos de notificación en tiempo real, incluyendo XMPP.

Lanzado sobre los términos de la versión 2 de la GNU General Public License, Zabbix es Software Libre.

Puede encontrar más información en (1) (de donde se ha extraído el contenido de este apartado) y (2).

3.2. Instalación de Zabbix en Ubuntu Server

Vamos a ver ahora los pasos que tenemos que seguir para configurar todo el andamiaje de las tareas de monitorización que vamos a realizar en esta sesión. Estos pasos se agrupan en tres secciones diferentes. **Se recomienda que se siga el orden propuesto a continuación para poder realizar la tareas con éxito.**

3.2.1. Instalar los repositorios, el servidor y el front-end

Vamos a instalar la arquitectura necesaria de Zabbix en Ubuntu, en función del esquema conceptual que hemos visto antes.

1. Para esta tarea, simplemente vamos a la página del manual donde indica «4. Installation» y vemos que para Ubuntu Server 16.04 nos da (tras lectura detenida de la página) los siguientes primeros comandos:

```
wget https://repo.zabbix.com/zabbix/3.4/ubuntu/pool/main/z/...
  ...zabbix-release/zabbix-release_3.4-1+xenial_all.deb
dpkg -i zabbix-release_3.4-1+xenial_all.deb
```

   a) Y el resultado que se muestra tras ejecutar `wget` se muestra en la figura 1.

   b) Y ahora vamos a ejecutar el comando `dpkg`. El resultado se muestra en la figura 2.

Figura 1: Resultado de ejecutar `wget` en Ubuntu.

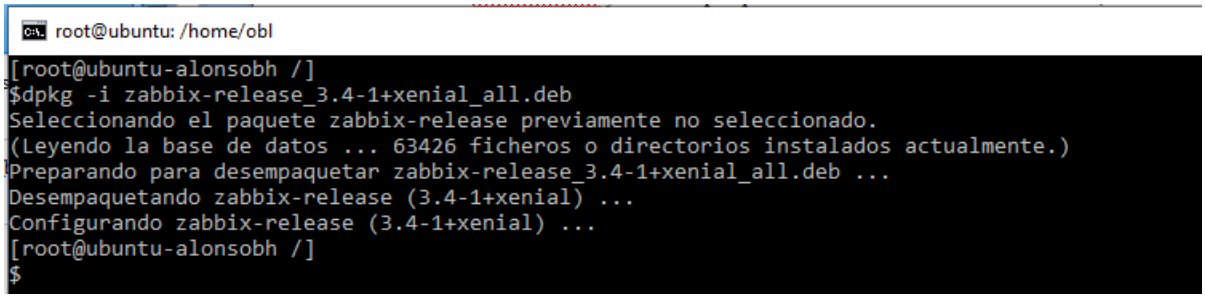

Figura 2: Resultado de ejecutar `dpkg` en Ubuntu.

2. Ahora actualizamos los repositorios con la orden

```
apt update
```

3. Ahora, para instalar el servidor de Zabbix con soporte paraMySQL ejecutamos:

```
apt install zabbix-server-mysql
```

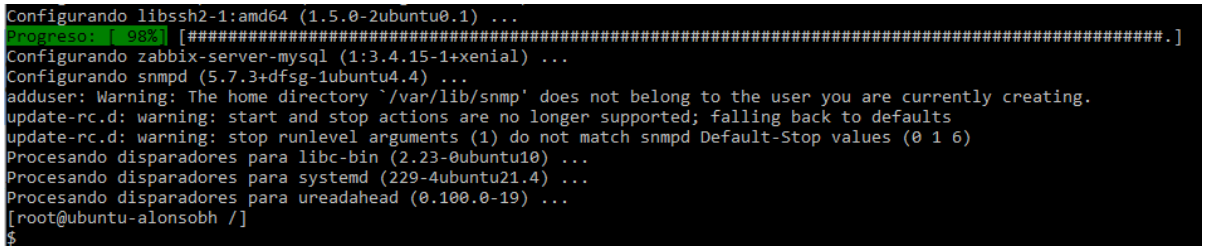

Figura 3: Final de la ejecución del comando anterior.

