



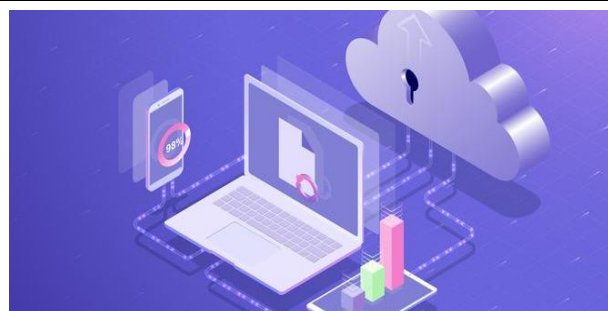
Universidad de Guadalajara

CUCEI – Centro Universitario de Ciencias Exactas e Ingenierías

Mtro. Michel Emanuel López Franco

Computación tolerante a fallas

Reporte 17



Velasco Hernandez, Victor Manuel

Murillo Cortes, Jeanette

| D06 | 2022A |

| 16/Mayo/2022 |

Failure injection





Universidad de Guadalajara
CUCEI
Computación tolerante a fallas

Jeanette, Murillo Cortes
|Código: 216455164 | | 2022A |
|16/ Mayo /2022 | |D06|



Reporte 17-Failure injection.

Objetivo:

Generar ejemplo.

Introducción:

El uso de Chaos Engineering para realizar pruebas de funcionamiento de los proyectos desplegados sobre tecnología de contenedores brinda la posibilidad de predecir problemas dentro de los sistemas de computo o clúster, los cuales nos hacen tener mejores maneras de manejar eventualidades, lo cuál hace que sea posible la realización de procesos cada vez más complejos con mayor seguridad, lo cuál resulta ser una excelente herramienta para lograr una mejora en el manejo de operaciones realizadas dentro de un clúster.

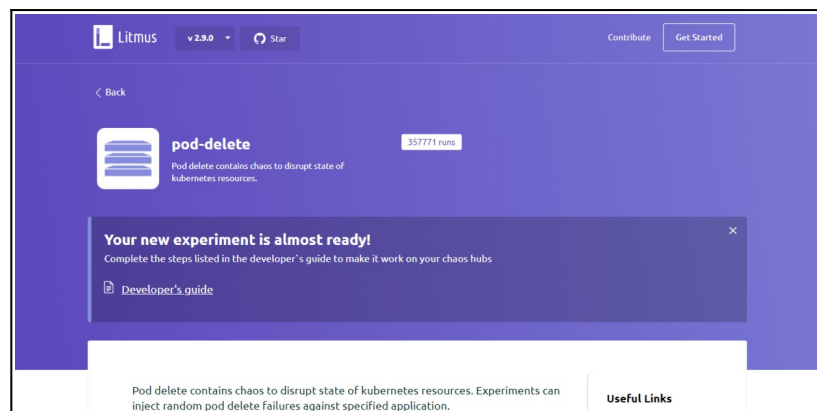
Desarrollo:

Para esta actividad, se utilizara el proyecto IoT de la materia para hacer pruebas con una herramienta de Chaos Engineering llamada Litmus, es open-source, proyecto encubado de la CNCF (Cloud Native Cloud Foundation), con este haremos pruebas de "pod-delete" .

Lo primero que se tiene que hacer, es instalar Litmus en el clúster, esto será posible con el comando:

```
kubectl apply -f https://litmuschaos.github.io/litmus/litmus-operator-v2.2.0.yaml
```

Lo siguiente, consiste en buscar dentro del ChaosHub de Litmus, el experimento "pod-delete".





