Running Jobs and CronJobs

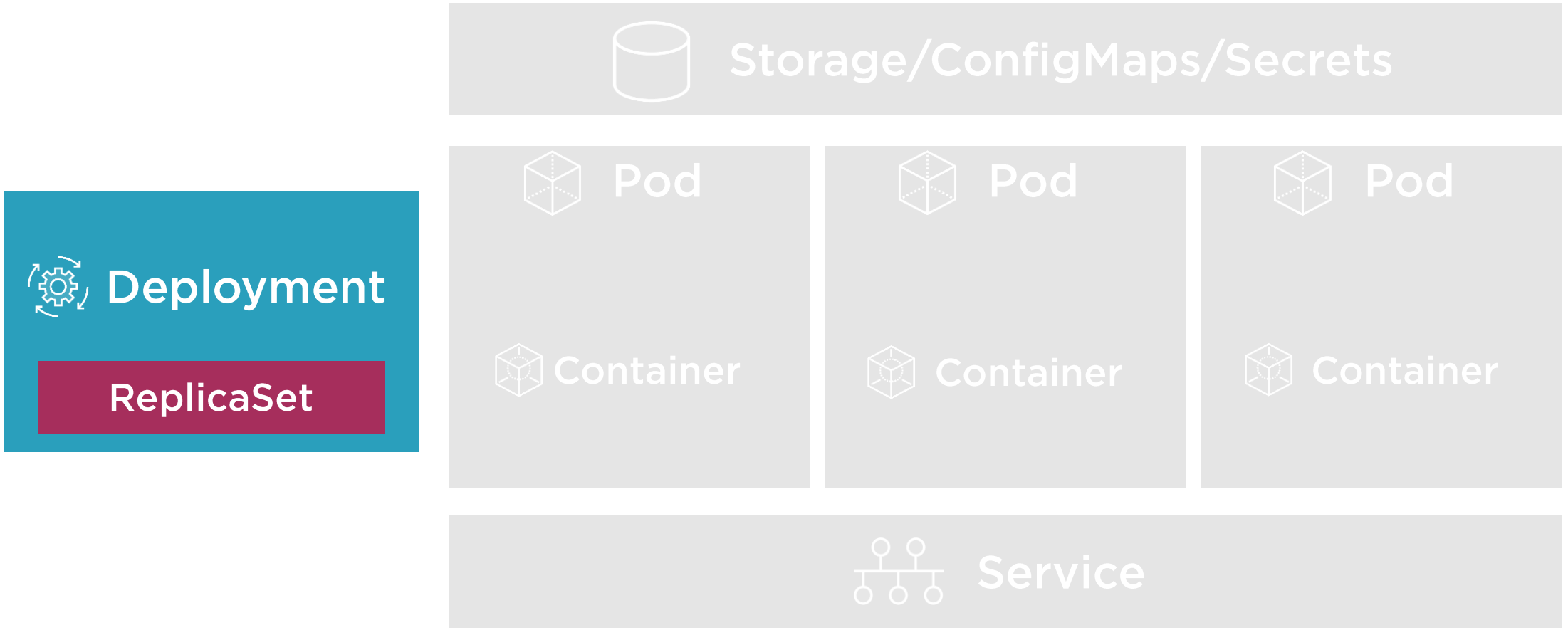
Dan Wahlin
WAHLIN CONSULTING

@danwahlin www.codewithdan.com

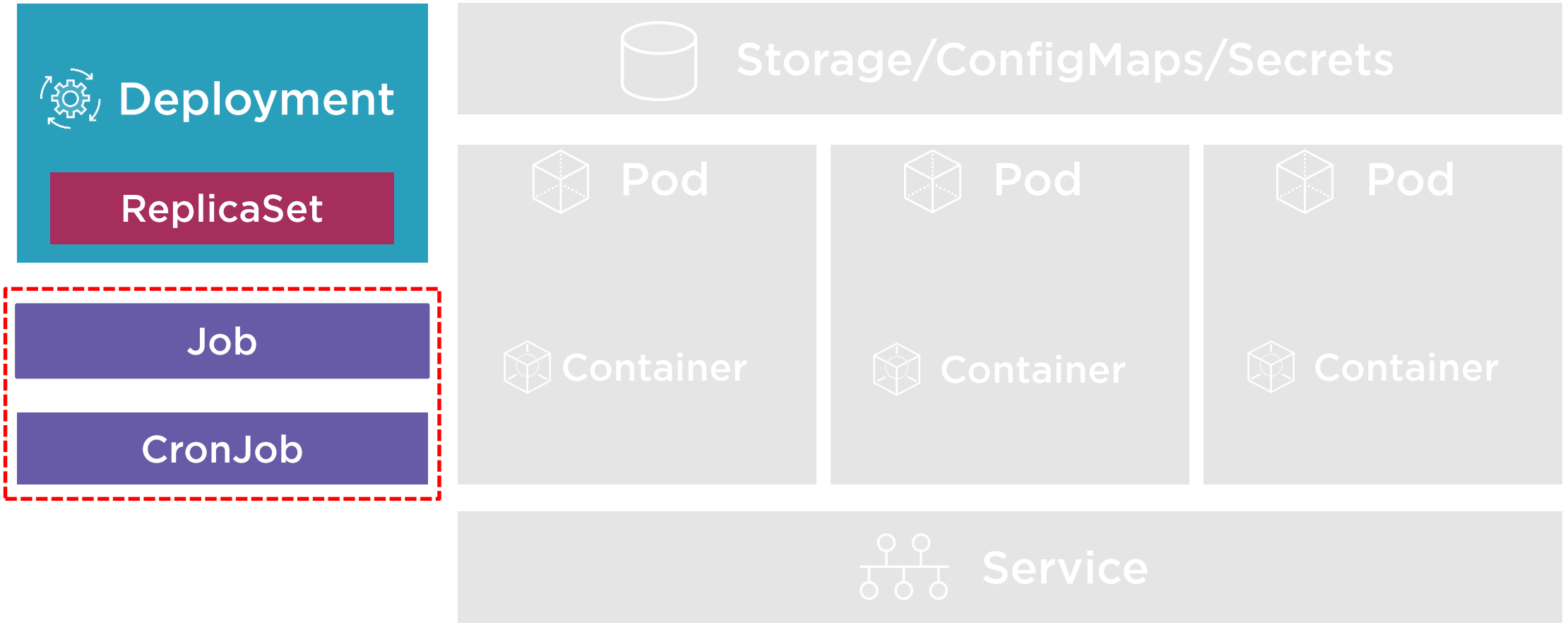
Module Overview

- Understanding Jobs
- Understanding CronJobs
