Если вы видите что-то необычное, просто сообщите мне.

Настройка безотказного K8s

Независимо от того, пользуетесь ли вы k8s недолго, или вы все еще проверяете его, это говорит, что вы уже имели дело с ним ранее. Но что же такое настройка безотказного k8s кластера?

Что такое события k8s?

Приходилось ли решать проблемы k8s при его использовании? Это может быть довольно сложно, но понимание событий и состояний может сильно помочь. K8s события представляют из себя то, что случается внутри кластера. Событие это тип ресурса создаваемый автоматически, когда происходит изменения состояния кластера. Как вы можете увидеть, событие очень важный ресуср при решении проблем. Прочитайте по поводу state/event управления и таймеров по подробнее, это вам поможет в работе.

Поток управления состоянием

Если вы понимаете что такое поток управления состоянием, легко понять почему некоторые состояния падают, и как можно это предотвратить, давайте капнем глубже:

Kubelet в каждой ноде кластера обновляет API сервевр основываясь на частоте укаазнной в `node-status-update-frequence` параметре. Значение по умолчанию 10 секунд. Затем, переодически, controller-manager проверяет состояние ноды через API сервер. Частота

настроенна в `node-monitor-peroid` параметре и по умолчанию составляет 5 секунд. Если controller-manager видит, что нода не здорова в течении `node-monitor-grace-period`(по-умолчанию 40 секунд), то он помечает её как `unhealthy` через controller-manager. Затем controller-manager ожидаает `pod-eviction-timeout`(по-умолчанию 5 минут) и говорит API серверу убрать поды установив для них состояние `terminate`. Kube proxy получает уведомление о удалении ноды от API сервера. Kube proxy удаляет недоступный под.

Что случается с кластером, когда нода не может этого сделать, основываясь на временных ограничениях. В примере выше, это займент 5 минут и 40 секунд(`node-monitor-grace-period` + `pod-eviction-timeout`) для удаления недоступного пода и возвращения в режим готовности. Это не проблема если `deployment` имеет несколько подов(значение `replica` больше чем 1) и поды на здоровой ноде могут обрабатывать все запросы без проблем. Если `deployment` имеет один под или здоровый под не может обрабатывать запросы, тогда 5 минут и 40 секунд это не приемлемое время недоступности сервиса, поэтому лучшее решение настроить переменные в кластере для ускорения реакции на проблемы. Как это сделать, спросите вы? Давайте пройдемся вместе:

Изменения конфигурации для улучшения безотказности кластера.

Решение точно работает для Kubernetes v1.18.3

Сокращаем node-status-update-frequency

`node-status-update-frequency` - параметр kubelet, он имеет значение по-умолчанию 10 секунд.

Шаги для того, чтотбы заменить значение по-умолчанию

1. Изменяем параметр kublet во всех нодах(master и workers) через файл /var/lib/kubelet/kubeadm-flags.env

```
vi /var/lib/kubelet/kubeadm-flags.env
```

2. Добавляем “--node-status-update-frequency=5s” параметр в конец следующей линии

```
KUBELET_KUBEADM_ARGS="--cgroup-driver=systemd --network-plugin=cni --pod-infra-container-image=k8s.gcr.io/pause:3.2 --node-status-update-frequency=5s"
```

3. Сохранаяем файл.
4. Рестартим kubelete.

```
systemctl restart kubelet
```

5. Повторяем шаги 1-4 на всех нодах.

Сокращаем node-monitor-period и node-monitor-grace-period

`node-monitor-period` и `node-monitor-grace-period` настройки controleler-manager b и их значения по-умолчанию 5 секунд и 40 секунд соотвественно.

Шаги для того чтобы их изменить

1. Настроим kube-controller-manager в мастер нодах.

```
vi /etc/kubernetes/manifests/kube-controller-manager.yaml
```

2. Добавим следующие два параметра в kube-controller-manager.yaml файл

```
- --node-monitor-period=3s
- --node-monitor-grace-period=20s
```

После добавления двух параметров, конфигурация должна выглядеть примерно так:

```
spec:
  containers:
  - command:
  - kube-controller-manager
  . . . [There are more parameters here]
  - --use-service-account-credentials=true
  - --node-monitor-period=3s
  - --node-monitor-grace-period=20s
  image: k8s.gcr.io/kube-controller-manager:v1.18.4
  imagePullPolicy: IfNotPresent
  ...
```

3. Перезапускаем докер

```
systemctl restart docker
```

4. Повторяем шаги 1-3 на всех мастер нонах

Сокращаем pod-eviction-timeout

`pod-eviction-timeout` можно сократить установив дополнительный флаг для API сервера.

Шаги для изменения параметра

1. Создаем новый файлkubeadm-apiserver-update.yaml в /etc/kubernetes/manifests папки мастер ноды

```
cd /etc/kubernetes/manifests/
vi kubeadm-apiserver-update.yaml
```

2. Добавляем следующее содержание в kubeadm-apiserver-update.yaml

```
apiVersion: kubeadm.k8s.io/v1beta2
kind: ClusterConfiguration
  kubernetesVersion: v1.18.3
  apiServer:
```

```
extraArgs:
  enable-admission-plugins: DefaultTolerationSeconds
  default-not-ready-toleration-seconds: "20"
  default-unreachable-toleration-seconds: "20"
```

> “ Убеждаемся, что kubernetesVersion совпадает с вашей версией Kubernetes

3. Сохраняем
4. Выполняем следующую команду для применения настроек

```
kubeadm init phase control-plane apiserver --config=kubeadm-apiserver-update.yaml
```

5. Проверяем, что изменения которые были в kube-apiserver.yaml примеенены для `default-not-ready-toleration-seconds` и `default-unreachable-toleration-seconds`

```
cat /etc/kubernetes/manifests/kube-apiserver.yaml
```

6. Повторяем шаги 1-5 для всех мастер нод.

Шаги выше меняеют `pod-eviction-timeout` для всего кластера, но есть еще один способ изменить `pod-eviction-timeout`. Это можно сделать добавив `tolerations` во все `deployment`, что позволит применить конфиг только на определенный`deployment`. Для такой настройки `pod-eviction-timeout`, добавьте следующие строки в описание `deployment`:

```
tolerations:
- key: "node.kubernetes.io/unreachable"
  operator: "Exists"
   effect: "NoExecute"
  tolerationSeconds: 20
- key: "node.kubernetes.io/not-ready"
  operator: "Exists"
  effect: "NoExecute"
  tolerationSeconds: 20
```

> Если вы работаете с управляемым сервисом Kubernetes, таким как Amazon EKS или AKS, то у вас не будет возможности обновить `pod-eviction-timeout` в кластере. Необходимо использовать `tolerations` для `deployment`.

Вот и всё, вы успешно обработали события K8s.

Revision #6
Created 2022-11-08 14:21:19 UTC by gasick
Updated 2023-04-16 19:36:18 UTC by gasick