Day 3

NobleProg

Day 3

- Configmaps & Secrets
- Monitoring
- Capstone project
- Wrap Up & Questions

NobleProg

Day 3

NobleProg

Day 3

(c) https://www.reddit.com/r/ProgrammerHumor/comments/143274m/yet_another_kubernetes_meme_yakm/

NobleProg

cd openshift/Day3/config

Container config

NobleProg

Container config

cd openshift/Day3/config

```yaml
basic-statefulset.yaml

apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mysql
spec:
  serviceName: mysql
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
      - name: mysql
        image: mysql:latest
        env:
        - name: MYSQL_ROOT_PASSWORD
          value: "supersecurepassword"
        - name: MYSQL_USER
          value: "myuser"
        - name: MYSQL_PASSWORD
          value: "mypassword"
        - name: MYSQL_DATABASE
          value: "mydatabase"
        ports:
        - containerPort: 3306
```

NobleProg

Configmaps

- Key Value storage
- Text-Data storage

NobleProg

Configmaps

- Key Value storage
- Text-Data storage

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-keyvalue-config
data:
  APP_ENV: "production"
  DEBUG_MODE: "false"
```

NobleProg

Configmaps

- Key Value storage
- Text-Data storage

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-keyvalue-config
data:
  APP_ENV: "production"
  DEBUG_MODE: "false"
```

Configmaps

- Key Value storage
- Text-Data storage

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-keyvalue-config
data:
  APP_ENV: "production"
  DEBUG_MODE: "false"
```

```yaml
Deployment.yaml

...
spec:
  containers:
      - name: myapp
        image: myregistry/myapp:latest
        envFrom:
          - configMapRef:
              name: my-keyvalue-config
...
```

Configmaps

- Key Value storage
- Text-Data storage

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-keyvalue-config
data:
  APP_ENV: "production"
  DEBUG_MODE: "false"
```

```
Deployment.yaml

...
spec:
  containers:
    - name: myapp
      image: myregistry/myapp:latest
      envFrom:
        - configMapRef:
            name: my-keyvalue-config
...
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-config
data:
  config.json: |
    {
      "app_env": "production",
      "debug": false
    }
  settings.yaml: |
    debug: false
    app_env: production
```

Configmaps

- Key Value storage
- Text-Data storage

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-keyvalue-config
data:
  APP_ENV: "production"
  DEBUG_MODE: "false"
```

```
Deployment.yaml

...
spec:
  containers:
     - name: myapp
       image: myregistry/myapp:latest
       envFrom:
         - configMapRef:
             name: my-keyvalue-config
...
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-config
data:
  config.json: |
    {
      "app_env": "production",
      "debug": false
    }
  settings.yaml: |
    debug: false
    app_env: production
```

NobleProg

Configmaps

- Key Value storage
- Text-Data storage

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-keyvalue-config
data:
  APP_ENV: "production"
  DEBUG_MODE: "false"
```

```yaml
Deployment.yaml

…
spec:
  containers:
     - name: myapp
       image: myregistry/myapp:latest
       envFrom:
         - configMapRef:
             name: my-keyvalue-config
…
```

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-config
data:
  config.json: |
    {
      "app_env": "production",
      "debug": false
    }
  settings.yaml: |
    debug: false
    app_env: production
```

```yaml
Deployment.yaml

…
spec:
      containers:
      - name: myapp
        image: myregistry/myapp:latest
        volumeMounts:
          - name: config-volume
            mountPath: /etc/config
      volumes:
        - name: config-volume
          configMap:
            name: my-config
…
```

Configmaps

- Key Value storage
- Text-Data storage

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-keyvalue-config
data:
  APP_ENV: "production"
  DEBUG_MODE: "false"
```

```yaml
Deployment.yaml

…
spec:
  containers:
     - name: myapp
       image: myregistry/myapp:latest
       envFrom:
         - configMapRef:
             name: my-keyvalue-config
…
```