Universidad de Guadalajara
CUCEI
Computación tolerante a fallas

Jeanette, Murillo Cortes
|Código: 216455164 | | 2022A |
|16/ Mayo /2022 | |D06|

Ya estando en la pagina correspondiente, buscamos el comando de intalación del chaos experiment "pod-delete", con el comando:

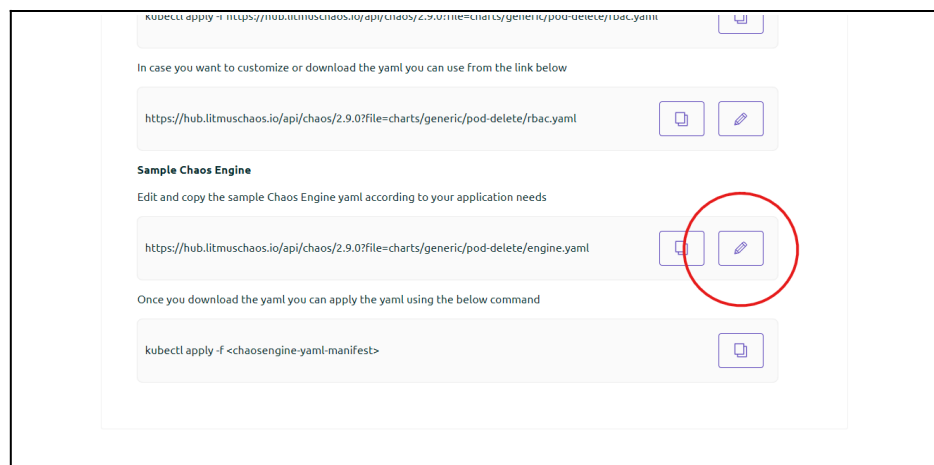
```
kubectl apply -f https://hub.litmuschaos.io/api/chaos/2.9.0?file=charts/generic/pod-delete/experiment.yaml
```

Después, al experimento se le conceden permisos(RBAC) para que el experimento trabaje, con el comando:

```
kubectl apply -f https://hub.litmuschaos.io/api/chaos/2.9.0?file=charts/generic/pod-delete/rbac.yaml
```

```
Administrador: Windows PowerShell
PS C:\WINDOWS\system32> kubectl apply -f https://litmuschaos.github.io/litmus/litmus-operator-v2.2.0.yaml
namespace/litmus unchanged
serviceaccount/litmus created
clusterrole.rbac.authorization.k8s.io/litmus created
clusterrolebinding.rbac.authorization.k8s.io/litmus created
deployment.apps/chaos-operator-ce created
customresourcedefinition.apiextensions.k8s.io/chaosengines.litmuschaos.io created
customresourcedefinition.apiextensions.k8s.io/chaosexperiments.litmuschaos.io created
customresourcedefinition.apiextensions.k8s.io/chaosresults.litmuschaos.io created
PS C:\WINDOWS\system32> kubectl apply -f https://hub.litmuschaos.io/api/chaos/2.2.0?file=charts/generic/pod-network-latency/experiment.yaml
chaosexperiment.litmuschaos.io/pod-network-latency created
PS C:\WINDOWS\system32> kubectl apply -f https://hub.litmuschaos.io/api/chaos/2.9.0?file=charts/generic/container-kill/experiment.yaml
chaosexperiment.litmuschaos.io/container-kill created
PS C:\WINDOWS\system32> kubectl apply -f https://hub.litmuschaos.io/api/chaos/2.9.0?file=charts/generic/container-kill/experiment.yaml
```

Al final de esa pagina, hay un apartado llamado “Sample Chaos Engine”, en la cuál, nos da la opción para modificar un archivo YAML, para adaptar el experimento a nuestro proyecto, seleccionamos editar:

