ESCALADO

El escalado es necesario siempre que queramos que nuestras aplicaciones sean capaces de manejar un mayor número de usuarios al mismo tiempo, despachar un mayor número de solicitudes en paralelo, etc.

En definitiva, con el escalado lo que se busca es aumentar la capacidad y la disponibilidad de nuestra plataforma, así como reducir las posibilidades de tiempo de inactividad.

En Cloud Foundry PaaS disponemos de 2 tipos de escalado:

1. ESCALADO HORIZONTAL

El escalado horizontal de una aplicación consiste en aumentar el número de instancias en ejecución para una determinada aplicación desplegada en la nube Cloud Foundry.

El escalado horizontal puede ser positivo o negativo, esto es crear nuevas instancias de la aplicación o destruir instancias existentes de la misma.

Para escalar horizontalmente, se utiliza el siguiente comando:

- **$ cf scale APP -i INSTANCES**
- **$ cf scale myApp -i 5**

Donde “INSTANCES” es el número de instancias y “APP” es el nombre de la aplicación la cual queremos escalar.

En el ejemplo podemos apreciar cómo se le asignan 5 instancias a la aplicación “myApp”.

2. ESCALADO VERTICAL

El escalado vertical de una aplicación en Cloud Foundry consiste en variar/modificar las prestaciones que las instancias de la aplicación tiene atribuidas.

Esto nos permitirá limitar el espacio en disco que Cloud Foundry atribuye a las instancias de una determinada aplicación, el límite de memoria que se le aplica a las instancias, etc.

Para escalar verticalmente se utiliza una variación del comando de escalado horizontal. Veámoslo con unos ejemplos:

- **$ cf scale APP -k DISK**
- **$ cf scale myApp -k 512M**

Aquí podemos ver como limitamos la cantidad de disco que Cloud Foundry le asigna a las instancias de la aplicación “myApp”, limitándola a una cantidad igual a 512 Megabytes de disco usado para cada instancia de dicha aplicación

- **$ cf scale APP -m MEMORY**
- **$ cf scale myApp -m 1G**

Aquí podemos ver como limitamos la cantidad de memoria que Cloud Foundry le asigna a las instancias de la aplicación “myApp”, limitándola a una cantidad igual a 1 Gigabyte de memoria usado para cada instancia de dicha aplicación.

ACERCA DEL COMANDO “SCALE”

Nos permite ver o cambiar el recuento de instancias, el límite de espacio en disco y el límite de memoria para una aplicación.

$ cf scale APP_NAME [-i INSTANCES] [-k DISK] [-m MEMORY] [-f]

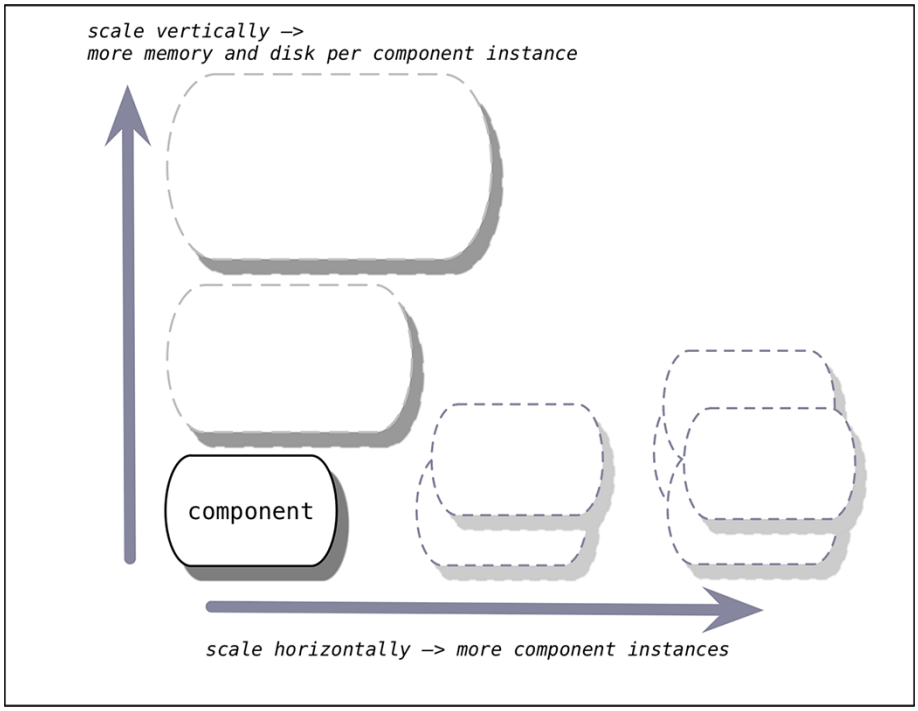
- **f:** Nos permite forzar el reinicio de la aplicación sin aviso.
- **i:** Número de instancias
- **k:** límite de disco que se usa para cada instancia de la aplicación (por ejemplo, 256M, 1024M, 1G).
- **m:** límite de memoria que se usa para cada instancia de la aplicación (por ejemplo, 256M, 1024M, 1G).

