**OPENSHIFT WEEK: *“SCALING & AUTOSCALING”.***

******

# El siguiente documento detallará el paso a paso, el manejo del: SCALING & AUTOSCALING de los *CONTENEDORES* en OPENSHIFT.

|  |  |
| --- | --- |
| **DESCRIPCIÓN** | **DETALLE** |
| 1. **REQUERIMIENTOS:** | | |
| El único requerimiento es tener ***INSTALADA*** la herramienta: **SIEGE** nivel de: **VARIABLES DE ENTORNO**.  Esta herramienta permitirá ***PROBAR*** más adelante  la aplicación del ***RECURSO***: **AUTOSCALING**.   * **WINDOWS:**     *$ siege –version*   * **LINUX:**   *$ sudo apt update -y*  *$ sudo apt install siege -y*  *$ siege –version*   * **MAC:**   *$ brew install siege*  *$ siege --version* |  | |
| 1. **ESCALAMIENTO DE *‘CONTENEDORES’*:** | | |
| Horizontal pod autoscaling - Mastering Kubernetes [Book]  El objetivo aquí es el poder ***CRECER*** a nivel de ***INSTANCIAS*** ***(PODs),*** para aplicarlos en los escenarios requeridos. Estos pueden ser: ***MANUÁLMENTE*** o ***AUTOMÁTICAMENTE***. | | |
| 1. **SCALING *INSTANCES*:** | | |
| Este procedimiento es el **MANUAL** & para ello se debe  crear un ***ESCENARIO*** de prueba, corriendo el **Script** ***YAMEL***:    *$ cat >* ***Script\_Microservicio.yaml***  *...*  *…..*  *…….*  *$ oc create -f Script\_Microservicio.yaml*  *$ oc get all -n dummy-csm-crga* |  |
|  |
| Luego, se ingresa el ***COMANDO*** para el **AUMENTO *MANUAL*** hacia 5 ***INSTANCIAS***:  *$ oc scale deployment dummy-micro-deploy --replicas=5 -n dummy-csm-crga*  *$ oc get pods -n dummy-csm-crga* |  |
| Luego, se realiza una prueba del ***MICROSERVICIO*** desplegado. Los ***REQUEST*** enviados serán distribuidos  a nivel de ***TODAS*** las ***INSTANCIAS*** desplegadas:  *$ curl http://dummy-micro-route-dummy-csm-crga.cluster-cla-crga-ccc03eca20d26e6ac64511f874a64b9b-0000.br-sao.containers.appdomain.cloud/dummy-micro-01/get/personas* |  |
| Finalmente, para el **DECREMENTO** ***MANUAL*** de ***INSTANCIAS*** hacia: 1 *(dejarlo como estaba inicialmente),* se ingresa:  *$ oc scale deployment dummy-micro-deploy --replicas=1 -n dummy-csm-crga*  *$ oc get pods -n dummy-csm-crga* |  |
| 1. **HORIZONTAL AUTOSCALING *INSTANCES*:** | | |
| El objetivo aquí es el poder ***CRECER*** a nivel de ***INSTANCIAS*** ***(PODs),*** esto aplicado en los escenarios de ***TRÁFICO CAMBIANTE,*** sobre todo, pero de manera: ***AUTOMÁTICA***.  **IMPORTANTE:** *“Antes que nada, se debe conocer que el* ***HORIZONTAL AUTOSCALING****, requiere que esté* ***ACTIVO*** *algo que se llama:* ***METRIC-SERVER****, esto sirve  para poder desde* ***OPENSHIFT*** *medir el control* ***CPU*** *&* ***MEMORIA*** *tanto a nivel de:* ***NODOS*** *&* ***PODs*** *respectivamente.*  *En* ***KUBERNETES*** *no está instalado, pero si en* ***OPENSHIFT******(NO se requiere)****”:*  *$ git clone* [*https://github.com/kodekloudhub/kubernetes-metrics-server.git*](https://github.com/kodekloudhub/kubernetes-metrics-server.git)  *$ cd kubernetes-metrics-server*  *$ oc create -f ./* | |
