**Virtual Machine Performance Benchmark Tool**

**Manual de Usuario**

**Carlos Eduardo Gómez Montoya**1,2**, MSc, PhD(c).**

**Cristian David Sierra Barrera**1**, Ing.**

**Harold Enrique Castro Barrera**1**, PhD.**

1**Universidad de Los Andes**

2**Universidad del Quindío**

**2017**

**CONTENIDO**

1 INTRODUCCIÓN 1

2 REQUERIMIENTOS 2

3 CONFIGURACIÓN DE LA APLICACIÓN 3

4 PRUEBAS DE DESEMPEÑO 4

4.1 Perfiles de hardware para las máquinas virtuales 4

4.2 Escenarios de ejecución simultánea 5

4.3 Configuración de los escenarios de aplicación 5

4.3.1 Perfiles de hardware para las máquinas virtuales 5

4.3.2 Escenarios de ejecución simultánea 6

4.3.3 Número de pruebas 7

4.3.4 Nombre del archivo de salida 7

4.3.5 Prioridades 8

4.3.6 Pruebas de carga en el cliente 8

4.3.7 Benchmarking 8

4.3.8 Prefijo del nombre de las máquinas virtuales 9

5 EJECUCIÓN DE LA PRUEBA 11

6 RESULTADOS 12

# INTRODUCCIÓN

La computación en la nube ha venido tomando cada vez más fuerza en distintos ambientes, impactando todos los sectores de la economía. La virtualización es una de las tecnologías que han permitido que la computación en la nube pueda ser llevada a la realidad sin importar las dimensiones del usuario que requiere sus servicios.

El análisis de la ejecución simultánea de máquinas virtuales es esencial para la asignación de recursos computacionales en la computación en la nube. La herramienta de software Virtual Machine Performance Benchmark Tool se ha desarrollado con el fin de apoyar el desarrollo de la tesis doctoral “Confiabilidad en sistemas desktop cloud” que actualmente se encuentra en desarrollo por el estudiante doctoral Carlos Eduardo Gómez Montoya en la Universidad de Los Andes. Este programa es una aplicación de software creada para establecer la configuración de máquinas virtuales y definir distintas aplicaciones de carga, de tal forma que se pueda evaluar un método de benchmarking elegido por el usuario, entre los disponibles en el sistema. Dado que realizar estas pruebas generalmente es un trabajo repetitivo y tedioso, la principal funcionalidad consiste en la automatización de escenarios y máquinas virtuales de acuerdo con parámetros que son ajustados por el usuario a través de un archivo de texto en formato JSON con una sintaxis específica.

Esta herramienta de software es de gran utilidad porque no solo se trata de la ejecución automática de despliegues de software sino que además entrega los resultados en forma de valores separados por coma (.csv), los cuales pueden ser analizados en una hoja de Excel o en cualquier programa de análisis estadísticos que reciba datos en este formato.

Se trata de una aplicación cliente – servidor en la cual el servidor corre en el host mientras que el cliente corre en la máquina virtual. El servidor enciende la máquina virtual, mide el tiempo que se tarda la máquina virtual en estar lista para operar y dependiendo de los parámetros de configuración especificados por el usuario, ejecuta y termina las aplicaciones de benchmarking y apaga la máquina virtual. De igual manera, mientras las máquinas virtuales estén creadas, el servidor puede modificar sus especificaciones, otra vez, de acuerdo con los parámetros de entrada especificados.

El presente manual fue diseñado con el propósito de señalar las funcionalidades que tiene el sistema para ofrecer a posibles usuarios. Además, se indica el procedimiento para hacer uso de las herramientas de medición y las configuraciones que deben realizarse para aprovechar la flexibilidad que permite la herramienta.

# REQUERIMIENTOS

Aunque el programa fue desarrollado para su ejecución en ambientes Windows, este programa puede correr de manera correcta en cualquier plataforma que tenga instalado Java Runtime Environment (JRE) versión 7 o superior y Oracle Virtual Box versión 4.3 o superior.

Adicionalmente, la para que se puedan ejecutar las pruebas, es necesario que se disponga de máquinas virtuales ya creadas, debido a que su creación está fuera del alcance del software desarrollado.

# CONFIGURACIÓN DE LA APLICACIÓN

El servidor necesitan unos parámetros generales de configuración los cuales se definen en el archivo *configuration.properties*.