Hay que tener en cuenta que el hecho de escalar una aplicación a 5 instancias y 32GB de memoria, implica que cada una de las 5 instancias tendrá como límite 32GB de memoria, y no que cada una de las 5 instancias tenga 32/5(GB) como límite de memoria.

3. ESCALADAO DE LOS COMPONENTES DE UNA IMPLEMENTACIÓN DE CLOUD FOUNDRY Y CONCEPTO DE “AZ” (ZONAS DE DISPONIBILIDAD)

Lo cierto es que las aplicaciones que desplegamos en nuestra nube Cloud Foundry no es lo único que puede ser escalable.

Otro aspecto que puede escalarse son los propios componentes de la implementación de una nube Cloud Foundry.



Es decir, hablamos de escalar la plataforma en sí misma, escalar la implementación de Cloud Foundry a través del escalado de sus componentes, donde el escalado horizontal implicará la creación de un mayor número de máquinas virtuales (VM's) que ejecuten nuevas instancias de los componentes de la plataforma.

Es por ello que una ampliación horizontal para ciertos componentes tiene como implicación el aumentar la capacidad de alojar aplicaciones.

El vertical agregará memoria y disco a cada instancia del componente, tal y como lo hace el vertical sobre las aplicaciones.

A la hora de hacer un escalado vertical, se ha de considerar:

- Tener suficiente espacio libre existente en las celdas del subsistema Diego de las máquinas virtuales anfitrionas para que las aplicaciones que se espera que se desplieguen se puedan organizar y ejecutar con éxito (Son en las celdas de Diego donde se aceptan y ejecutan las tareas e instancias de "Long Running Process" [LRP] de las diferentes aplicaciones que se ejecutan en la nube).

- Una asignación suficiente de espacio en disco y memoria en su implementación de tal manera que, si una máquina virtual anfitriona está inactiva, todas las instancias de aplicaciones se pueden colocar en las máquinas virtuales host restantes.

- El espacio libre para manejar una AZ (zona de disponibilidad) cae/baja si se hace un despliegue/implementación en múltiples AZ.

El escalado de componentes de una implementación de Cloud Foundry surge a partir de los riesgos que tiene la venida abajo de una de las AZ (zonas de disponibilidad) que componen la implementación de la nube de Cloud Foundry.

Lo cierto es que cuando se producen updates de productos o upgrades en la plataforma, es frecuente que alguna de las máquinas virtuales VM's que ejecutan instancias, se venga abajo haciéndola temporalmente inasequible.

Es por ello que una correcta distribución de los componentes de la implementación de la nube Cloud Foundry a través de las distintas Zonas de Disponibilidad que la compongan y su correspondiente escalado a un nivel suficiente de redundancia garantizará una alta disponibilidad tanto durante las actualizaciones como durante las interrupciones y con ello se puede garantizar un tiempo de inactividad con tendencia al 0.

La implementación de Cloud Foundry en tres o más AZ y la asignación de múltiples instancias de componentes a diferentes ubicaciones de AZ permiten que una implementación funcione sin interrupciones cuando alguna de las AZ queda inoperable.

Cloud Foundry mantendrá su disponibilidad mientras la mayoría de las AZ sigan siendo accesibles. Por ejemplo, una implementación de tres AZ se mantiene en alto cuando una AZ completa se cae, y una implementación de cinco AZ puede soportar una interrupción de hasta dos AZ sin afectar en el tiempo de actividad.

4. AUTOESCALADOR

En caso de que usemos el proveedor Pivotal Software para Cloud Foundry, dispondremos de una aplicación para auto escalar aplicaciones integrada en la interfaz de usuario Apps Manager.

