Mondragon Unibertsitatea

Goi Eskola Politeknikoa

Faculty of Engineering

ANÁLISIS DE LA CONCURRENCIA

MUDLEY

Daniel Luengo, Egoitz San Martin, Unai Aguinaco, Unai Mendieta, Unai Orive

Contenido

Calculos previos

Los servicios de Mudley están construidos sobre APACHE. Sin una configuración previa, APACHE tiene una capacidad máxima para 150 conexiones simultáneas. Para poder cambiar este número se requiere cambiar la configuración, aunque también es necesario un sistema capaz de soportarlo.

Uno de los parámetros más importantes para la mejora de concurrencia es la RAM, cuanto mayor sea, mayores capacidades de conexiones simultáneas podrá soportar el servidor.

Utilizando el siguiente comando podremos saber una media aproximada de la cantidad de RAM que consume cada conexión de APACHE. El resultado puede variar dependiendo de los servicios activos en el momento.

ps -ylC apache2 --sort:rss | awk '{SUM += $8; I += 1} END {print SUM/I/1024}'

Lógicamente un servidor no ejecuta únicamente los procesos que el usuario ha ejecutado o programado, hay varios procesos/servicios que se ejecutan de fondo, otros que son indispensables para el funcionamiento correcto del servidor… Para poder calcular la cantidad de RAM que consumen se puede utilizar el comando de abajo:

ps -N -ylC apache2 --sort:rss | awk '{SUM += $8} END {print SUM/1024}'

Con estos datos se podría calcular el número de hilos que el sistema podría soportar, la fórmula que tendríamos que aplicar es la siguiente:

Nº de hilos= (RAM total - RAM de otros procesos)/RAM media por conexión

Aplicar la cifra conseguida podría no ser lo más óptimo, dado que utilizar todos los hilos sería llevar la RAM hasta el límite. Es recomendable reducir el número de hilos, por ejemplo si hay 230 bajarlo hasta 200, de esta manera se le da al servidor un margen para poder trabajar de manera más cómoda.

Otro dato que hay que tener en cuenta es el número de núcleos o Cores que el sistema tiene, pues estos son los que permitirán el trabajo de forma paralela.

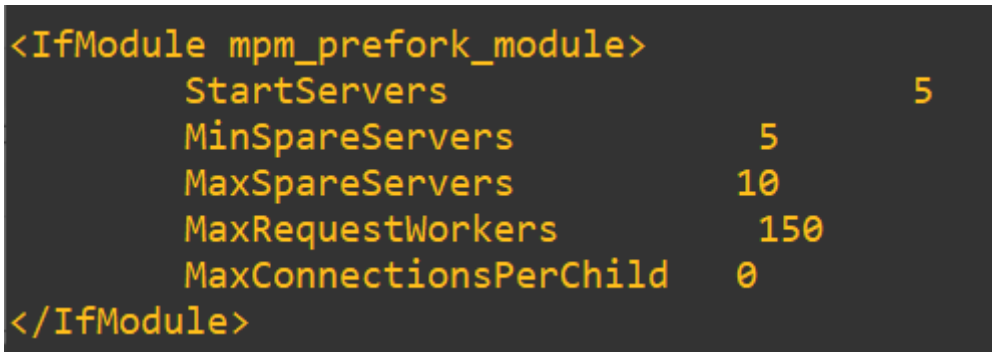
Por último, existe otro cálculo que podría ayudar. Este cálculo indica cuál sería el número de usuarios simultáneos recomendable para el servidor.

(nº de cores / media en segundos de tiempo de respuesta de la web) * 60 * Frecuencia de Clicks en segundos = nº maximo de usuarios simultaneos

Con este parámetro se podría controlar mejor la cantidad de usuarios simultáneos que el servicio puede aguantar; por lo que como se explicará más adelante, se podrá adaptar mejor a las necesidades.

