Antonio Javier Rodríguez Romero. ISE. B2

ISE. Prácticas B2: Simulación de Carga de Trabajo y Monitorización

Todos los ejercicios y pruebas se pueden hacer tanto en Windows como en Linux con VirtualBox instalado, aunque desde mi experiencia es mejor desde Ubuntu.

1. Docker. “Hello World”

Primero instalamos docker: buscar por internet “Instalar docker linux” y no suele dar fallos. Si se hace desde Windows con wsl, busca específicamente para eso ya que varían ciertas cosas.
Funcionamiento docker:
- docker pull <imagen>: descarga del repositorio la imagen indicada.
- docker run <imagen>: crea un contenedor que ejecutará la imagen indicada, si no está descargada la descarga.
- docker compose up: cuando en el directorio existe un archivo docker-compose.yml, toma la configuración indicada en este y ejecuta varias imágenes en un contenedor.

Para el “hello world” ejecutar simplemente: **docker run hello-world**

2. Benchmark con Phoronix

Phoronix se puede instalar tanto en docker como directamente en el SO anfitrión. Aunque con docker desktop he conseguido que funcione, es más fácil y dará menos fallos si lo instalas en el host.
El funcionamiento de esta herramienta es: descargar del repositorio benchmarks que testearán distintos apartados o programas del sistema.
Para ejecución desde terminal con el programa instalado en nuestro host, el comando será: **phornix-test-suite…**
- install
- benchmark
- run
- list-all-tests
- result-file-to-<formato>

Yo he ejecutado el test **pts/stress-ng**.
Los resultados se guardarán en ~/.phoronix-test-suite/test-results con el nombre que se le haya indicado.

3. Carga Http con ab

Necesitamos una MV con Rocky (Rocky es lo que te pide). Instalamos httpd o nginx en función de la utilidad que usemos para montar el servidor http.
Con Apache(httpd) si queremos poner un index.html para que no salga la página por defecto, esto se hará en */var/*www/html/
Si queremos hacerlo con nginx: */usr/*share/nginx/html/

Puede que después de iniciar el servicio httpd o nginx no nos deje acceder al servidor desde nuestro host (<ip>:80). Instala y activa firewalld y ejecuta firewall-cmd –permanent –add-service=http y recarga con firewall-cmd –reload.

Con el servidor montado instalamos el paquete que tiene ab en nuestro host: apache2-utils.

El uso de este comando es fácil:
ab [opciones] <ip>:80/ (Importante el último /)
Las principales opciones:
-n número de request
-c concurrencia
-t timelimit en segundos
-w enseñar los resultados en html tables
-g guardar resultados en formato gnuplot
-e guardar resultados en formato csv

Es importante también entender los resultados que te muestra en terminal el comando.
Te enseña los tiempos de ejecución de todas las request como conjunto y después desglosa cada una por separado.

4. Carga Http con Jmeter

Para tests más complejos se usa Jmeter. Haciendo un clone del repositorio indicado en el guión, tendremos tanto las herramientas para ejecutar jMeter, que lo ejecutaré desde mi host, como los archivos y el docker compose para simular una web en el servidor Rocky sobre la que haremos el testing.

Para ejecutar en terminal solo:
./jmeter -n -t <archivoTest.jmx> -l <archivoResultado.jtl>

5. Stress + Top

stress es una herramienta de carga del sistema que puede generar threads que carguen o cpu (--cpu N), entradas y salidas (--io N), memoria virtual (--vm) o almacenamiento (--hdd). Su uso junto con top sirve para ver el estado del sistema al someterlo a ciertas condiciones.

En rocky linus no existe stress, o por lo menos yo no he sido capaz de encontrarlo. Su alternativa es una versión posterior, stress-ng.

6. Uso de cron

Primero lo instalamos. Veremos que existe un archivo por defecto etc/cron.d/dailyjobs que ejecutará los archivos que haya en /etc/cron.daly cada día, en /etc/cron.weekly cada semana y en /etc/cron.monthly cada mes.

Si necesitamos especificar más aún cuando queremos que se hagan las tareas: **crontab -e**
Si es la primera vez que ejecutamos nos saldrá un archivo vacío o solo con comentarios para editar. Aquí añadiremos qué queremos que se ejecute y cada cuanto en el formato:
<minuto> <hora> <dia(mes)> <mes> <dia(semana)> <comando>.
Donde pongamos un * significa que da igual, si ponemos */n será que lo haga cada 5 unidades de tiempo correspondientes, si ponemos a-b en todos los valores entre a y b, si ponemos a,b,c en los valores a,b y c.
En el ejercicio que nos piden, habría que añadir
***/5 * * * * logger -t "ISE" "<Iniciales>: $(date) - $(uptime)"**

La utilidad logger lo único que hace es hacer un registro en el log del sistema de lo que le indiquemos con la tag indicada justo después de -t.

7. Logs de arranque del sistema

La monitorización pasa también por revisar los logs de los diferentes procesos o del mismo sistema. Se guardan en **/var/logs** y algunos estarán guardados en texto plano o en binario, lo que permite más flexibilidad de representación. Estos se consultarán con herramientas como btmp o wtmp

Para que no se hagan demasiado largos, existen herramientas de rotación de logs de manera que tengamos varios archivos .log refiriendose al mismo proceso o sistema.
Logrotate se encarga de realizar esta labor.

Todos los logs del sistema pasan por **systemd-journald**, normalmente de forma volatil en cada arranque del sistema, pero creando el directorio ***/var/*log/journal** se conservarán.

Para consultar de forma fácil el log del sistema usamos **journalctl** con:
-b desde el último reboot
-b -N el enésimo reboot pasado.
--list-boots consultar ids d cada reboot
--since desde
--until hasta
Para expresar fechas y horas hay gran flexibilidad. Las expresiones más utiles son:
- “N hours/minutes/seconds/days… ago”
- “yyyy-mm-dd hh:mm:ss”
- yesterday
…

Podemos consultar fácilmente también según la unidad que generase el log con **-u**, por PID con **_PID=** o **_UID= _GID=**. Con -F vemos todas las variables posibles.

Por último podemos filtrar también por la prioridad con **-p**. Las prioridades son:

- **0**: emerg
- **1**: alert
- **2**: crit
- **3**: err
- **4**: warning
- **5**: notice
- **6**: info
- **7**: debug

y según la que indiquemos nos enseñará los logs con esa prioridad y superior.
En el ejercicio evaluable, la solución será: **journalctl -b -p warning(o 4)**

Antonio Javier Rodríguez Romero. ISE. B2

8. Monitorización con Grafana+Prometheus

Para monitorizar un servidor utilizaremos Prometheus, que recoje datos pero no permite gran manejo de ellos, y Grafana, que leerá los datos de Prometheus y nos permitirá mostrarlos en un formato más entendible y con la posiblidad de poner alertas sobre ciertas mediciones.

Para posibilitar que prometheus lea los datos, tendremos que instalar en el servidor a estudiar una herramienta que haga estos visibles, en este caso Node Exporter, el cual configuraremos como servicio en nuestro servidor y en la configuración de prometheus lo definiremos como target.

Instalamos grafana y prometheus en nuestro host y node exporter en la mv rocky.