| Luego, para ***PROBAR*** este escenario se requiere ejecutar  los ***COMANDOS IMPERATIVOS*** siguientes:    *$ oc autoscale deployment dummy-micro-deploy --min=1 --max=4 --cpu-percent=40 -n dummy-csm-crga*  *$ oc get pods,hpa -o wide -n dummy-csm-crga*    **IMPORTANTE:** *“Esta modalidad a nivel de* ***COMANDOS****, aplica una* ***REGLA*** *que funciona solo a nivel de* **CPU***”.* |  |
| Luego, es necesario obtener el **ROUTE** creado del ***MICROSERVICIO***, ejecutando:  *$ oc get routes -n dummy-csm-crga* |  |
| Luego, se requiere probar ***MASIVAMENTE*** contra  el ***MICROSERVICIO*** desplegado. Así mismo, los ***REQUEST*** serán distribuidos a nivel de ***TODAS*** las ***INSTANCIAS*** desplegadas y/o se vayan **ACTIVANDO**:    **SIEGE** permitirá enviar ***REQUEST*** de manera ***PARALELA***, esto significa que en la demostración se enviarán:  ***4 PETICIONES***, cada una con ***20 CONCURRENCIAS*** *(en* ***TOTAL*** *se deberán ejecutar:* ***80******Mensajes****)*.  *$ siege --concurrent=20 --reps=4 -v* [*http://* *dummy-micro-route-dummy-csm-crga.cluster-cla-crga-ccc03eca20d26e6ac64511f874a64b9b-0000.br-sao.containers.appdomain.cloud/dummy-micro-01/get/personas*](http://my-route-csm.apps-crc.testing/dummy-micro-01/get/personas)  **IMPORTANTE:** ***“****La ejecución del* ***TRÁFICO MASIVO*** *hará que el* ***CPU****, como se muestra la* ***IMAGEN****, se incremente  & genere la* **ELASTICIDAD** *a nivel de* ***INSTANCIAS*** *de* ***PODs”****.*  *$ oc get pods,hpa -n dummy-csm-crga* |  |
| **(VERSIÓN LINUX)**  **(VERSIÓN MAC)** |
| Luego, se debe verificar que después de un ***TIEMPO***  de realizada la ***PRUEBA*** de: **REQUEST MASIVOS**,  la plataforma se debe haber estabilizado a nivel de ***CPU***.  *(La* ***ELIMINACIÓN*** *de los* ***PODs*** *adicionales es* ***AUTOMÁTICA****).*  *$**oc get pods,hpa -n dummy-csm-crga* |  |
| Finalmente, solo para **COMPLEMENTAR** que la modalidad  en ***SCRIPT*** del: **HORIZONTAL AUTOSCALING**, **NO** solamente funciona a nivel de ***CPU***, sino que también  funciona ***SIMULTÁNEAMENTE*** a nivel de ***MEMORIA***.  *“Esto vuelve más* ***SENSIBLE*** *la activación del:* ***HORIZONTAL AUTOSCALING****”.*  *apiVersion: autoscaling/v2beta2*  *kind: HorizontalPodAutoscaler*  *metadata:*  *name: dummy-micro-deploy*  *namespace:* ***dummy-csm-crga***  *spec:*  *scaleTargetRef:*  *apiVersion: apps/v1*  *kind: Deployment*  *name: dummy-micro-deploy*  *minReplicas: 1*  *maxReplicas: 4*  *metrics:*  *- type: Resource*  *resource:*  *name: cpu*  *target:*  *type: Utilization*  *averageUtilization:* ***40***  *- type: Resource*  *resource:*  *name: memory*  *target:*  *type: Utilization*  *averageUtilization:* ***60***  *$**oc get pods,hpa -n dummy-csm-crga*  **IMPORTANTE:** *“Se aprecia como la* ***REGLA*** *del* **HORIZONTAL AUTOSCALING***, ahora está aplicara a nivel de:* ***CPU*** *&* **MEMORIA***”.* |  |