4. El *front-end* se instalará con un comando similar:

```
apt install zabbix-frontend-php
```

5. A los demonios *server* (el nuestro) y *proxy* les hace falta una base de datos para poder manejar los datos. La podemos crear usando el sistema gestor de bases de datos (SGBD) **MySQL** que teníamos instalado de la práctica anterior:

```
mysql -u root -p <password>
```

Ahora, en el terminal de SQL ejecutamos:

```
mysql> create database zabbix character set utf8 collate utf8_bin;
mysql> grant all privileges on zabbix.* to zabbix@localhost
       identified by 'practicas,ISE';
mysql> quit;
```

6. **Importar el esquema inicial:** Para ello usamos la orden:

```
zcat /usr/share/doc/zabbix-server-mysql/create.sql.gz | mysql -u zabbix
  -p zabbix
```

Este proceso era largo, y el resultado (correcto) se muestra en la figura 4.

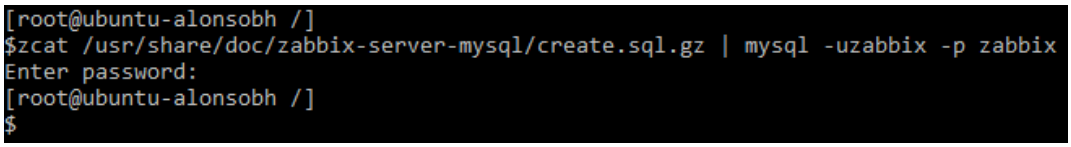

Figura 4: Volcado de pantalla de la importación del esquema.

7. Configurando la base de datos. Editemos ahora el fichero `zabbix_server.conf` para usar la base de datos creada. Por ejemplo, el contenido que añadamos puede ser (descomentando la línea oportuna) el de la figura 5.

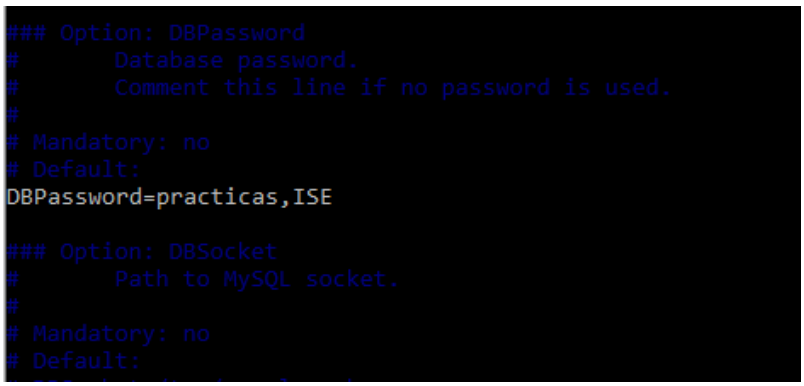

Figura 5: Estableciendo la clave de la base de datos del server Zabbix.

8. En el campo **DBPassword** usaremos la clave para la base de datos de Zabbix en MySQL: `practicas,ISE`.

9. Ahora sólo nos queda arrancar el servidor Zabbix con las órdenes que se ejecutan en la figura 6.

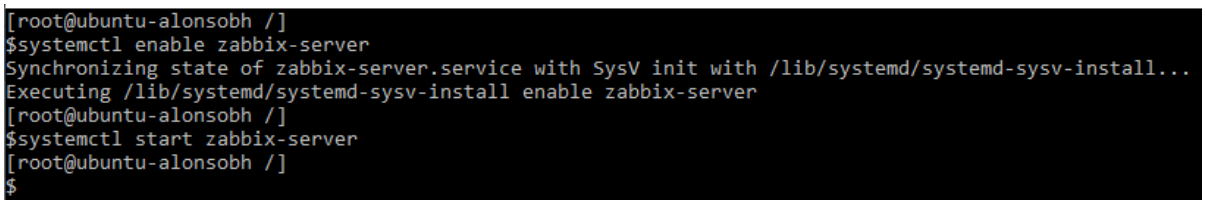

Figura 6: Arrancando el servidor Zabbix (en Ubuntu).

