

# Продвинутые абстракции

# План

- DaemonSet
- StatefulSet
- Job / CronJob
- Role Based Access Control (RBAC)

# Задача мониторинга

- На каждой ноде автоматически запускается агент
- Управляются агенты из одной точки
- Конфигурируются так же из одной точки



- Static Pod
- Pod Anti Affinity
- Daemon Set

# Задача мониторинга

- На каждой ноде автоматически запускается агент
- Управляются агенты из одной точки
- Конфигурируются так же из одной точки



- Static Pod (2)
- Pod Anti Affinity
- Daemon Set

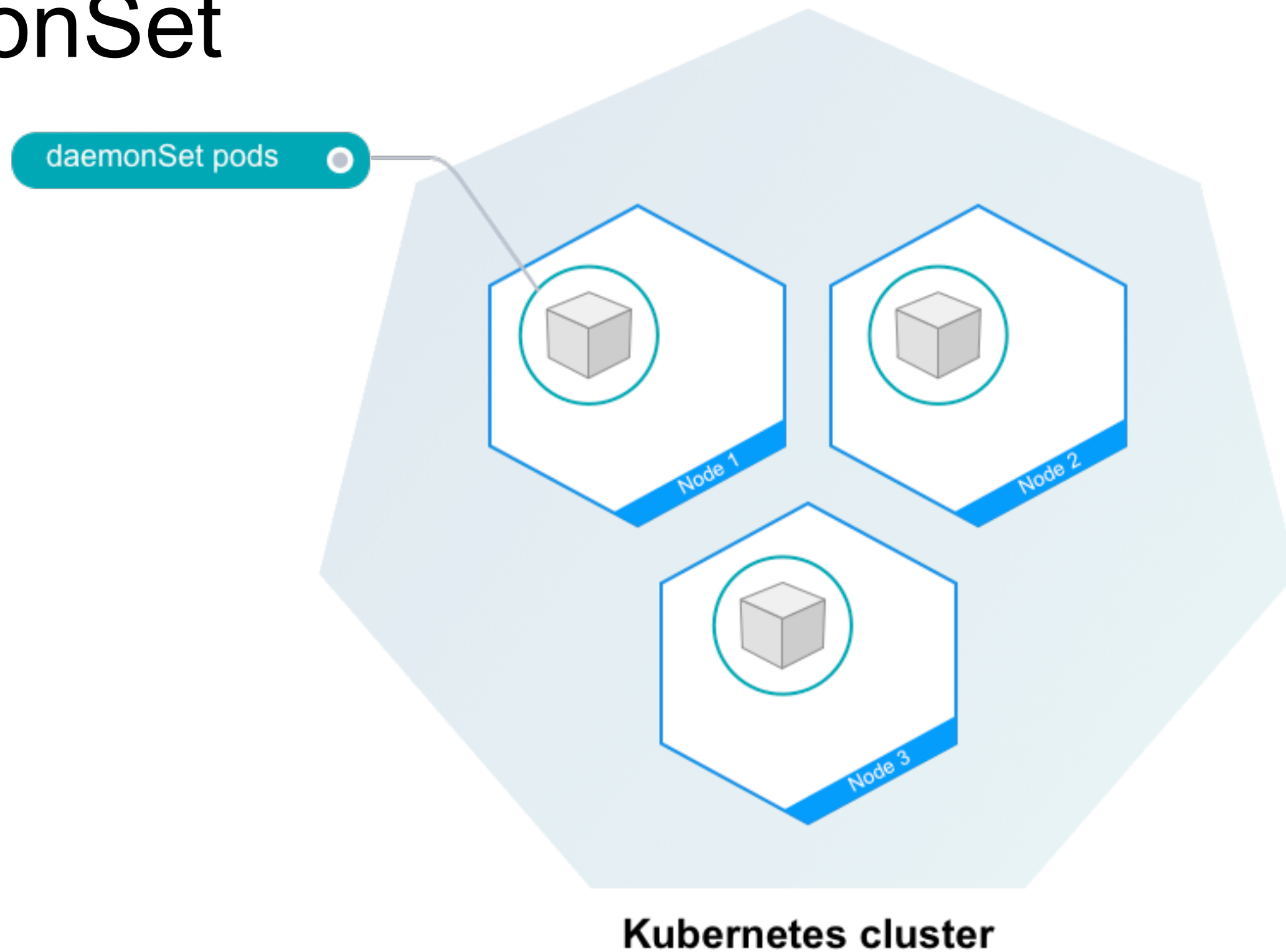
# Задача мониторинга

- На каждой ноде автоматически запускается агент
- Управляются агенты из одной точки
- Конфигурируются так же из одной точки



- Static Pod (2)
- Pod Anti Affinity (4)
- Daemon Set

# DaemonSet



# DaemonSet

- Запускает поды на **всех** нодах кластера
- При добавлении ноды – **добавляет** под
  - При удалении ноды GC **удаляет** под
- Описание практически полностью соответствует Deployment

# Tolerations

<https://kubernetes.io/docs/concepts/configuration/taint-and-toleration/>

```
$ kubectl get node master-...
```

taints:

- effect: NoSchedule

key: node-role.kubernetes.io/master

```
$ cat daemonset.yaml
```

tolerations:

- effect: NoSchedule

key: node-role.kubernetes.io/master



# Tolerations

Запустить под демонсета на всех нодах

# StatefulSet

- Позволяет запускать группу подов (как Deployment)
  - Гарантирует их **уникальность**
  - Гарантирует их **последовательность**
- PVC template
  - При удалении **не удаляет** PVC
- Используется для запуска приложений с **сохранением состояния**
  - Rabbit
  - DBs
  - Redis
  - Kafka
  - ...

# Affinity

affinity:

nodeAffinity:

requiredDuringSchedulingIgnoredDuringExecution:

nodeSelectorTerms:

- matchExpressions:

- key: `kubernetes.io/e2e-az-name`

operator: In

values:

- `e2e-az1`

- `e2e-az2`

# Affinity

affinity:

preferredDuringSchedulingIgnoredDuringExecution:

- weight: 1

preference:

matchExpressions:

- key: another-node-label-key

operator: Exists

# Headless Service

- `.spec.clusterIP: None`
- Резолвится в IP всех эндпоинтов
- Создает зону с именами всех эндпоинтов

# Job

- Создает под для выполнения задачи
- Перезапускает поды до успешного выполнения задачи
  - Или истечения таймаутов
    - `activeDeadlineSeconds`
    - `backoffLimit`

# CronJob

- Создает Job по [расписанию](#)
- Важные параметры
  - **Starting Deadline**
  - **concurrencyPolicy**

# CronJob

A cron job creates a job object *about* once per execution time of its schedule.

We say “about” because there are **certain circumstances** where two jobs might be created, or no job might be created.

We attempt to make these **rare**, **but** do not completely prevent them.



# CronJob

A cron job creates a job object *about* once per execution time of its schedule.

We say “about” because there are **certain circumstances** where two jobs might be created, or no job might be created.

We attempt to make these **rare**, **but** do not completely prevent them.



# RBAC

- Role
- RoleBinding
- ClusterRole
- ClusterRoleBinding
- ServiceAccount

# RBAC на практике

```
cd practice/advanced-abstractions/rbac
```

```
kubectl apply -f ./
```

```
kubectl get service --as=system:serviceaccount:default:user
```

```
kubectl delete service --as=system:serviceaccount:default:user kubernetes
```

# Перерыв