- Creating a Job and CronJob
- Jobs in Action
- CronJobs in Action

Kubernetes Resources



Kubernetes Resources



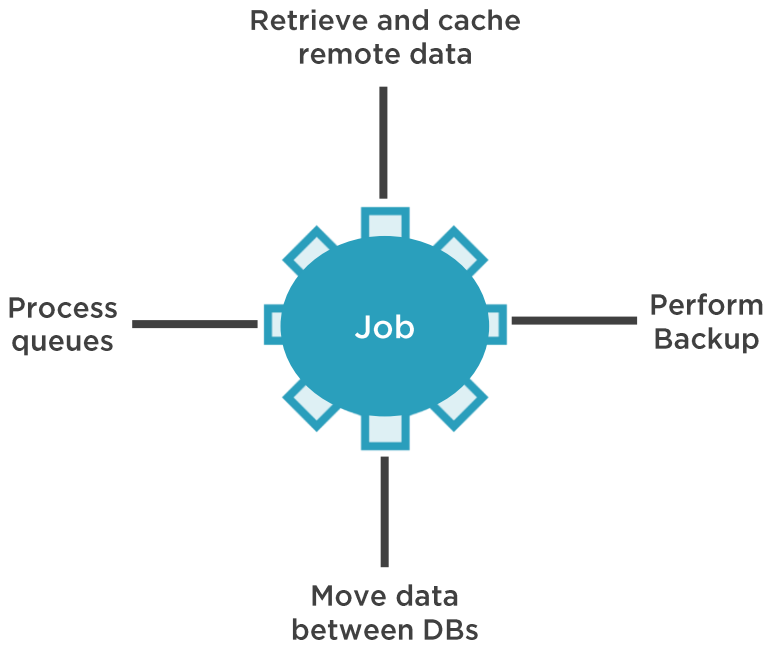
Understanding Jobs

Have you ever needed to a run a job that performs a task and then terminates?

