K8S 18: Kubernetes. Manifests. Resources

Описание:

Важным аспектом работы планировщика в Kubernetes являются ресурсы. Для более полной утилизации хостов планировщик должен знать объем доступных ресурсов и запросы на эти ресурсы. Все это прописывается в описании контейнера.

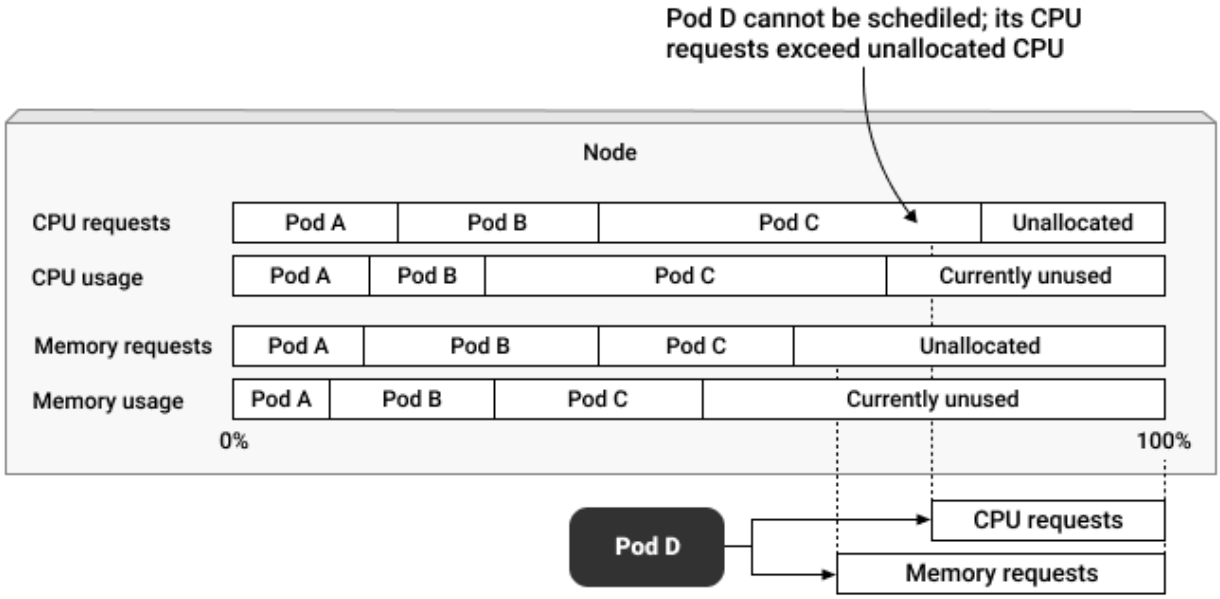
```
apiVersion: v1
kind: Pod
metadata:
  name: requests-pod
spec:
  containers:
  - image: busybox
    command: ["dd", "if=/dev/zero", "of=/dev/null"]
    name: main
    resources:
      requests:
        cpu: "200m"
        memory: "10Mi"
```

В примере выше мы запрашиваем 10 мегабайт памяти и 200 миллиядер процессора (1/5 ядра). Если применить такую конфигурацию и запустить после этого в контейнере утилиту top, то мы увидим, что использовано более 50% cpu, а это никак не соответствует 1/5.