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-config
data:
  config.json: |
    {
      "app_env": "production",
      "debug": false
    }
  settings.yaml: |
    debug: false
    app_env: production
```

```yaml
Deployment.yaml

…
spec:
      containers:
      - name: myapp
        image: myregistry/myapp:latest
        volumeMounts:
          - name: config-volume
            mountPath: /etc/config
      volumes:
        - name: config-volume
          configMap:
            name: my-config
…
```

```yaml
Deployment.yaml

…
spec:
      containers:
      - name: myapp
        image: myregistry/myapp:latest
        volumeMounts:
          - name: config-volume
            mountPath: /etc/config
            subPath: diffconfigname.json
      volumes:
        - name: config-volume
          configMap:
            name: my-config
            items:
              - key: config.json
                path: config.json
…
```

Configmaps

- Key Value storage
- Text-Data storage

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-keyvalue-config
data:
  APP_ENV: "production"
  DEBUG_MODE: "false"
```

```
Deployment.yaml

…
spec:
  containers:
     - name: myapp
       image: myregistry/myapp:latest
       envFrom:
         - configMapRef:
             name: my-keyvalue-config
…
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-config
data:
  config.json: |
    {
      "app_env": "production",
      "debug": false
    }
  settings.yaml: |
    debug: false
    app_env: production
```

```
Deployment.yaml

…
spec:
      containers:
      - name: myapp
        image: myregistry/myapp:latest
        volumeMounts:
          - name: config-volume
            mountPath: /etc/config
      volumes:
        - name: config-volume
          configMap:
            name: my-config
…
```

```
Deployment.yaml

…
spec:
      containers:
      - name: myapp
        image: myregistry/myapp:latest
        volumeMounts:
          - name: config-volume
            mountPath: /etc/config
            subPath: diffconfigname.json
      volumes:
        - name: config-volume
          configMap:
            name: my-config
            items:
              - key: config.json
                path: config.json
…
```

```
Deployment.yaml

…
spec:
      containers:
      - name: my-container
        image: my-image:latest
        envFrom:
        - configMapRef:
            name: my-configmap
        - secretRef:
            name: my-secret
        env:
        - name: SPECIAL_VAR
          value: "manual-override"
        - name: MYSQL_DATABASE
          value: „custom-db"
…
```

Secrets (very secure)

- Key Value storage
- Text-Data storage

Secrets (very secure)

- Key Value storage
- Text-Data storage

```
Oc create secret generic my-secret \
  --from-literal=DB_USER=username \
  --from-literal=DB_PASSWORD=password
```

NobleProg

Secrets (very secure)

- Key Value storage
- Text-Data storage

```
Oc create secret generic my-secret \
  --from-literal=DB_USER=username \
  --from-literal=DB_PASSWORD=password
```

```
apiVersion: v1
kind: Secret
metadata:
  name: my-secret
type: Opaque
data:
  DB_USER: dXNlcm5hbWU=
  DB_PASSWORD: cGFzc3dvcmQ=
```

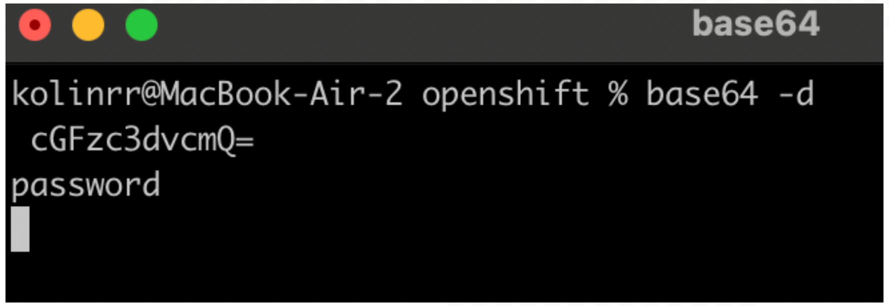
NobleProg

Secrets (very secure)