A continuación, y a manera de ejemplo, se encuentran los archivos de propiedades en los cuales se especifican parejas en forma de atributo=valor. Estos atributos se suministran desde los archivos de propiedades para que la configuración sea flexible y no haya que hacer modificaciones al código fuente de la aplicación.

|  |
| --- |
| *Contenido archivo “configuration.properties”* |
| #These are the information about the server configuration  port=20000  virtualMachinesHome=D:\\VMs\\  virtualBoxHome=C:\\Program Files\\Oracle\\VirtualBox\\  # ===================================================================  # Log files configuration  # ===================================================================  #The label of each entry in the log file.  labelLogFile=AdminServer  # The directory path where the log file will be stored.  pathLog=Logs  # The log filename.  logFileName=AdminServer.log |

En el servidor, es necesario que pueda identificar el número de puerto en el que va a escuchar, el tiempo máximo que espera por una conexión especificado en segundos, y el directorio en el cual se almacenan las máquinas virtuales y donde está instalado Oracle Virtual Box.

El cliente debe saber dónde localizar al servidor y el puerto en el que escucha. Así que al ejecutar el archivo *jar*, se debe anexar como parámetro la dirección IP y el puerto separados por espacio. Si no se añaden los argumentos, por defecto esta máquina virtual ejecuta pruebas de carga sin conectarse a un servidor. En este caso, las pruebas de carga se definen en el script *all.sh*, que corre al interior de la máquina virtual (ver documento “Preparación de Máquina Virtual”).

|  |
| --- |
| *Comando de ejecución* |
| Java –jar TCPClient.jar 192.168.0.125 20000 |

# PRUEBAS DE DESEMPEÑO

Las pruebas de desempeño consisten en la ejecución simultánea de máquinas virtuales preparadas cuidadosamente para la ejecución de unas pruebas de desempeño junto con la ejecución de una aplicación cliente-servidor que controla tanto la ejecución de las aplicaciones como la ejecución de las máquinas virtuales, la captura de los tiempos que han tomado las aplicaciones en ejecutarse y el reporte de los resultados obtenidos.

Una prueba de desempeño se compone de escenarios de ejecución simultánea en los cuales se seleccionan la cantidad de máquinas virtuales a ejecutarse al mismo tiempo, junto con su perfil de hardware. Esta información se especifica en archivos de configuración con extensión **pbt** en la carpeta **data** en la raíz del proyecto.

Las pruebas se realizan en las máquinas virtuales mediante la ejecución del programa **stress-ng**. Este programa es un generador de cargas de trabajo configurables sobre sistemas POSIX en términos de CPU, memoria, operaciones de entrada y salida y disco. De manera simultánea, se ejecuta una prueba de benchmark en el host. Por ejemplo, calcular los primeros n números primos, dónde n puede ser especificado en los archivos de extensión **pbt**. Este tipo de pruebas se utiliza para obtener el tiempo que tarda la máquina física en ejecutar la prueba de benchmark especificada sin que se ejecuten máquinas virtuales. Este tiempo se convierte en la línea de base para luego, al ejecutar los escenarios de ejecución simultánea con máquinas virtuales se pueda determinar el impacto que dicha ejecución tiene sobre el escenario base.

## Perfiles de hardware para las máquinas virtuales

Cada máquina virtual que se va a emplear en las pruebas de desempeño debe tener un perfil de hardware que se encuentre especificado mediante el nombre, la cantidad de megabytes de memoria RAM y la cantidad de Cores de procesamiento. Es de anotar que tanto la memoria RAM como los Cores de procesamiento máximos deben ser un subconjunto de los recursos que tenga la máquina física. Por ejemplo, en la Tabla 1, se determinan cuatro perfiles que pueden ser seleccionados para la creación de escenarios de ejecución simultánea que serán descritos en la siguiente subsección.

**Tabla 1. Perfiles de hardware para las máquinas virtuales**

|  |  |  |  |
| --- | --- | --- | --- |
| **Perfil** | **Nombre** | **#Cores** | **RAM** |
| **** | Small | 1 | 1024 |
| **** | Medium | 2 | 2048 |
| **** | Large | 4 | 4096 |
| **** | Extralarge | 4 | 6 |

## Escenarios de ejecución simultánea

Los escenarios de ejecución simultánea consisten en la selección de un conjunto de una o varias máquinas virtuales que serán desplegadas al mismo tiempo para la ejecución de una prueba de desempeño. En la tabla 2 se pueden apreciar tres escenarios: el primero se compone de una máquina virtual de perfil 1, el segundo corresponde a dos máquinas virtuales, ambas de perfil 1, mientras que el tercer escenario incluye tres máquinas virtuales, una de perfil 1 y dos de perfil 2.

