|  |  |  |
| --- | --- | --- |
| Material de apoyo para página web y redes - IPN | Práctica  Apache HTTP server benchmarking tool | Historia e Identidad Gráfica - ESCOM |

Elaborada por: M. en C. Ukranio Coronilla

En esta clase nos interesa aprender a realizar un banco de pruebas para los

servidores HTTP de manera que nos permita evaluar el rendimiento de un servidor HTTP, simulando múltiples usuarios que realizan solicitudes concurrentes.

Para obtener el desempeño del servidor utilizaremos la herramienta **ab** (Apache HTTP server benchmarking tool) cuya página se muestra en:

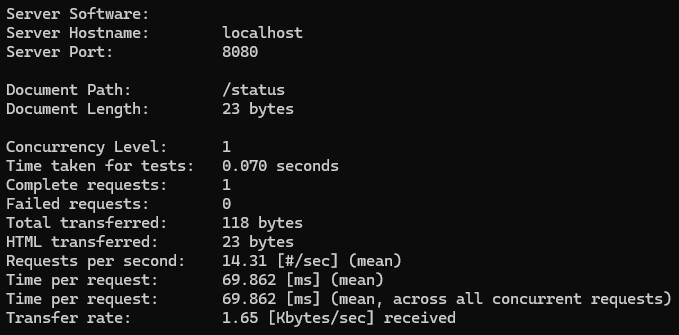
<https://httpd.apache.org/docs/trunk/programs/ab.html> Proceda a instalarla en su sistema LINUX con el comando:

sudo apt install apache2-utils

Y pruébela ejecutando en una terminal el servidor en el puerto 8080 y en otra terminal la herramienta ab con:

ab <http://localhost:8080/status>

Obtendrá una salida del programa como la siguiente:



# Ejercicio 1

Revise con ayuda del manual de la aplicación (véase man), el significado de cada uno de los términos compilados en la siguiente tabla y descríbalos con sus propias palabras:

|  |  |
| --- | --- |
| **Item** | **Descripción** |
| Complete requests | Este ítem indica el numero de solicitudes que fueron completadas de manera exitosa. |
| Failed requests | Este ítem indica el numero de solicitudes que tuvieron algun fallo durante su proceso, e indica que tipo de fallo tuvieron en su ejecuión(lectura, contexión, contenido, etc). |
| Total transferred | Indica el numero de bytes totales que se recibieron desde el servidor, |
| HTML transferred | Esto indica el número total de bytes de documentos recibidos desde el servidor (es decir, el contenido), excluyendo los bytes de los headers. |
| Requests per second | Es el promedio de solicitudes que se completan en el servidor por segundo, tomando en cuenta el numero de solicitudes totales y el tiempo que total que se tomó en completarlos. |
| Time per request | Similar al anterior, en este se muestra el promedio de tiempo que le toma al servidor completar una solicitud. Primer valor calculado de la siguiente manera: concurrency \* timetaken \* 1000 / done. Segundo valor calculado imetaken \* 1000 / done. |
| Transfer rate | Mide la velocidad promedio en la que se tranfieren los bits de las solicitudes. Este valor se mide con el balor de bits por segundo (bps). Se calcula siguiendo la siguiente formula: totalread / 1024 / timetaken |

El programa ab puede recibir diversas opciones. Con ayuda del manual de la aplicación consense con su equipo para llenar la siguiente tabla:

|  |  |
| --- | --- |
| **Opción** | **Descripción** |
| -n | Numero de solicitudes que se le van a realizar al servidor |
| -c | Numero de concurrencias con el que se le van a arealizar las solicitudes al servidor |
| -k | Permite realizar múltiples solicitudes dentro de una misma sesión HTTP. |
| -v | 4 y superiores muestran información sobre los encabezados, 3 y superiores muestran los códigos de respuesta (404, 200, etc.) |
| -p | El archivo contiene datos que se enviaran mediante POST. |
| -T | Es el encabezado a utilizar para las solicitudes POST/PU |

# Ejercicio 2

Usando las opciones adecuadas, envíe al endpoint /status del servidor mil

solicitudes no concurrentes, donde cada solicitud va en una sesión independiente de HTTP. Anote el tiempo obtenido de promediar cinco ejecuciones para llenar la tabla.