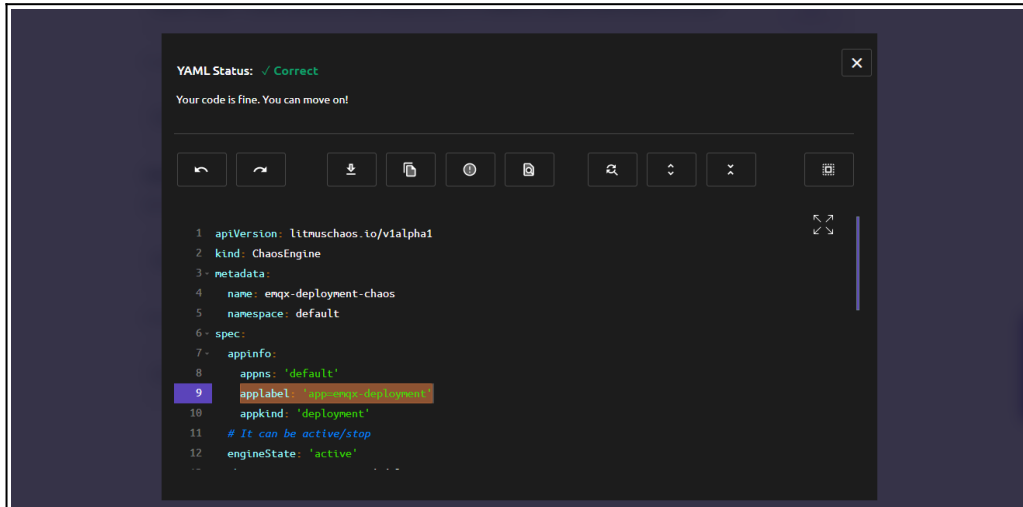




Universidad de Guadalajara
CUCEI
Computación tolerante a fallas

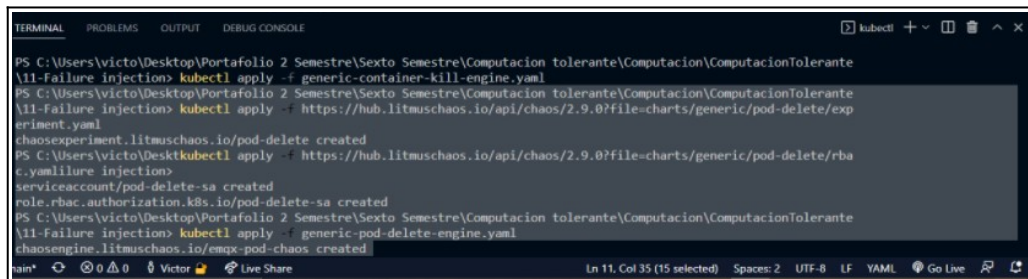
Jeanette, Murillo Cortes
|Código: 216455164 | | 2022A |
|16/ Mayo /2022 | |D06|

Nos aparecerá la siguiente apartado, los campos del archivo mostrado, los modificamos para que encaje con el proyecto a utilizar, y descargamos el archivo.



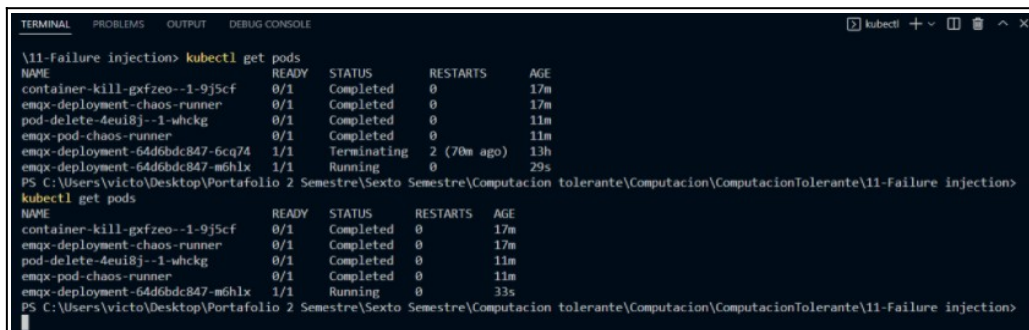
```
1 apiVersion: litmuschaos.io/v1alpha1
2 kind: ChaosEngine
3 metadata:
4   name: emqx-deployment-chaos
5   namespace: default
6 spec:
7   appinfo:
8     appns: 'default'
9     applabel: 'emqx-deployment'
10    appkind: 'deployment'
11    # It can be active/stop
12    engineState: 'active'
```