Esto ayudará a controlar el coste de ejecutar aplicaciones mientras se mantiene el rendimiento de la aplicación. Para equilibrar el rendimiento y el coste de la aplicación, se puede utilizar esta aplicación para lo hacer lo siguiente:

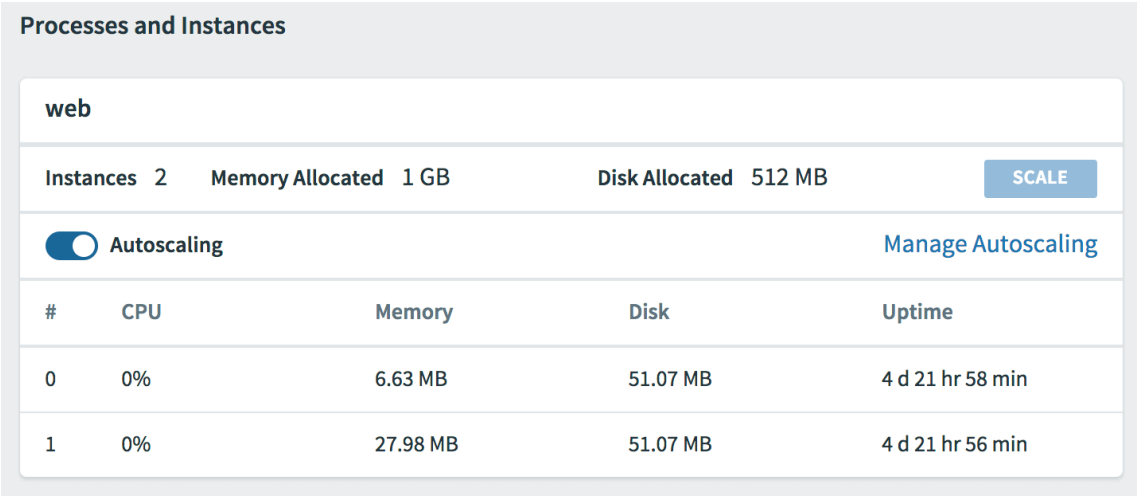
- Configurar reglas que ajusten los recuentos de instancias según los umbrales de las métricas, como el uso de la CPU.

- Modificar el número máximo y mínimo de instancias para una aplicación, ya sea manualmente o siguiendo un programa.

Para usar la aplicación de auto escalar, se debe crear una instancia del servicio "App Autoscaler" y vincularla a cualquier aplicación que queramos escalar automáticamente.

Una vez hecho esto, nos vamos al administrador de aplicaciones (App Manager) y desde ahí seleccionamos una de las aplicaciones del espacio en el que se creó el servicio autoscaler de la aplicación (la instancia anteriormente mencionada)

Como paso siguiente, nos vamos a la pestaña "Procesos e instancias" y activamos el botón "Autoscaling".



Desde la pestaña de administración de auto escalado, “Manage Autoscaling”, podemos limitar el número mínimo y máximo de instancias de la aplicación.

Si en algún momento decidimos apostar otra vez por escalar manualmente una aplicación la cual anteriormente hayamos vinculado a una instancia del servicio de auto escalado, se desactivará el auto escalado que hayamos configurado. Cuidado con esto.

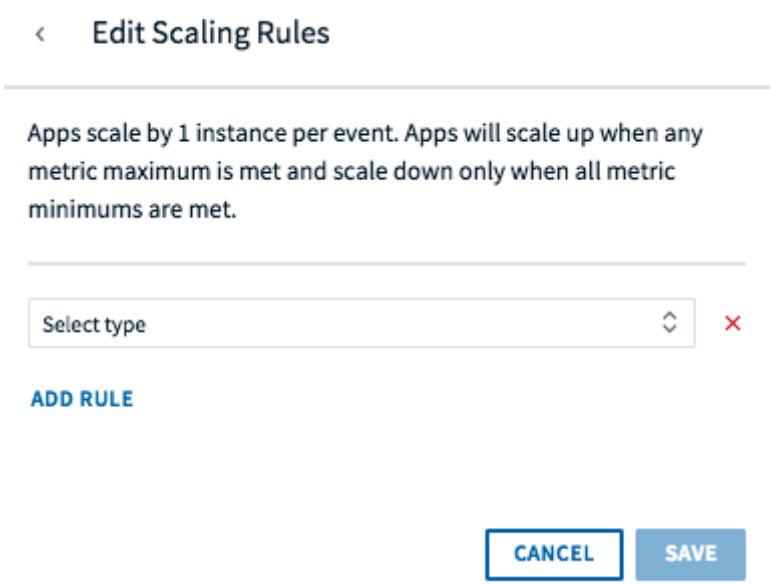
Para mantener las aplicaciones disponibles sin desperdiciar recursos, la aplicación Autoscaler incrementa o disminuye el número de instancias en base a una comparación que lleva a cabo entre la métrica (CPU, RAM, disco, etc.) y el umbral mínimo/máximo establecido para dicha métrica

La aplicación Autoscaler escala las aplicaciones de la siguiente manera:

- Incremento en una instancia cuando cualquier métrica excede el umbral máximo especificado.
- Disminuye en una instancia solo cuando todas las métricas caen por debajo del umbral mínimo especificado.

