|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | Escuela Politécnica Superior  Ingeniería Informática  Prácticas de Sistemas Informáticos 2 | | | |
| **Grupo** | **2401** | **Práctica** | 2 | **Fecha** | 17/04/2021 |
| **Alumno/a** | | De las Heras Moreno, Martín | | | |
| **Alumno/a** | | Valderrábano Zamorano, Santiago Manuel | | | |

# Práctica 2: Rendimiento

## Ejercicio número 1:

Para realizar este ejercicio, hemos instalado JMeter, copiado las aplicaciones P1-base, P1-ws y P1-ejb (servidor remoto) y las hemos desplegado.

Después, hemos ido siguiendo los pasos que se indican en el enunciado para construir los planes de pruebas indicados. Todo esto se ha guardado en el fichero ***P2.jmx*** el cual se adjunta en el fichero de entrega.

Añadido a esto comentar que, para los siguientes ejercicios, como se recomienda en el enunciado, hemos añadido un árbol de resultados que desactivaremos tras comprobar que todo funciona correctamente.

## Ejercicio número 2:

Al realizar este ejercicio, debido a la situación actual y que no tenemos acceso a los ordenadores de los laboratorios, hemos utilizado un único PC en el que desplegamos ambas máquinas virtuales.

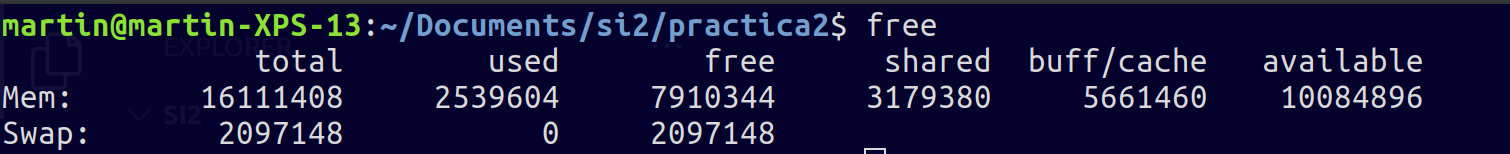
Antes de desplegar las aplicaciones que se van a utilizar, hemos revisado y modificado los siguientes parámetros:

* P1-base:
  + build.properties:
    - as.host=**10.1.7.2**
  + postgresql.properties:
    - db.host=**10.1.7.1**
    - db.client.host=**10.1.7.2**
* P1-ws:
  + build.properties:
    - as.host.client=**10.1.7.2**
    - as.host.server=**10.1.7.1**
  + postgresql.properties:
    - db.host=**10.1.7.1**
    - db.client.host=**10.1.7.1**
* P1-ejb-servidor-remoto:
  + build.properties:
    - as.host.client=**10.1.7.1**
    - as.host.server=**10.1.7.1**
  + postgresql.properties:
    - db.host=**10.1.7.1**
    - db.client.host=**10.1.7.1**
* P1-ejb-cliente-remoto:
  + build.properties:
    - as.host.client=**10.1.7.2**
  + postgresql.properties:
    - db.host=**10.1.7.1**
    - db.client.host=**10.1.7.1**
  + glassfish-web.xml
    - **10.1.7.1**

Tras realizar estos cambios, desplegamos las aplicaciones y hemos hecho un pago de prueba en cada una para probar que funcionan correctamente.

A continuación, se nos pide que mostremos las salidas de los comandos ***free*** y ***nmon*** (tras pulsar la tecla “M”) ejecutados tanto en el PC como en las máquinas virtuales.