10. Instalación del ***front-end***. El archivo de configuración `/etc/apache2/conf-enabled/zabbix.conf` contiene los parámetros necesarios para la configuración del *front-end* de Zabbix. Algunos de esos

parámetros son de PHP y es necesarios retocarlos, fundamentalmente la zona horaria. Lo vemos en la figura 7.

```
# Define /zabbix alias, this is the default
<IfModule mod_alias.c>
    Alias /zabbix /usr/share/zabbix
</IfModule>

<Directory "/usr/share/zabbix">
    Options FollowSymLinks
    AllowOverride None
    Order allow,deny
    Allow from all

    <IfModule mod_php5.c>
        php_value max_execution_time 300
        php_value memory_limit 128M
        php_value post_max_size 16M
        php_value upload_max_filesize 2M
        php_value max_input_time 300
        php_value max_input_vars 10000
        php_value always_populate_raw_post_data -1
        php_value date.timezone Europe/Madrid
    </IfModule>
    <IfModule mod_php7.c>
        php_value max_execution_time 300
        php_value memory_limit 128M
        php_value post_max_size 16M
        php_value upload_max_filesize 2M
        php_value max_input_time 300
        php_value max_input_vars 10000
        php_value always_populate_raw_post_data -1
        php_value date.timezone Europe/Madrid
    </IfModule>
</Directory>
```

Figura 7: Volcado de `/etc/apache2/conf-enabled/zabbix.conf` tras cambiar la zona horaria.

11. Configuramos el front-end:

    a) Activamos el puerto 80 con la orden `ufw allow 80/tcp`.

    b) Nos conectamos desde el navegador de la máquina nativa al servidor Ubuntu y accedemos al fichero zabbix mediante la dirección siguiente en la barra de búsqueda: **192.168.56.105/zabbix**. En el navegador tiene que aparecer algo similar a la figura 8.

    c) Y nos saldrá la pantalla de inicio del instalador. Pulsamos en el botón «Next step»

    d) Vamos dejando los parámetros por defecto en cada ventana pulsando en «Next step», y ponemos, al llegar el caso, un nombre al servidor (es opcional).

    e) Finalmente, accedemos con las credenciales por defecto de Zabbix en la intranet: `Admin / zabbix`. Podemos ver la interfaz que debemos encontrarnos en la figura 10.

    Por ahora lo dejamos aquí para el front-end.

12. Ahora instalamos e iniciamos el agente en Ubuntu con las órdenes siguientes:

```
apt install zabbix-agent          # Instalar el agente en Ubuntu.
service zabbix-agent start        # Para iniciar el servicio.
```

    Y el resultado de estas órdenes lo podemos ver en la figura 9.

3.3. Instalación de Zabbix en centOS

Los pasos son mucho más cortos en este caso, pues **sólo tenemos que cargar el repositorio e instalar el agente**. La página del manual en este caso tendrá la misma página padre, pero accedemos ahora a la instalación en RedHat («1 Red Hat Enterprise Linux/CentOS»).