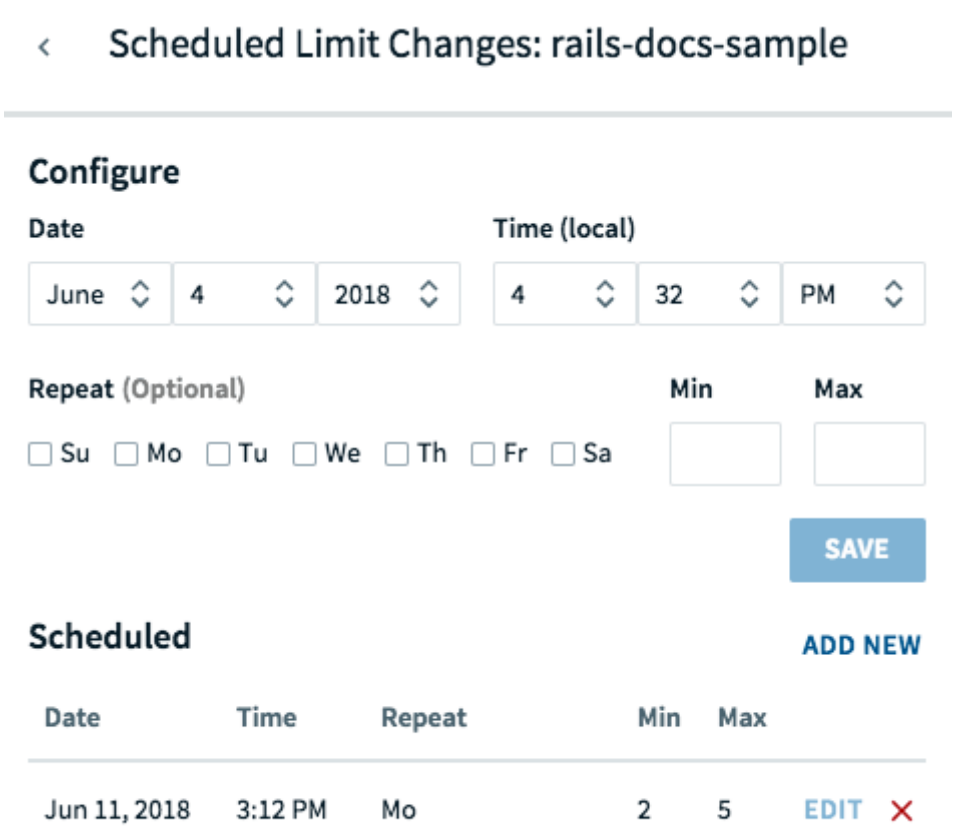
Algunas métricas en las que podemos basarnos para crear las reglas del auto escalador de aplicaciones:

- **Uso de la CPU:** Porcentaje de CPU promedio para todas las instancias de la aplicación.
- **Utilización de memoria de un contenedor:** Porcentaje de memoria promedio para todas las instancias de la aplicación.
- **Rendimiento en peticiones HTTP:** Total de solicitudes HTTP por segundo (dividido por el número total de instancias de la aplicación).



Debido a que la demanda de aplicaciones a menudo sigue un programa semanal, diario o por horas, se puede programar la aplicación Autoscaler para que cambie el rango de instancias permitido para seguir los aumentos esperados o los períodos de inactividad que se presuponen, ya que sabemos que la demanda de la aplicación difiere según el momento.

- **Fecha y hora:** establezca la fecha y la hora del cambio de número de instancias.
- **Repetir:** establezca el día de la semana en el que desea repetir el cambio.
- **Mínimo Y Máximo:** Establezca el rango permitido dentro del cual App Autoscaler puede cambiar el número de instancias para una aplicación.



También existen otras opciones de código abierto como **app-autoscaler** y **cf-autoscaler**.

REFERENCIAS

https://docs.cloudfoundry.org/concepts/high-availability.html

https://docs.cloudfoundry.org/devguide/deploy-apps/cf-scale.html

http://cli.cloudfoundry.org/en-US/cf/scale.html

https://stackoverflow.com/questions/45735971/when-scaling-a-cloud-foundry-app-is-the-memory-split-among-the-instances

https://docs.pivotal.io/pivotalcf/2-4/appsman-services/autoscaler/using-autoscaler.html

https://github.com/cloudfoundry-incubator/app-autoscaler

https://github.com/cloudfoundry-samples/cf-autoscaler