***free(PC)***

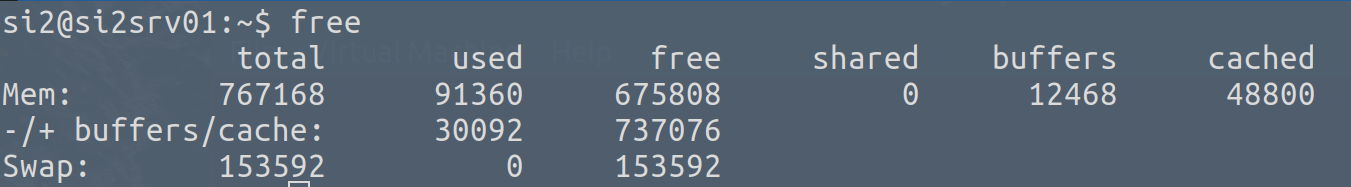


***nmon(PC)***

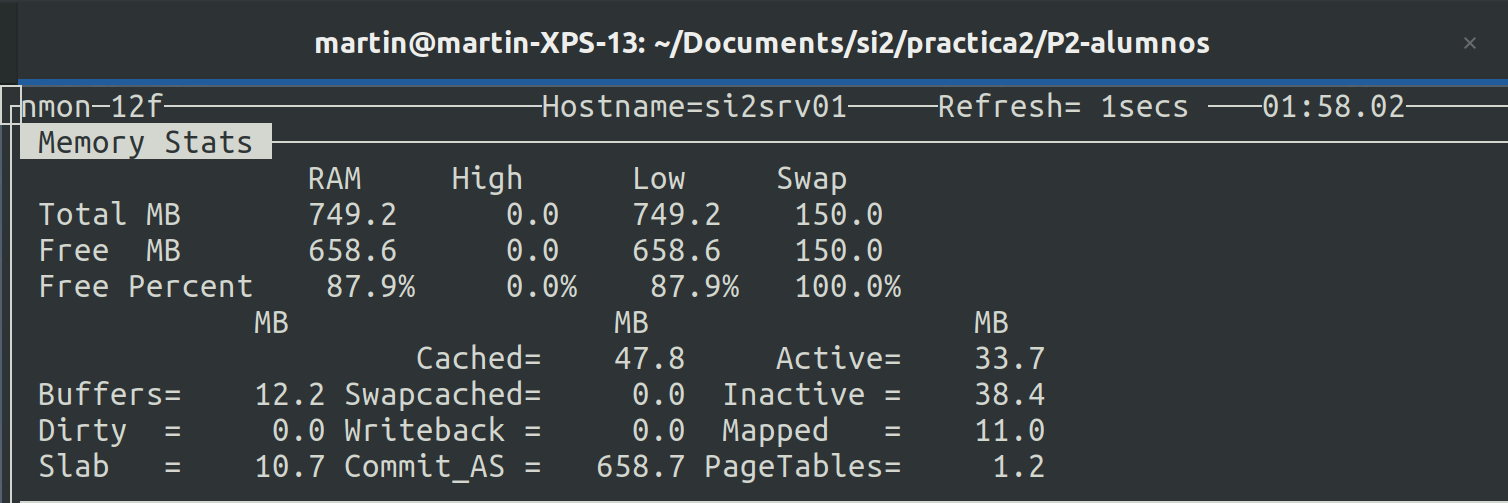
Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

***free (VM1)***



***nmon (VM1)***

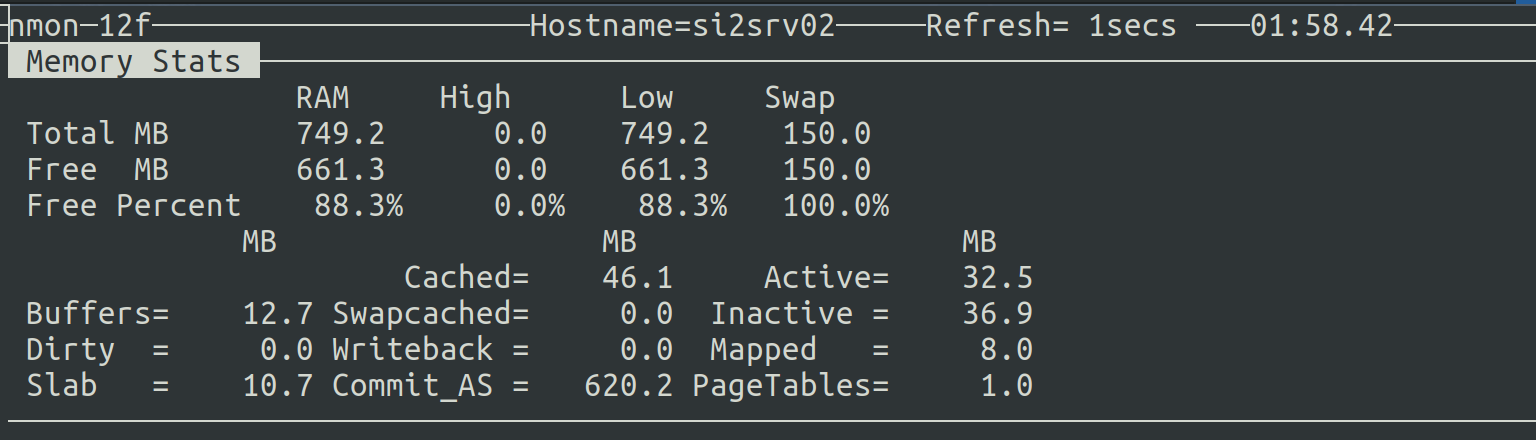


***free (VM2)***

Texto

Descripción generada automáticamente

***nmon (VM2)***



## Ejercicio número 3:

En este ejercicio ejecutaremos el plan de pruebas preparado anteriormente sobre las 3 versiones que hemos preparado de la práctica haciendo uso de JMeter.

Para comprobar que se han realizado correctamente todos los pagos, hemos observado tres elementos distintos:

* ***Result Tree*** (JMeter)

Texto

Descripción generada automáticamente

* ***Base de Datos Visa*** (VM1)

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente con confianza media

* ***Aggregate Report*** (JMeter)

Texto

Descripción generada automáticamente

Se puede observar en la última captura que el Error es del 0% por lo que se han ejecutado correctamente.

A continuación, analizamos los resultados. Para entender el comportamiento tenemos que mirar las siguientes columnas:

* ***Throughpt*** 🡪 Es el Rendimiento. Califica la calidad de la aplicación ejecutada.
* ***Average*** 🡪 Indica el tiempo medio de respuesta en milisegundos.