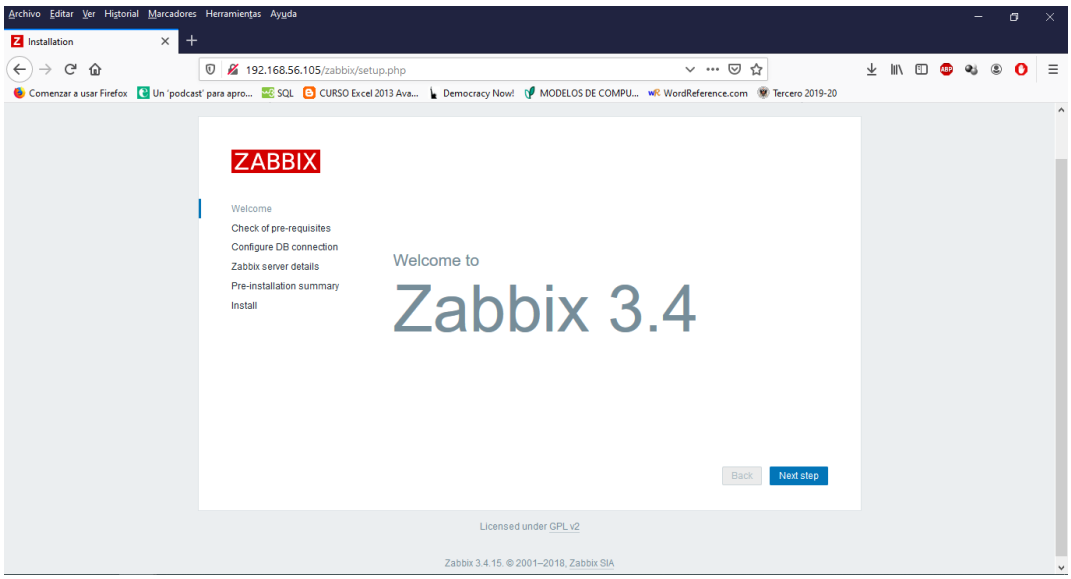

Figura 8: Página de inicio al entrar al *front-end*.

1. Para instalar el repositorio, en esta página nos proporcionan la orden:

```
rpm -ivh https://repo.zabbix.com/zabbix/3.4/rhel/7/x86_64/...
    ...zabbix-release-3.4-2.el7.noarch.rpm
```

2. Y a continuación instalamos el agente con los comandos habituales (ver figura 11).

```
yum install zabbix-agent
service zabbix-agent start
```

3.4. Configuración de parámetros en el *front-end*

3.4.1. Creando los hosts a monitorizar

De nuevo en el *front-end* que teníamos abierto, vamos a crear los dos hosts que vamos a monitorizar. Para ello vamos a la pestaña Hosts de la parte superior, vamos a la opción «Create Hosts» y nos aparecerá el menú siguiente (ver figura 12):

Ponemos el nombre, tanto para Ubuntu como para centOS y ajustamos la IP: `192.168.56.105` y `192.168.56.110`, respectivamente. Dejamos el puerto por defecto (`10050`) y deberemos activarlo para el cortafuegos en los dos servidores (comandos `ufw` en Ubuntu y `firewall-cmd` y `semanage` respectivamente).

Después, reiniciar los servicios como siempre se ha realizado. Además, podemos añadir cada servidor creado al grupo *Linux Servers*, tal y como aparece en la figura 12.

3.4.2. Tareas a monitorizar: HTTP y SSH en Ubuntu y centOS

Lo que tenemos que hacer, una vez que hemos creado los dos host, es configurar la monitorización de dos servicios: HTTP y SSH.

El proceso es muy sencillo, y se basa en el uso de «Templates», que son la representación del elemento del sistema que será medido[3]. Para crear un Template accedemos a cada *host* de la lista, seleccionamos «Templates» y nos saldrá el menú para añadir/crear templates asociadas a ese host (ver figura 13).

[3]Realmente, son los «Ítems» los que representan ese elemento a medir. Nuestros Templates tienen cada uno un «Ítem».

```
$apt install zabbix-agent
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes NUEVOS:
  zabbix-agent
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 168 no actualizados.
Se necesita descargar 165 kB de archivos.
Se utilizarán 638 kB de espacio de disco adicional después de esta operación.
Des:1 http://repo.zabbix.com/zabbix/3.4/ubuntu xenial/main amd64 zabbix-agent amd64 1:3.4.15-1+xenial [165 kB]
Descargados 165 kB en 1s (155 kB/s)
Seleccionando el paquete zabbix-agent previamente no seleccionado.
(Leyendo la base de datos ... 64884 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../zabbix-agent_1%3a3.4.15-1+xenial_amd64.deb ...
Desempaquetando zabbix-agent (1:3.4.15-1+xenial) ...
Procesando disparadores para man-db (2.7.5-1) ...
Procesando disparadores para systemd (229-4ubuntu21.4) ...
Procesando disparadores para ureadahead (0.100.0-19) ...
Configurando zabbix-agent (1:3.4.15-1+xenial) ...
Procesando disparadores para systemd (229-4ubuntu21.4) ...########################################.............]
Procesando disparadores para ureadahead (0.100.0-19) ...
[root@ubuntu-alonsobh /]
$systemctl enable zabbix-agent
Synchronizing state of zabbix-agent.service with SysV init with /lib/systemd/systemd-sysv-install...
Executing /lib/systemd/systemd-sysv-install enable zabbix-agent
[root@ubuntu-alonsobh /]
$systemctl start zabbix-agent
[root@ubuntu-alonsobh /]
$systemctl status zabbix-server
● zabbix-server.service - Zabbix Server
   Loaded: loaded (/lib/systemd/system/zabbix-server.service; enabled; vendor preset: enabled)
   Active: active (running) since dom 2019-11-17 01:27:49 CET; 32min ago
```