Configuración del servidor

Una vez con los cálculos hechos, hay que establecer el nuevo número de hilos en el servidor. En el caso de este proyecto, se utiliza un sistema que trabaja con el OS de ubuntu. En la carpeta de */etc/apache2/mods-available/* se encuentra el fichero *mpm_prefork.conf* donde se tendrá que configurar todo lo relacionado con la concurrencia. Para ello se utilizará PHP normal. La información relativa al PHP se podrá ver en el siguiente apartado.



Como se ve en la imagen en este fichero hay varios parámetros, pero el más importante para los diferentes temas de concurrencia es *MaxRequestWorkers (en versiones de apache anteriores se denominaba como maxClients)*, como se ha especificado al principio, de serie viene en 150 peticiones simultáneas como máximo.

Como se está cambiando un parámetro predeterminado, hay que añadir encima del *MaxRequestWorkers* el parámetro *ServerLimit*, el cual especifica el número de peticiones extra que podría haber en el servidor una vez pasado el límite del *MaxRequestWorkers*. Es decir el valor del *ServerLimit* siempre debería ser ligeramente superior al *MaxRequestWorkers*, por ejemplo, establecerlo en 300 si el otro parámetro llega hasta 280. El máximo que puede tener ServerLimit es de 2000. Para aumentarlo aún más allá de este límite, se deberá modificar el valor de *MAX_SERVER_LIMIT* en el archivo fuente mpm y reconstruir el servidor.

Para que funcione correctamente, se utiliza el módulo mpm_prefork, suele estar inicializado; pero, en caso de que no esté funcional, se puede utilizar el siguiente comando para inicializarlo.

```
sudo a2enmod mpm_prefork
```

La última comprobación que sería recomendable llevar a cabo es el reinicio del servicio apache para que todos los cambios se apliquen de manera correcta.

```
sudo /etc/init.d/apache2 restart
```

PHP y PHP-FPM

A la hora de configurar el servidor hay que tener en cuenta también la cantidad de tráfico que el servidor va a tener que soportar. Cuando la carga de tráfico no es grande, utilizar PHP podría servir (PHP es un lenguaje de programación de código abierto), pero, si el servidor va a tener que soportar un tráfico alto, utilizar PHP-FPM sería lo mejor. PHP-FPM es la implementación alternativa más popular de PHP FastCGI, que cuenta con características adicionales realmente útiles para sitios web de alto tráfico.

En lo que respecta a la concurrencia, previamente se ha explicado cómo configurar el servidor con PHP, pero se podría considerar utilizar PHP-FPM; estos son los pasos que habría que seguir para ello:

El fichero a configurar es el siguiente:

sudo nano /etc/apache2/mods-available/mpm_event.conf

Una vez dentro del fichero, se pueden ver los siguientes parámetros:

```
<IfModule mpm_event_module>
        StartServers                     2
        MinSpareThreads          25
        MaxSpareThreads          75
        ThreadLimit                      64
        ThreadsPerChild          25
        MaxRequestWorkers         150
        MaxConnectionsPerChild   0
</IfModule>
```

La mayor diferencia entre PHP (mpm_prefork) y PHP-FPM (mpm_event), en este caso, es que el segundo ofrece más parámetros, lo que otorga un mayor control sobre los hilos del servidor.

Concurrencia en Mudley

Mudley es una aplicación web que en una primera instancia, va a estar dirigida al País Vasco. Teniendo en cuenta el alcance de la web y el nicho de mercado al que se dirige, la aplicación tendrá un máximo de 200 usuarios concurrentes.

No solo hay que tener en cuenta el número de usuarios que van a utilizar el servidor de manera concurrente, el tipo de servidor también influye. Mudley trabaja con los servidores que ofrece AWS (Amazon Web Service). Dependiendo del tipo que se escoja, la RAM puede variar, por ejemplo, t2.nano solo tiene 0.5GB, y por el otro lado, r2.2xlarge tiene 32GB, por lo que, como se puede ver, hay un rango bastante amplio. En estos servidores también varía el número de cores.

Mudley funciona de manera distribuida, los servicios están divididos en diferentes servidores; uno para la IA (intelingencia artifical), otro para la web y un último para la base datos.