Si observamos estos parámetros, se observa claramente que la aplicación que mejor rendimiento da es la P1-base (tanto throughput como average son mayores que las otras 2), después observamos que la segunda mejor es P1-ejb y por último P1-ws. (Conclusiones al final del ejercicio junto con ejb local)

Después, se nos pide que realicemos la misma batería de pruebas, pero únicamente con la aplicación P1-ejb-cliente (local). A continuación, se muestran los resultados obtenidos:

* ***Result Tree*** (JMeter)

Texto

Descripción generada automáticamente

* ***Base de Datos Visa*** (VM1)

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

* ***Aggregate Report*** (JMeter)

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Volvemos a observar los parámetros ***throughput*** y ***average*** del *Aggregate Report*.

Se observa que los resultados son muy similares a los de P1-base. De hecho, la media (average) es exactamente la misma y el rendimiento (throughput) es ligeramente inferior.

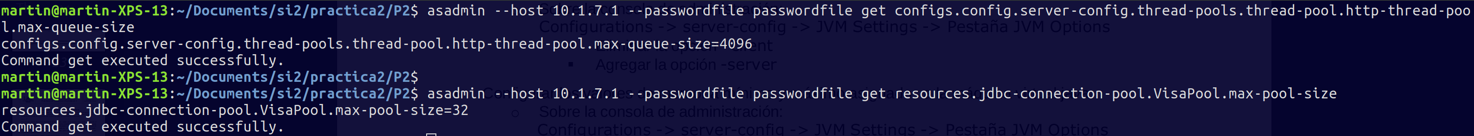
Para terminar, analizando la diferencia de rendimientos nos surge la duda de cuál es razón por la que P1-base P1-ejb (local) tienen un rendimiento tan superior al resto. Esto se puede deber a diferentes factores como pueden ser el ordenador en el que se ejecutan las pruebas o el tipo de arquitectura. Creemos que esta última puede ser uno de los factores principales ya que al ser ambas de tipo “local” no dependen de un intermediario para acceder a las funciones del servidor, sino que lo hacen directamente a través del navegador.

## Ejercicio número 4:

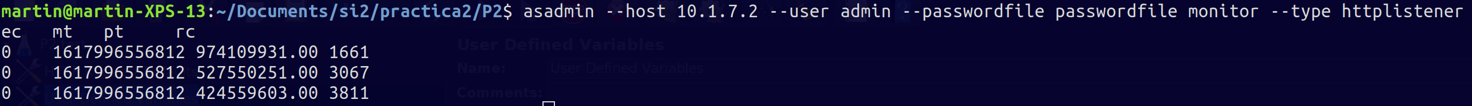
Para este ejercicio, hemos adaptado la configuración del servidor como se indica en el enunciado. Después, hemos obtenido el archivo que contiene esta configuración en la maquina virtual *($opt/glassfish4/glassfish/domains/domain1/config/domain.xml*) y lo hemos copiado en nuestro PC haciendo uso del comando ***scp.***

A continuación, revisamos el fichero ***si2-monitor.sh*** (que se nos provee como material de la práctica), en busca de los mandatos ***asadmin*** que debemos ejecutar en la VM1 para conocer los siguientes parámetros:

1. **Max Queue Size del Servicio HTTP**
2. **Maximum Pool Size del Pool de conexiones a nuestra DB**



Monitorizar el número de errores en las peticiones al servidor:



Donde cada columna representa lo siguiente:

* **ec:** Contador de errores
* **mt:** Tiempo máximo de respuesta
* **pt:** Tiempo acumulado requerido para responder cada solicitud
* **rc:** Número de solicitudes.

## Ejercicio número 5:

Para este ejercicio hemos accedido a la consola de administración de Glassfish y buscamos los parámetros requeridos según se indica en el enunciado. Después introducimos los valores en la tabla indicada de la hoja de cálculo SI2-P2-curvaProductividad.ods.

Tabla

Descripción generada automáticamente

## Ejercicio número 6:

Para realizar este ejercicio, ejecutamos el siguiente comando para activar la monitorización en el servidor:

***$nmon -f -t -s 1 -c 30***

A continuación, ejecutamos de nuevo el plan de pruebas y observamos el fichero .nmon generado haciendo uso de la herramienta NMON Visualizer:

Una captura de pantalla de una computadora

Descripción generada automáticamente

Imagen que contiene Gráfico de embudo

Descripción generada automáticamente

Una captura de pantalla de una computadora

Descripción generada automáticamente

## Gráfico de embudo Descripción generada automáticamente con confianza media

## Imagen que contiene Gráfico de embudo Descripción generada automáticamente

## Ejercicio número 7:

Este ejercicio, como se indica en el enunciado, no requiere que se responda ni que se adjunten capturas.

## Ejercicio número 8:

Este ejercicio, como se indica en el enunciado, no requiere que se responda ni que se adjunten capturas.

## Ejercicio número 9:

Este ejercicio, como se indica en el enunciado, no requiere que se responda ni que se adjunten capturas.