Ahora varie la cantidad de conexiones concurrentes desde 2 hasta 12 para encontrar con que cantidad de conexiones concurrentes el servidor tiene un mejor desempeño, es decir que atiende más solicitudes por segundo. Anote el tiempo obtenido de promediar cinco ejecuciones en cada caso para llenar la tabla.

Repita el ejercicio usando dos computadoras conectadas en una LAN por medio de un celular utilizado como punto de acceso (esto para que no afecte a la medición el tráfico en la red LAN de la escuela) y llene la siguiente tabla:

|  |  |  |
| --- | --- | --- |
| Caso | Local | LAN |
| Tiempo usado para 1000 solicitudes en sesiones independientes y sin  concurrencia | Promedio:0.4228 segundos | Promedio: 23.8372 segundos. |
| Cantidad de conexiones concurrentes optimas y número máximo de  solicitudes por segundo  conseguidos | El número de conexiones concurrentes optimas es: 10  Número máximo de solicitudes por segundo conseguidos: 11093.98 | El número de conexiones concurrentes optimas es: 6  Número máximo de solicitudes por segundo conseguidos: 807.39 |

**Importante**: En el caso que no se pueda realizar la conexión entre dos computadoras si se utiliza WSL aun deshabilitando los firewalls, se puede utilizar la terminal de Windows MS-DOS para correr el

servidor.

# Ejercicio 3

Para el caso de dos computadoras conectadas en una LAN. permita que se realicen múltiples solicitudes dentro de una sesión HTTP (opción -k) y vea si aumenta la cantidad de solicitudes por segundo probando con valores entre 1 y 12 conexiones concurrentes. Escriba el número máximo de solicitudes por segundo conseguidos junto con el número de conexiones concurrentes y explique brevemente en que consiste la característica Keep-Alive del protocolo HTTP.

2922.42 Solicitudes por segundo con 11 conexiones concurrentes

Lo que realiza Keep-Alive del protocolo HTTP dentro de estas pruebas de rendimiento es que mantiene abierta la comunicación con el servidor, a diferencia de no mantener activar Keep-Alive. Esto mejora bastante el numero de solicitudes en el servidor, pues al no cerrar y volver a abrir la comunicación para cada solicitud, el tiempo de respuesta se vuleve notoriamente mas corto.

# Ejercicio 4

Levante el servidor al que se le agregó el endpoint /searchtoken y utilice las opciones adecuadas para que la aplicación ab pueda enviar al servidor una solicitud exitosa al endpoint **searchtoken** verificando con el parámetro **-v** que la solicitud se contesta de manera correcta por el servidor. Para enviar los datos desde el cliente tendrá que usar el parámetro -p y agregar un archivo con los datos para que se

considere una solicitud POST. Adjuntar captura de pantalla probando que se resuelve correctamente la solicitud.

***Importante***: Es posible que la solicitud no envíe correctamente los datos y parezca que no hay ocurrencias en la búsqueda de la cadena. Esto se puede deber a que su editor está insertando un carácter salto de línea en el archivo con los datos. En estos casos se recomienda usar otro editor y verificar que el tamaño del archivo en bytes corresponde exactamente con el número de caracteres insertados.

# Ejercicio 5

Suponiendo que todos los clientes acceden al endpoint **searchtoken** con búsquedas en una cadena con 17576 tokens, ¿cuál sería la máxima cantidad de solicitudes por segundo que puede resolver su servidor en una LAN (no usar la opción -k pues se supone que son distintos usuarios en internet)?

**Sugerencia**: Comience con un numero bajo de solicitudes totales y usuarios

concurrentes y vaya incrementándolos para observar cómo se comporta el servidor bajo diferentes niveles de carga.

Adjunte la captura de pantalla donde se observen las opciones introducidas al comando ab y la salida del programa con su valor óptimo para este ejercicio.