"A Job creates one or more Pods and ensures that a specified number of them successfully terminate."

~ **Kubernetes Documentation**

Understanding Jobs



A Job creates a Pod(s) that performs a task or batch process

Unlike standard Pods, a Job does not run indefinitely

A Job can be configured to run multiple Pods in parallel

Successful completions are tracked

Once a Job is deleted its Pods are removed

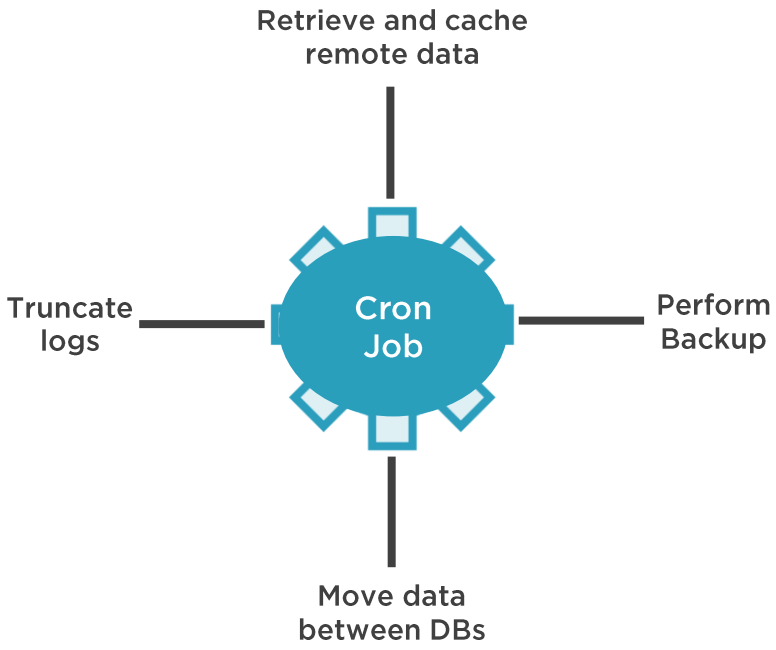
Understanding CronJobs

Have you ever needed to a run a job on a scheduled basis?

"A Cron Job creates Jobs on a time-based schedule."