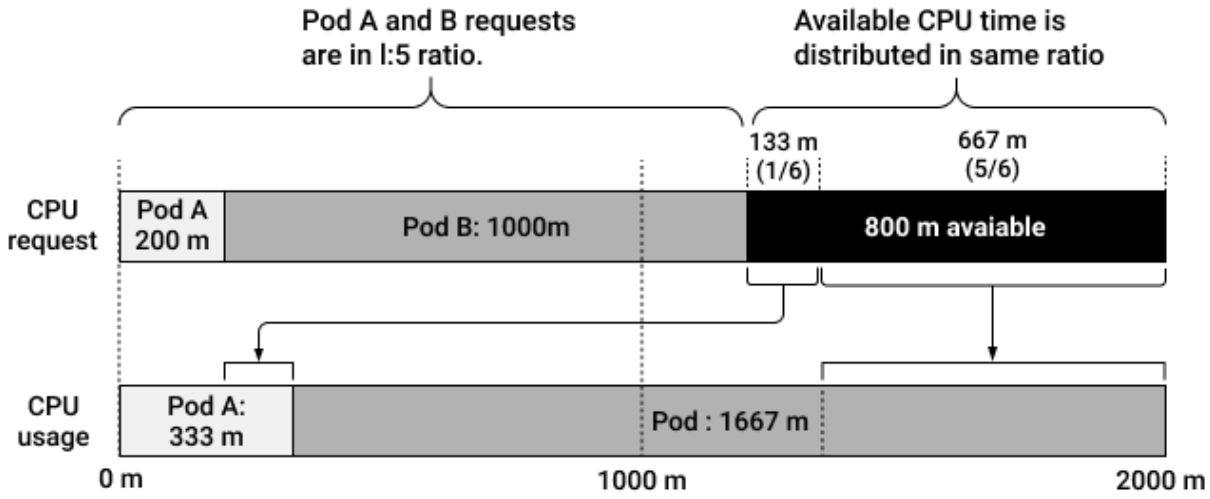
```
kubectl exec -it requests-pod -- /bin/top

Mem: 1572244K used, 163008K free, 1480K shrd, 79780K buff, 817380K
cached
CPU: 50.1% usr 49.8% sys  0.0% nic  0.0% idle  0.0% io  0.0% irq  0.0%
sirq
Load average: 8.49 3.78 1.58 5/503 38
  PID  PPID USER     STAT   VSZ %VSZ CPU %CPU COMMAND
    1     0 root     R     1296  0.0   0 99.9 dd if /dev/zero of
/dev/null
   34     0 root     R     1304  0.0   0  0.0 /bin/top
```

Это происходит потому, что мы не указали лимиты и Kubernetes позволил процессу в контейнере использовать все возможные ресурсы. Существует большая вероятность возникновения такой ситуации, когда ресурсов будет недостаточно и мы не сможем больше запускать нагрузку на узле кластера.



Запросы на CPU в свою очередь не только решают, где будет запущен pod, но и как будут утилизированы процессы при избытке ресурсов.



Лимиты на ресурсы указываются подобным образом:

```
apiVersion: v1
kind: Pod
metadata:
  name: requests-pod
spec:
  containers:
```

```
  - image: busybox
    command: ["dd", "if=/dev/zero", "of=/dev/null"]
    name: main
    resources:
      limits:
        cpu: 100m
        memory: 20Mi
```

Если не указать requests, то они будут равны limits. После применения изменения проверяем результат:

```
kubectl exec -it requests-pod -- /bin/top

Mem: 1309420K used, 425832K free, 1480K shrd, 90784K buff, 503788K
cached
CPU:  9.1% usr  7.1% sys  0.1% nic 83.4% idle  0.0% io  0.0% irq  0.0%
sirq
Load average: 0.16 0.19 0.18 2/494 11
  PID  PPID USER     STAT   VSZ %VSZ CPU %CPU COMMAND
    1     0 root     R     1296  0.0   0  9.9 dd if /dev/zero of
/dev/null
    6     0 root     R     1304  0.0   0  0.0 /bin/top
```

Полезные ссылки:

- Managing Compute Resources for Containers (official docs)
- Prevent Downtime with Proper Kubernetes Resource Planning

Задание:

1. Отредактируйте ваш deployment из предыдущего задания, добавив в него запрос на ресурсы в размере memory = 64Mi и cpu = 250m, а также лимиты в размере memory = 128Mi и cpu = 500m.
2. Примените новую конфигурацию. Команду и вывод сохраняем.
3. Ответьте на вопрос: сколько CPU мы определили как лимит для одного пода?
4. Проверьте назначение аннотации о лимите деплойменту. Команду и вывод сохраняем.
5. Пришлите в ответе все сохраненные команды и выводы, ответ на 3 пункт, а также отредактированный файл deployment.