- Key Value storage
- Text-Data storage

```
Oc create secret generic my-secret \
  --from-literal=DB_USER=username \
  --from-literal=DB_PASSWORD=password
```

```
apiVersion: v1
kind: Secret
metadata:
  name: my-secret
type: Opaque
data:
  DB_USER: dXNlcm5hbWU=
  DB_PASSWORD: cGFzc3dvcmQ=
```



```
kolinrr@MacBook-Air-2 openshift % base64 -d
 cGFzc3dvcmQ=
password
```

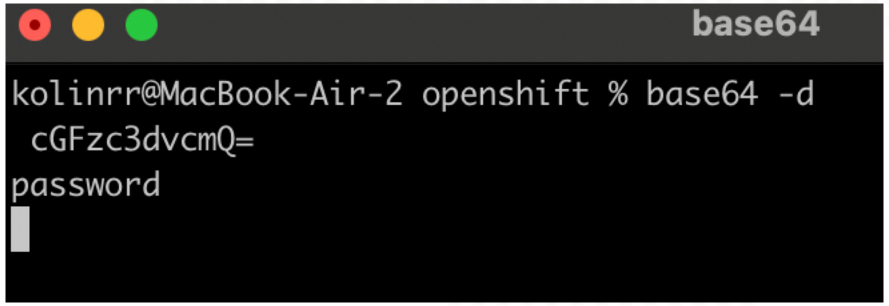
NobleProg

Secrets (very secure)

- Key Value storage
- Text-Data storage

```
Oc create secret generic my-secret \
  --from-literal=DB_USER=username \
  --from-literal=DB_PASSWORD=password
```

```
apiVersion: v1
kind: Secret
metadata:
  name: my-secret
type: Opaque
data:
  DB_USER: dXNlcm5hbWU=
  DB_PASSWORD: cGFzc3dvcmQ=
```



base64

```
kolinrr@MacBook-Air-2 openshift % base64 -d
 cGFzc3dvcmQ=
password
```

NobleProg

Secrets (very secure)

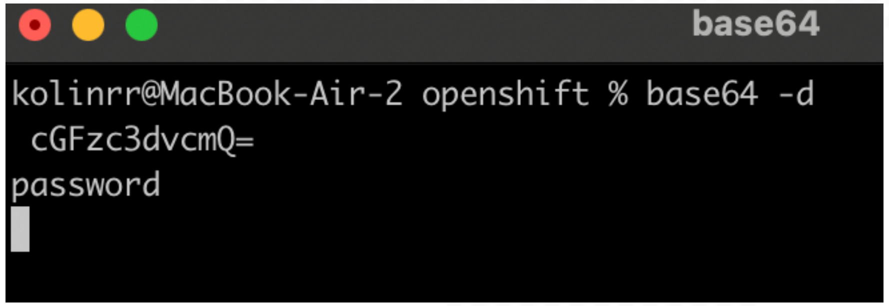
- Key Value storage
- Text-Data storage

```
Oc create secret generic my-secret \
  --from-literal=DB_USER=username \
  --from-literal=DB_PASSWORD=password
```

```
apiVersion: v1
kind: Secret
metadata:
  name: my-secret
type: Opaque
data:
  DB_USER: dXNlcm5hbWU=
  DB_PASSWORD: cGFzc3dvcmQ=
```

```
Deployment.yaml

…
spec:
  containers:
     - name: myapp
       image: myregistry/myapp:latest
       envFrom:
        - secretRef:
           name: my-secret
…
```



```
kolinrr@MacBook-Air-2 openshift % base64 -d
 cGFzc3dvcmQ=
password
```

NobleProg

Secrets (very secure)