**Tabla 2. Escenarios de ejecución simultánea de máquinas virtuales.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Escenario** | **#SVMs** | **VM1** | **VM2** | **VM3** | **Total** | |
| **#Cores** | **RAM** |
| **1** | **1** | **** |  |  | **1** | **1024** |
| **2** | **2** | **** | **** |  | **2** | **2048** |
| **3** | **3** | **** | **** | **** | **5** | **5120** |

**#SVM: Número de VMs que se ejecutan de forma simultánea**

El número de cores de procesamiento y la cantidad de memoria RAM de la máquina física determina la cantidad máxima de máquinas virtuales que se pueden ejecutar de manera simultánea en un host, en caso de que se presente un consumo total de los recursos en las máquinas virtuales.

## Configuración de los escenarios de aplicación

Como se mencionó, los archivos con extensión **pbt** en la carpeta **data** en la raíz del proyecto son los archivos de configuración de las pruebas de desempeño a realizar sobre un host. Estos archivos están escritos en formato JSON, lo que indica que debe cumplir con una serie de elementos sintácticos cuya revisión corre por cuenta y responsabilidad del usuario. Existen ocho parámetros que el usuario puede editar para especificar una prueba y beneficiarse de la ejecución automática: 1) profiles, 2) scenarios, 3) numberOfTests, 4) OutputFilename, 5) priorities, 6) clientTests, 7) benchmarks y 8) machineNamePrefix.

### Perfiles de hardware para las máquinas virtuales

El parámetro *profiles* es un arreglo de objetos que se refiere a los perfiles que pueden tener las máquinas virtuales. Note que las máquinas virtuales tienen varios parámetros por ajustar, entre los más importantes se encuentran la cantidad de memoria RAM y el número de núcleos de procesamiento (cores) a utilizar.

{

"name": "<Nombre del Perfil>",

"RAM": <Cantidad de RAM en GigaBytes>,

"cores": <Número de Cores>

}

Ahora bien, se pueden tener varias configuraciones de perfiles por tal razón, puede crear cuantos perfiles requiera para describir las pruebas que desee realizar. Para anexarlos, tan sólo añada otro esquema de perfil después del último separándolos por una coma. Un ejemplo de la configuración de perfiles que se puede realizar es la siguiente:

"profiles": [

{"name": "#1", "RAM": 1, "cores": 1},

{"name": "#2", "RAM": 2, "cores": 2},

{"name": "#3", "RAM": 4, "cores": 4},

{"name": "#4", "RAM": 8, "cores": 6}

]

### Escenarios de ejecución simultánea

El parámetro *scenarios* es un arreglo de objetos en el que se configuran los tipos de máquinas virtuales a utilizar simultáneamente en la ejecución de una prueba.

El parámetro se compone de una lista de escenarios y cada escenario corresponde a una lista de perfiles de máquinas virtuales. De este modo es posible describir varios escenarios diferentes con variedades de máquinas virtuales.

La forma de realizar la descripción de un escenario es la siguiente:

[

{"nameProfile": "<Nombre del Perfil 1>"},

{"nameProfile": "<Nombre del Perfil 2>"}

]

Esta descripción anterior corresponde a un escenario que tendrá dos máquinas virtuales con los nombres especificados en el lugar indicado. Es de anotar que se debe tener cuidado al adicionar los parámetros puesto que el nombre del perfil a utilizar debe ser igual al nombre de algún perfil definido en el parámetro *profiles*.

Ahora bien, en una prueba, es normal tener varios escenarios, los cuales deben hacer parte de una lista separados por comas.

Un ejemplo de cómo describir escenarios es el siguiente:

"scenarios": [

[

{"nameProfile": "#4"}

],

[

{"nameProfile": "#2"},

{"nameProfile": "#3"}

],

[

{"nameProfile": "#1"},

{"nameProfile": "#1"},

{"nameProfile": "#1"}

]

]

Observe que la lista de escenarios está delimitada por los símbolos "[" y "]" respectivamente y dentro de estos corchetes van los distintos escenarios.

En la descripción anterior, se describen tres escenarios: el primer escenario tiene una máquina virtual con el perfil #4, el segundo escenario tiene dos máquinas virtuales dónde la primera máquina virtual tiene perfil #2 y la segunda tiene perfil #3, finalmente, el último escenario tiene tres máquinas virtuales y todas tienen perfil #1.