Figura 9: Instalando el agente Zabbix, activándolo y viendo su estado.

Entramos en «Select», se abre una nueva ventana y ahí seleccionamos los tipos de plantilla y damos a «Update» para salir y que se creen los Templates.

Repetimos esto para centOS y Ubuntu, y tendremos la monitorización casi hecha. Hay un problema que surge con SSH, y es que hay que añadir en el frontend el puerto 22022 al template de este servicio. Bastará con hacerlo sobre el de uno de los dos SO para que afecte a todos. El proceso es:

> Ir al menú «Hosts» → seleccionar la opción «Items» de alguna fila → ir a «SSH service is running» → en el campo «Key» hay que escribir `net.tcp.service[ssh,,22022]` → pinchar en «**Update**».

3.4.3. Añadiendo gráficos a la monitorización de HTTP y SSH

Visualizar la monitorización mediante gráficas es muy útil para el análisis de la información extraída por el monitor. Vamos a crear gráficas para monitorizar SSH y HTTP en ambos Sistemas Operativos. El proceso se hará para centOS y se repetirá exactamente igual para Ubuntu.

El proceso es:

> Ir a «Hosts» → seleccionar la columna «Graphs» de la fila del *host* que corresponda → seleccionamos «Create graph» → ponemos nombre (significativo) al gráfico y al final, añadimos el ítem de ese *host* que queremos evaluar → Seleccionamos «Add» y ya lo hemos creado.
>
> **Para visualizar los gráficos**, vamos a «Monitoring» → «Graphs» y seleccionamos el que queremos ver.

En la figura 14 se muestra un ejemplo del gráfico del servicio SSH sobre centOS, donde hemos experimentado una caída del servicio reciente con su posterior recuperación (gráfica en V). Como podemos observar, al haber ajustado la hora del servidor (en Ubuntu, claro, que es donde está el servidor Zabbix) nos aparecen las fechas correctas en el gráfico (hemos ajustado el zoom del gráfico a 5m):