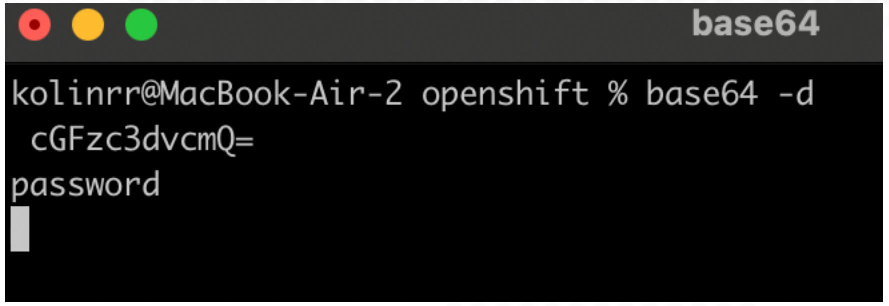
- Key Value storage
- Text-Data storage

```
Oc create secret generic my-secret \
  --from-literal=DB_USER=username \
  --from-literal=DB_PASSWORD=password
```

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: my-secret
type: Opaque
data:
  DB_USER: dXNlcm5hbWU=
  DB_PASSWORD: cGFzc3dvcmQ=
```

```yaml
Deployment.yaml

…
spec:
  containers:
      - name: myapp
        image: myregistry/myapp:latest
        envFrom:
          - secretRef:
              name: my-secret
…
```



```
kolinrr@MacBook-Air-2 openshift % base64 -d
 cGFzc3dvcmQ=
password
```

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: my-secret
type: Opaque
data:
  db.conf: |
    ZGJfdXNlcj11c2VybmFtZQpkYl9wYXNzd29yZD1wYXNzd29yZA==
```

Secrets (very secure)

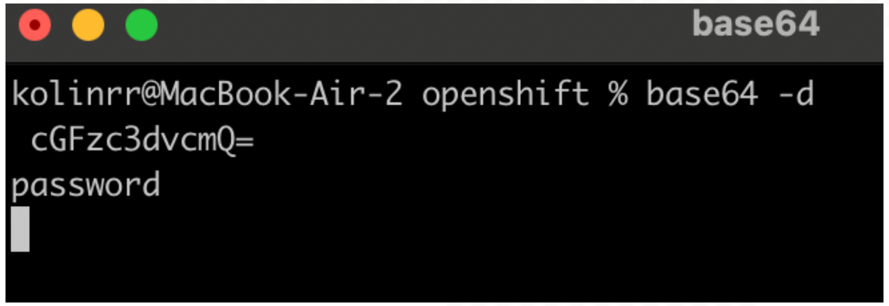
- Key Value storage
- Text-Data storage

```
Oc create secret generic my-secret \
  --from-literal=DB_USER=username \
  --from-literal=DB_PASSWORD=password
```

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: my-secret
type: Opaque
data:
  DB_USER: dXNlcm5hbWU=
  DB_PASSWORD: cGFzc3dvcmQ=
```

```yaml
Deployment.yaml

…
spec:
  containers:
     - name: myapp
       image: myregistry/myapp:latest
       envFrom:
         - secretRef:
             name: my-secret
…
```



```
kolinrr@MacBook-Air-2 openshift % base64 -d
 cGFzc3dvcmQ=
password
```

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: my-secret
type: Opaque
data:
  db.conf: |
    ZGJfdXNlcj11c2VybmFtZQpkYl9wYXNzd29yZD1wYXNzd29yZA==
```

NobleProg

Secrets (very secure)