### Número de pruebas

El parámetro *numberOfTests* es un número entero que corresponde al número de mediciones a realizar por cada configuración de escenarios.

"numberOfTests": <número de evaluaciones>

Por ejemplo:

"numberOfTests": 5

Este valor es importante porque usualmente los investigadores obtienen los resultados luego de varias ejecuciones para sus análisis estadísticos.

### Nombre del archivo de salida

Como su nombre lo indica, este parámetro corresponde al nombre del archivo de los resultados de las mediciones.

"OutputFilename": "<Nombre del archivo de salida>"

Por ejemplo:

"OutputFilename": "test"

### Prioridades

El sistema permite realizar ejecuciones de las máquinas virtuales en prioridades “Idle” o “Normal” que maneja el sistema operativo Windows. En caso que el host se ejecute en otro sistema operativo, el sistema se ejecuta en prioridad normal.

La forma como se especifican las prioridades es como una lista, con los nombres entre "", separados por comas, tal como se muestra a continuación.

"priorities": ["idle", "normal"]

Al poner una lista de prioridades, se hará una iteración de todas las pruebas para todas las prioridades seleccionadas.

### Pruebas de carga en el cliente

Este parámetro se refiere a la aplicación que será ejecutada en las máquinas virtuales pertenecientes a las pruebas. Para su información, la máquina virtual estándar contiene las siguientes pruebas de carga: prueba de memoria, prueba de CPU, prueba de disco, prueba de operaciones de Entrada/Salida y todas las pruebas anteriores.

"clientTests": ["MEMORY", "CPU", "DISK", "IO", "ALL"]

Se debe elegir al menos uno de estos parámetros. Note que los puede elegir todos. Lo que realizará la aplicación de mediciones es ejecutar todos sus escenarios por cada configuración de cliente elegida.

Ejemplo. Si fue elegido MEMORY y CPU, entonces todos los escenarios serán ejecutados y medidos ejecutando en las máquinas virtuales la prueba de carga de memoria y al finalizar, se vuelve a realizar el proceso ejecutando en las máquinas virtuales de los escenarios una prueba de CPU.

### Benchmarking

Como ya se mencionó, en esta versión del sistema, se tiene disponible una prueba de benchmarking para su ejecución en la máquina física, basada en el cálculo del número de primos con la finalidad de analizar el efecto en tiempo de ejecución del uso simultáneo de las máquinas virtuales de cada prueba. El benchmarking se trata de un algoritmo que busca los n primeros números primos, dónde n corresponde a un número determinado por el usuario, el factor de medición entonces será el tiempo que tarda en alcanzar el n-esimo número primo.

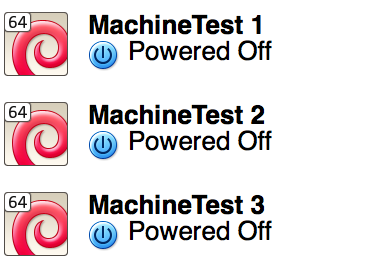
Usted como usuario puede determinar cuál es su límite de análisis en el benchmark especificado, ajustando el parámetro de número de primos.

"benchmark": ["Integers-Primes-FindNPrimes", 1000]

Entonces, se crea un arreglo para especificar el tipo de prueba. El primer parámetro corresponde al tipo de benchmark a ejecutar, los siguientes datos se usa para parametrizar la prueba.

### Prefijo del nombre de las máquinas virtuales

Finalmente, este parámetro corresponde al nombre de las máquinas virtuales que usted debe haber utilizado en la asignación de los nombres de las máquinas virtuales para poder ejecutarlos mediante esta herramienta de software. Se espera que usted haya creado n máquinas, donde n corresponde al número de máquinas virtuales del escenario que requiere la mayor cantidad de máquinas virtuales simultáneas. Por ejemplo, si el prefijo de los nombres de las máquinas virtuales es "MachineTest ", quiere decir que si se va a ejecutar un conjunto de tres máquinas virtuales, sus nombres deben ser "MachineTest 1", " MachineTest 2" y " MachineTest 3", tal como se puede apreciar en la Figura 1. Observe el espacio en blanco al final del prefijo.



“MachineTest “ corresponde al prefijo de las máquinas virtuales, y cada una es marcada con un número empezando desde 1.