Servidor Web

El servidor web de mudley es el servicio principal, este se conectará al resto de servidores. Utilizando la fórmula del nº máximo de usuarios simultáneos, se han conseguido los siguientes cálculos.

(1/1.8)*60*2=66

Para poder calcular la media en segundos de tiempo de respuesta de la web se ha utilizado un servicio web externo, *browserstack speedlab*. La frecuencia de clicks es una aproximación que se ha obtenido tras probar la web con varios usuarios diferentes.

En este caso se ha determinado que el servidor podría aguantar de manera correcta un máximo de 48 usuarios concurrentes, teniendo en cuenta que se utiliza un servidor tipo t2.small, que solo tiene un core. En este caso, como se han estimado un máximo de 200 usuarios concurrentes, se necesitarían unos 4 núcleos (el número de cores siempre es múltiplo de 2), siendo los servidores t2.large los primeros que ofrecen este número.

(**4/1.8)*60*2=266**

Si se considera que de normal el 75% del tiempo, en cuanto al funcionamiento de una web, los hilos se lo pasan esperando, podríamos decir que el número de hilos necesarios sería 4 veces el número de cores. En este caso, se utilizan 4 núcleos, el número de hilos sería de 16.

Si en algún momento el servicio evolucionara y se empezase a utilizar a mayor escala (nivel nacional o incluso internacional), se tendrían que hacer ciertos cambios en la configuración. En el caso de utilizar el servicio a nivel nacional, se estiman unos 1000 usuarios concurrentes, y en caso de tener alcance internacional, se apostaria por 4000 usuarios concurrentes. Lógicamente estos datos son orientativos y se tendrían que hacer varias pruebas para optimizar el servicio lo máximo posible.

A continuación se muestra una tabla con los datos que se han establecido para los diferentes alcances. Esta tabla contiene los datos relacionados con el País Vasco, España y el nivel internacional.

Escala	**CORES**	**Usuarios**		**MaxRequest Workers**	**ServerLimit**
		máximo	**Concurrentes**		
País vasco	4	10000	200	16	20
Nacional	16	80000	1000	64	80
Internacional	64	400000	4000	256	300

En el caso del nivel internacional (también podría aplicarse, aunque en menor medida al nivel nacional), sería recomendable dividir los servidores en más servidores, en vez de utilizar uno centralizado. De esta manera, se podría mejorar la latencia y el tiempo de respuesta del servidor.

Utilizando el metodo de dividir el servidor en servidores de menor tamaño, se podría hacer frente de una manera diferente a los problemas de concurrencia.

Escala	**CORES**	**Usuarios**		**MaxRequest Workers**	**Server limit**	**Nº servidor**
		máximo	**Concurrentes**			
País vasco	2	5000	100	8	10	2
Nacional	4	20000	250	16	20	4
Internacional	8	50000	500	32	40	8

Dividiendo de esta manera los servidores se podría garantizar una menor carga de trabajo, optimizando en algunos casos los servicios. Por ejemplo, cuando trabajamos a una escala

pequeña, podría ser un cambio poco significante; pero cuenado se trabaja a niveles más altos, como el internacional, podría se un cambio necesario.

Conclusión

La escala del servidor puede variar bastante las decisiones que se van a tomar respecto a la concurrencia. Lógicamente, si el servicio se dirige a una escala pequeña, como puede ser el País Vasco, utilizar un único servidor sería suficiente; pero cuando la escala es mayor, como a nivel internacional, lo lógico sería utilizar varios servidores, no solo para aligerar la carga de trabajo, también para mejorar el tiempo de respuesta y la latencia.

Servidor IA

La inteligencia artificial de Mudley está desplegada en un servidor que funciona independientemente de la web. Este servicio atenderá las diferentes peticiones que llegan, todas provenientes del servidor web (utilizando Node-Red), por lo que no tendrá que esperar a nadie, una vez llegue la petición, la IA empezará a trabajar.