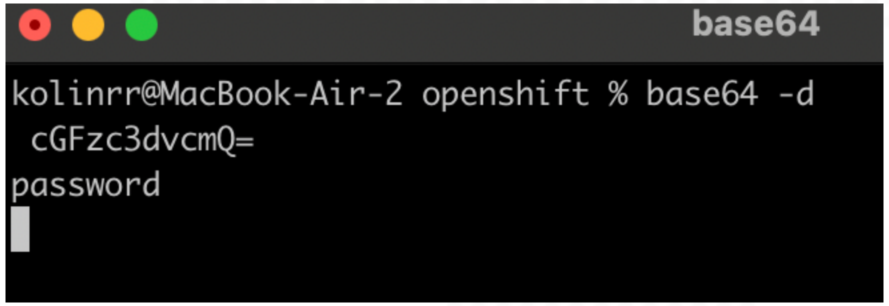
- Key Value storage
- Text-Data storage

```
Oc create secret generic my-secret \
  --from-literal=DB_USER=username \
  --from-literal=DB_PASSWORD=password
```

```
apiVersion: v1
kind: Secret
metadata:
  name: my-secret
type: Opaque
data:
  DB_USER: dXNlcm5hbWU=
  DB_PASSWORD: cGFzc3dvcmQ=
```

```
Deployment.yaml

…
spec:
  containers:
    - name: myapp
      image: myregistry/myapp:latest
      envFrom:
        - secretRef:
            name: my-secret
…
```



```
kolinrr@MacBook-Air-2 openshift % base64 -d
 cGFzc3dvcmQ=
password
```

```
apiVersion: v1
kind: Secret
metadata:
  name: my-secret
type: Opaque
data:
  db.conf: |
    ZGJfdXNlcj11c2VybmFtZQpkYl9wYXNzd29yZD1wYXNzd29yZA==
```

```
Deployment.yaml

…
spec:
     containers:
     - name: myapp
       image: myregistry/myapp:latest
       volumeMounts:
         - name: secret-volume
           mountPath: /etc/secret
           readOnly: true
     volumes:
       - name: secret-volume
         secret:
           secretName: my-secret
…
```

```
cd openshift/Day3/config
```

Config + Secrets Lab

NobleProg

cd openshift/Day3/config

Config + Secrets Lab

```
$ cd $HOME/openshift/Day3/config

//Compare the differences between 1. to 3.

$ oc apply -f 1.basic-statefulset.yaml

$ oc apply -f 2.configmaps-as-volume.yaml
$ oc apply -f 2.statefulset-with-config-as-vol.yaml

$ oc apply -f 3.configmap.yaml
$ oc apply -f 3.secret.yaml
$ oc apply -f 3.statefulset.yaml
```

NobleProg

Monitoring (Show)

- Openshift GUI internal Monitoring and Logs
- Kubernetes Container Log & Object-Events
- Openshift Observe

NobleProg

Monitoring Lab

```
cd openshift/Day3/monitoring
```

NobleProg

cd openshift/Day3/monitoring

Monitoring Lab

```
$ cd $HOME/openshift/Day3/monitoring

//Compare the differences between 1. to 3.

$ oc apply -f deployment.yaml
$ oc apply -f service.yaml
$ oc apply -f servicemonitor.yaml

$ oc label namespace default openshift.io/cluster-monitoring=true
```

NobleProg

cd openshift/Day3/monitoring

Monitoring Lab

```
$ cd $HOME/openshift/Day3/monitoring

//Compare the differences between 1. to 3.

$ oc apply -f deployment.yaml
$ oc apply -f service.yaml
$ oc apply -f servicemonitor.yaml

$ oc label namespace default openshift.io/cluster-monitoring=true
```

Open in Browser: https://console-openshift-console.apps-crc.testing/monitoring/targets

NobleProg

cd openshift/Day3/monitoring

Monitoring Lab

```
$ cd $HOME/openshift/Day3/monitoring

//Compare the differences between 1. to 3.

$ oc apply -f deployment.yaml
$ oc apply -f service.yaml
$ oc apply -f servicemonitor.yaml

$ oc label namespace default openshift.io/cluster-monitoring=true
```

Open in Browser: https://console-openshift-console.apps-crc.testing/monitoring/targets

Open in Browser: https://console-openshift-console.apps-crc.testing/monitoring/query-browser?query0=

NobleProg

cd openshift/Day3/monitoring

Monitoring Lab

```
$ cd $HOME/openshift/Day3/monitoring

//Compare the differences between 1. to 3.

$ oc apply -f deployment.yaml
$ oc apply -f service.yaml
$ oc apply -f servicemonitor.yaml

$ oc label namespace default openshift.io/cluster-monitoring=true
```

Open in Browser: https://console-openshift-console.apps-crc.testing/monitoring/targets