~ **Kubernetes Documentation**

Understanding CronJobs

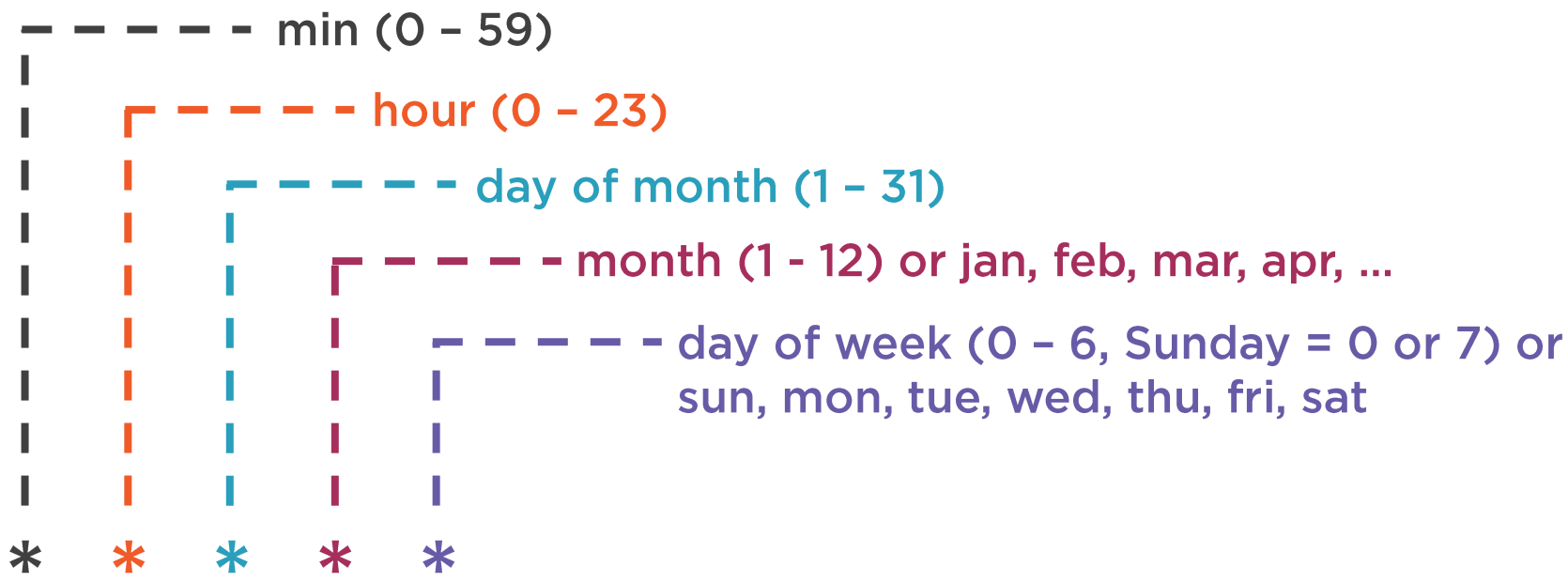


A CronJob is a Job that runs on a scheduled basis

Scheduled using the Cron format

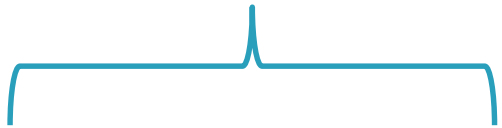
CronJob names must be less than 52 characters

Understanding the Cron Format



Cron Format Examples

Run at 22:30 every Monday



```
30  22  *  *  1
```

Run at 00:01 on the first day of each month

```
1  0  1  *  *
```

Additional Cron Formats

```
0 * * * *

0 0 * * *

0 0 * * 0

0 0 1 * *

0 0 1 1 *

*/1 * * * *
```

- **@hourly** – run once every hour
- **@daily** – run once every day at midnight
- **@weekly** – run once every week
- **@monthly** – run once every month
- **@yearly** – run once every year
- Run once every minute

Creating a Job and CronJob

Defining a Job

```
apiVersion: batch/v1
kind: Job
metadata:
  name: pie-counter
spec:
  template:
    metadata:
      name: pie-counter
    spec:
      restartPolicy: Never
      containers:
      - name: pie-counter
        image: alpine
        command:
         - "sh"
         - "-c"
         - "echo 'scale=1000; 4*a(1)' …;"
```

- Batch API
- Job kind
- Never try to restart (Never or OnFailure)
- Job command to run

Defining a Job that Requires Multiple Completions

```
apiVersion: batch/v1
kind: Job
metadata:
  name: pie-counter
spec:
  completions: 4
  template:
    ...
```

- Run 4 Pods sequentially

Defining a Job that Can Run in Parallel

```
apiVersion: batch/v1
kind: Job
metadata:
  name: pie-counter
spec:
  completions: 4
  parallelism: 2
  template:
    ...
```

- 4 Pods must complete successfully
- 2 Pods can run in parallel at a time

Creating a Job

A job can be created using the standard kubectl create or kubectl apply commands

```
# Create a new Job
kubectl create –f file.job.yml --save-config

# Creating or modifying a Job
kubectl apply –f file.job.yml
```

Defining a CronJob

```
apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: pie-counter
spec:
  concurrencyPolicy: Allow
  # Run the job every minute
  schedule: "*/1 * * * *"
  jobTemplate:
    spec:
      template:
        spec:
          restartPolicy: OnFailure
          containers:
          - name: pie-counter
            image: alpine
            command:
            - "sh"
            - "-c"
            - "echo 'scale=1000; 4*a(1)'
```

- CronJob batch API
- CronJob kind

- Allow multiple Pods to run event if their scheduling overlaps
- Cron format to use for scheduling

- Restart if there's a failure

- Command to run

Creating a CronJob

A CronJob can be created using the standard kubectl create or kubectl apply commands

```
# Create a new CronJob
kubectl create –f file.cronjob.yml --save-config

# Creating or modifying a CronJob
kubectl apply –f file.cronjob.yml
```

Jobs in Action

CronJobs in Action

Summary

- Jobs are used to run a task or batch process
- Successful completions are tracked
- CronJobs allow a task/batch process to be run on a scheduled basis
- Relies on the Cron format