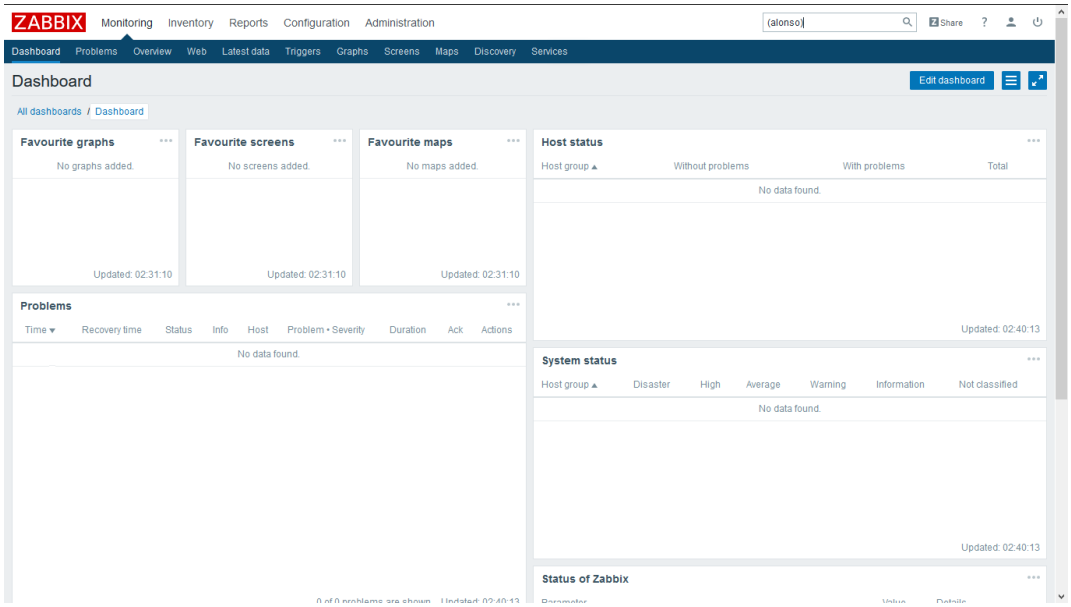

Figura 10: Pantalla principal de la interfaz del *front-end* en el navegador. Por ahora no tenemos nada configurado.

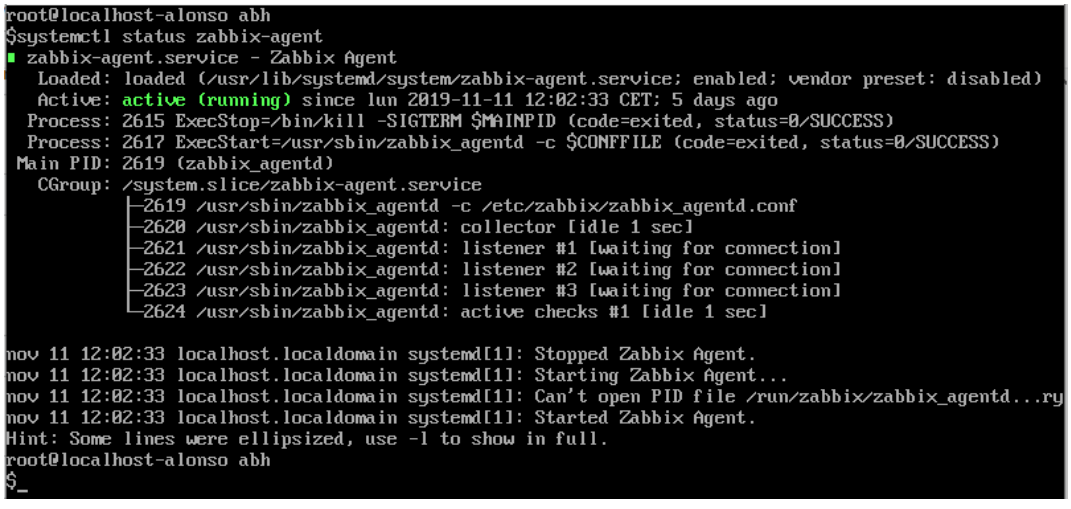

Figura 11: Instalando el agente en centOS.