" machineNamePrefix": "<Prefijo del nombre de las VMs>"

A continuación se presenta un ejemplo del contenido del archivo de configuración para una prueba de desempeño:

|  |
| --- |
| *Contenido del archivo data/configurationTests.pbt* |
| {  "profiles": [  {"name": "#1", "RAM": 1, "cores": 1},  {"name": "#2", "RAM": 2, "cores": 2},  {"name": "#3", "RAM": 4, "cores": 4}  ],  "scenarios": [  [{"nameProfile": "#2"},{"nameProfile": "#3"}]  ],  "numberOfTests": 2,  "OutputFilename": "testResults1",  "priorities": ["idle"],  "clientTests": ["MEMORY", "CPU"],  "benchmark": ["Integers-Primes-FindNPrimes", 1000],  "machineNamePrefix": "MachineTest"  } |

# EJECUCIÓN DE LA PRUEBA

Habiendo hecho la configuración de las máquinas virtuales y los parámetros en el archivo de configuración de la aplicación, es posible ejecutar la prueba que empezará inmediatamente.

Se ejecuta como un archivo *jar* convencional. Como resultado de la prueba se registran los eventos ocurridos en un archivo log, además de los resultados, los cuales se almacenan en el archivo cuyo nombre se especifica en el parámetro *OutputFilename* del archivo de configuración.

Así que para ejecutar el servidor asegúrese de contar con el siguiente esquema:

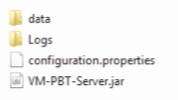


Figura 1 Esquema de ejecución

El archivo *jar* del servidor, acompañado del archivo *configuration.properties*, y la carpeta *data*, donde estarán los archivos tipo *pbt* a ejecutar. La creación de la carpeta Logs es opcional, puesto que el ejecutable la crea si no existe.

# RESULTADOS

Los resultados entregados corresponden al tiempo que tarda la ejecución de la prueba de carga en el host para los distintos escenarios especificados. Como ya se mencionó, el formato es .csv el cual es soportado por distintos programas de análisis estadístico y por Microsoft Excel.

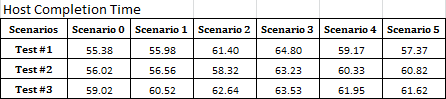
Un ejemplo de la salida de resultados se puede ver a continuación para cinco escenarios de ejecución simultánea de máquinas virtuales, además del escenario 0, donde no se ejecuta ninguna y sirve como línea de base. En este ejemplo, se ejecutaron tres pruebas.

|  |
| --- |
| Scenarios, Scenario 0, Scenario 1, Scenario 2, Scenario 3, Scenario 4, Scenario 5  Test #1, 59.46, 63.94, 64.35, 67.44, 61.17, 68.29  Test #2, 51.89, 52.71, 54.32, 57.26, 57.02, 56.86  Test #3, 59.00, 62.23, 62.29, 67.81, 62.91, 66.17 |

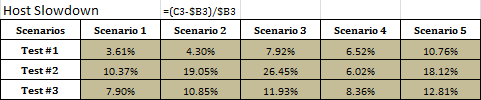
A partir de estos datos, se puede calcular el *slowdown* como el porcentaje de tiempo que afecta cada escenario al escenario 0.

Donde *ti* corresponde a cada uno de los tiempos obtenidos de los escenarios 1 al 5 y *t0* corresponde al tiempo del escenario 0 time. El escenario 0 es el escenario línea de base cuando no se han ejecutado máquinas virtuales.

A continuación se muestra un ejemplo de los cálculos de tiempo de terminación de la prueba de desempeño en el host cuando se ha cargado el archivo de resultados en Microsoft Excel.



Como consecuencia, se calcula el *slowdown* para cada prueba y escenario utilizando la fórmula mencionada. Como se puede apreciar, el escenario 0 no tiene *slowdown* por ser la línea de base, es decir, el punto de referencia.



A partir de estas tablas, es muy fácil construir gráficas estadísticas o realizar cualquier otro análisis estadístico.

Un ejemplo de los archivos de resultados son los siguientes:

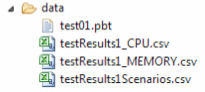


Figura 2 Archivos de Resultados

Se crea un archivo que tiene como formato el nombre asignado en el archivo de propiedades, en la propiedad *OutputFilename*, seguido por la cadena “Scenarios”. Ese archivo contiene los resultados de todas las pruebas organizados por escenarios determinados en los archivos tipo *pbt*.

Además, se crea un archivo por cada tipo de prueba de carga asignado en los archivos tipo *pbt*. Y cada formato csv, contiene los tiempos de ejecución.