Por ultimo, con el archivo YAML, lo aplicamos en nuestro cluster con el comando “`kubectl apply -f <nombre del archivo>`”.



```
PS C:\Users\victo\Desktop\Portafolio 2 Semestre\Sexto Semestre\Computacion tolerante\Computacion\ComputacionTolerante> kubectl apply -f generic-container-kill-engine.yaml
PS C:\Users\victo\Desktop\Portafolio 2 Semestre\Sexto Semestre\Computacion tolerante\Computacion\ComputacionTolerante> kubectl apply -f https://hub.litmuschaos.io/api/chaos/2.9.0?file=charts/generic/pod-delete/experiment.yaml
chaosexperiment.litmuschaos.io/pod-delete created
PS C:\Users\victo\Desktop\Portafolio 2 Semestre\Sexto Semestre\Computacion tolerante\Computacion\ComputacionTolerante> kubectl apply -f https://hub.litmuschaos.io/api/chaos/2.9.0?file=charts/generic/pod-delete/rbac.yaml
serviceaccount/pod-delete-sa created
role.rbac.authorization.k8s.io/pod-delete-sa created
PS C:\Users\victo\Desktop\Portafolio 2 Semestre\Sexto Semestre\Computacion tolerante\Computacion\ComputacionTolerante> kubectl apply -f generic-pod-delete-engine.yaml
chaosengine.litmuschaos.io/emqx-pod-chaos created
```

Ya se encontrará funcionamiento el experimento, a continuación la evidencia:



```
\11-Failure injection> kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
container-kill-gxfzoe-1-9j5cf        0/1     Completed 0           17m
emqx-deployment-chaos-runner         0/1     Completed 0           17m
pod-delete-4eui8j-1-whckg            0/1     Completed 0           11m
emqx-pod-chaos-runner               0/1     Completed 0           11m
emqx-deployment-64d6bdc847-6cq74     1/1     Terminating 2 (70m ago) 13h
emqx-deployment-64d6bdc847-m6hlx     1/1     Running    0           29s
PS C:\Users\victo\Desktop\Portafolio 2 Semestre\Sexto Semestre\Computacion tolerante\Computacion\ComputacionTolerante> kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
container-kill-gxfzoe-1-9j5cf        0/1     Completed 0           17m
emqx-deployment-chaos-runner         0/1     Completed 0           17m
pod-delete-4eui8j-1-whckg            0/1     Completed 0           11m
emqx-pod-chaos-runner               0/1     Completed 0           11m
emqx-deployment-64d6bdc847-m6hlx     1/1     Running    0           33s
PS C:\Users\victo\Desktop\Portafolio 2 Semestre\Sexto Semestre\Computacion tolerante\Computacion\ComputacionTolerante>
```

[1]-[4]



Universidad de Guadalajara
CUCEI
Computación tolerante a fallas

Jeanette, Murillo Cortes
|Código: 216455164 | | 2022A |
|16/ Mayo /2022 | |D06|

Conclusiones:

El uso de herramientas de Chaos Engineering son fundamentales para conocer los aspectos vulnerables de la gestión del clúster, resulta ser muy útil para analizar los fallos y corregirlos, además de que es esto nos ayudara a evitarnos muchos problemas, garantizando la preservación del código y la integridad de la información del clúster, convirtiéndolo en sistema tolerante a fallos, por lo que resulta ser fundamental conocer la naturaleza de este servicios para desarrollar cada vez, mejores funciones que garantizan el mejor acceso a la información de los programas, y la presencia de ejecución sin tenerlo en el equipo, lo cuál ayuda a tener una mejor comprensión de la gestión de los clúster, esto hace que podamos evitar posibles problemas de computo distribuido en el futuro.

Link de repositorio:

<https://github.com/Victor012396/ComputacionTolerante.git>

Bibliografía:

- [1] S. Parekh, "Step by step guide to chaos testing using Litmus Chaos toolkit", Medium, el 4 de noviembre de 2021. <https://medium.com/@sunitparekh/step-by-step-guide-to-chaos-testing-using-litmus-chaos-toolkit-c5480f0f6ad0> (consultado el 20 de mayo de 2022).
- [2] "ChaosCenter Cluster Scope Installation | Litmus Docs". <https://docs.litmuschaos.io/docs/getting-started/installation/> (consultado el 21 de mayo de 2022).
- [3] "Why should you choose Litmus? | Litmus Docs". <https://docs.litmuschaos.io/docs/introduction/features/> (consultado el 21 de mayo de 2022).
- [4] "ChaosHub | Cloud-Native Chaos Experiments". <https://hub.litmuschaos.io> (consultado el 21 de mayo de 2022).