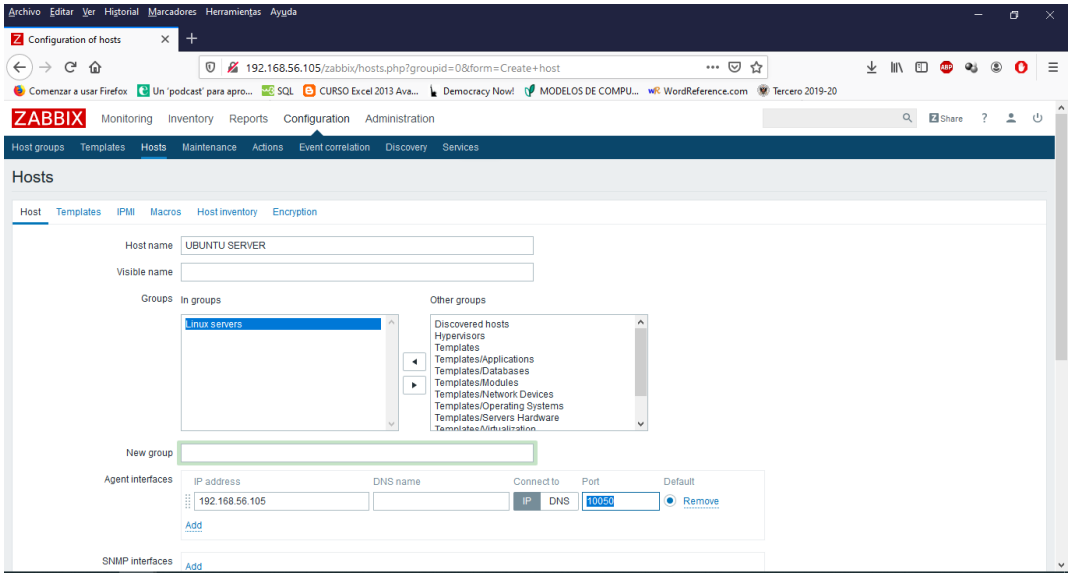

Figura 12: Creando los hosts.

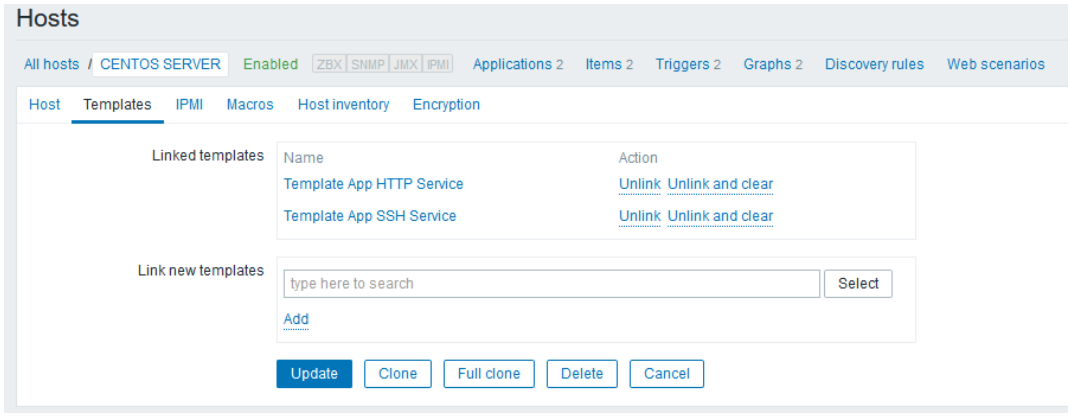

Figura 13: Creando «Templates» para la monitorización de los servicios HTTP y SSH.

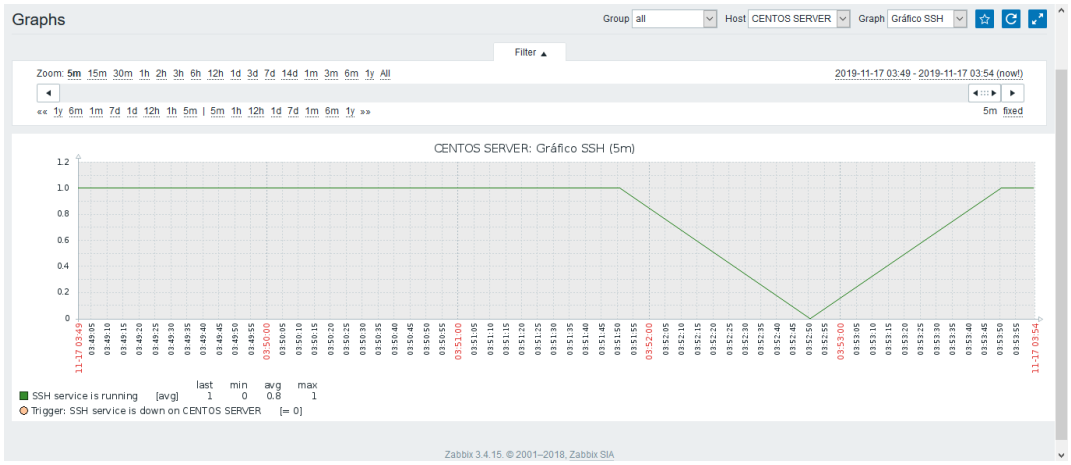

Figura 14: Monitorización con gráficos para SSH en centOS.