Siguiendo los mismos procedimientos que se han aplicado antes para el servidor web, se ha calculado el número de usuarios que podría soportar el servidor.

(1/2)*60*1=30

Se ha establecido que la frecuencia de clicks es de 1, puesto que no se puede acceder directamente al servicio IA. El tiempo de respuesta se ha determinado que es de 2 segundos tras las diferentes pruebas que se han hecho con el servidor. En el caso de utilizar un único nucleo, el servidor solo podría soportar un máximo de 30 usuarios simultaneos.

Como los procesos de la IA no esperan a nadie, los threads no esperán. Teniendo esto en cuenta, sabiendo que la carga de trabajo es bastante elevada se ha decidido que cada núcleo estará dedicado exclusivamente a un hilo. Si Tenemos en cuenta que el número de usuarios concurrentes máximos es de 200, lo más eficaz sería utilizar un servidor de 8 cores.

(8/2)*60*1=240

En la siguiente tabla se muestra como se tendría que construir el servidor en caso de usar un único núcleo.

Escala	**CORES**	**Usuarios**		**MaxRequest Workers**	**ServerLimit**
		máximo	**concurrentes**		

País vasco	8	10000	200	8	10
Nacional	32	80000	1000	32	40
Internacional	128	400000	4000	128	140

Como se puede ver, tanto la escala nacional como la internacional trabajan con un número de cores muy grande, por lo que lo más recomendable sería dividirlo en servidores más pequeños.

Escala	**CORES**	**Usuarios**		**MaxRequest Workers**	**Server limit**	**Nº servidor**
		máximo	**Concurrentes**			
País vasco	4	5000	100	4	5	2
Nacional	8	20000	250	8	10	4
Internacional	16	50000	500	16	20	8

Conclusión

A diferencia del servidor web, el servidor de IA puede recibir cargas de trabajo mayores, por lo que sería recomendable (más recomendable cuanto mayor sea el alcance) dividir el servidor en servidores de menor tamaño.

Servidor base de datos

Al igual que el servidor de IA, la base de datos, se encuentra en un servidor independiente al que solo accede el servidor web; pero el funcionamiento interno es diferente. Los hilos del servidor no se detienen en ningún momento a esperar nada que venga de fuera del servidor, pero si que se tiene que esperar la respuesta de mysql (está en el mismo servidor).

Teniendo en cuenta que los hilos se pueden detener durante un tiempo mientras se efectua la consulta, se ha decidido que se va a implementar un nucleo por cada dos threads; puesto que, a diferencia del servidor web, la consulta que se hace es interna.

(2/0,5)*60*1=240

Teniendo en cuenta la velocidad de la base de datos para responder (tan solo medio segundo) utilizando únicamente dos nucleos se podría conseguir el número de 200 usuarios simultaneos.

Escala	CORES	Usuarios		MaxRequest Workers	ServerLimit
		máximo	Concurrentes		
País vasco	2	10000	200	4	5
Nacional	8	80000	1000	16	20
Internacional	32	400000	4000	64	80

La base de datos, tanto a nivel nacional como en el País Vasco, no tiene sentido dividirla en varios servidores, porque es un servicio centralizado. En el caso del nivel internacional, habría que tener en cuenta las políticas de la empresa, se pretende trabajar con los mercados nacionales de cada país, o con un único mercado internacional. En caso de querer trabajar con mercados nacionales, esta podría ser la solución:

Escala	CORES	Usuarios		MaxRequest Workers	Server limit	Nº servidor
		máximo	Concurrentes			
Internacional	8	80000	1000	16	20	x

Se ha establecio el Nº de servidores como X porque dependería del número de paises donde se utiliza el servicio; por ejemplo, si la aplicación trabajase en 20 paises, el número x sería 20, es decir, un servidor por país.

Conclusión

El servidor de base de datos solo utilizará un servidor en el caso nacional y en el país vasco; pero, dependiendo las políticas de la empresa, podría utilizarse un único servidor o varios servidores (uno por país) a nivel internacional.