Open in Browser: https://console-openshift-console.apps-crc.testing/monitoring/query-browser?query0=

Add expression: rate(random_metric[5m])

NobleProg

Security Best Practices

- SecurityContextConstraints
  - RunAsUser: 1234 or Range / MustRunAsNonRoot (UserID != 0)
  - FSGroup: 1234 - access only to files with defined GID

Security Best Practices

- SecurityContextConstraints
  - RunAsUser: 1234 or Range / MustRunAsNonRoot (UserID != 0)
  - FSGroup: 1234 - access only to files with defined GID

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  replicas: 3
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      securityContext:
        runAsUser: 1001
        runAsUser: MustRunAsNonRoot
        runAsGroup: 3000
        fsGroup: 2000
      containers:
        - name: my-app
          image: my-app:v2
          securityContext:
            allowPrivilegeEscalation: false
            runAsNonRoot: true
```

cd openshift/Day3/security

Security Best Practices

- RBAC - Role Based Access Control

- NetworkPolicies (Advanced)

NobleProg

cd openshift/Day3/security

Security Best Practices

- RBAC - Role Based Access Control

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: pod-reader
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["list"]
- apiGroups: ["apps"]
  resources: ["deployments"]
  verbs: ["list"]
```

- NetworkPolicies (Advanced)

NobleProg

cd openshift/Day3/security

Security Best Practices

- RBAC - Role Based Access Control

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: pod-reader
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["list"]
- apiGroups: ["apps"]
  resources: ["deployments"]
  verbs: ["list"]
```

Verbs:
- get
- list
- create
- update

- NetworkPolicies (Advanced)

NobleProg

cd openshift/Day3/security

Security Best Practices

- RBAC - Role Based Access Control

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: pod-reader
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["list"]
- apiGroups: ["apps"]
  resources: ["deployments"]
  verbs: ["list"]
```

See commands.txt:

Verbs:
- get
- list
- create
- update

- NetworkPolicies (Advanced)

NobleProg

cd openshift/Day3/security

Security Best Practices

- RBAC - Role Based Access Control

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: pod-reader
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["list"]
- apiGroups: ["apps"]
  resources: ["deployments"]
  verbs: ["list"]
```

See commands.txt:

```
oc create serviceaccount pod-reader
```

Verbs:
- get
- list
- create
- update

- NetworkPolicies (Advanced)

NobleProg

cd openshift/Day3/security

Security Best Practices

- RBAC - Role Based Access Control

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: pod-reader
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["list"]
- apiGroups: ["apps"]
  resources: ["deployments"]
  verbs: ["list"]
```

See commands.txt:

```
oc create serviceaccount pod-reader

oc apply -f role.yaml -n NAMESPACE
```

Verbs:
- get
- list
- create
- update

- NetworkPolicies (Advanced)

NobleProg

cd openshift/Day3/security

Security Best Practices

- RBAC - Role Based Access Control

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: pod-reader
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["list"]
- apiGroups: ["apps"]
  resources: ["deployments"]
  verbs: ["list"]
```

See commands.txt:

```
oc create serviceaccount pod-reader

oc apply -f role.yaml -n NAMESPACE


oc create rolebinding pod-reader-binding \
  --role=pod-reader \
  --serviceaccount=NAMESPACE:pod-reader
```

Verbs:
- get
- list
- create
- update

- NetworkPolicies (Advanced)

NobleProg

cd openshift/Day3/security

Security Best Practices

- RBAC - Role Based Access Control

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: pod-reader
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["list"]
- apiGroups: ["apps"]
  resources: ["deployments"]
  verbs: ["list"]
```

Verbs:
- get
- list
- create
- update

See commands.txt:

```
oc create serviceaccount pod-reader

oc apply -f role.yaml -n NAMESPACE


oc create rolebinding pod-reader-binding \
  --role=pod-reader \
  --serviceaccount=NAMESPACE:pod-reader


oc create token pod-reader -n NAMESPACE
```

- NetworkPolicies (Advanced)

NobleProg

cd openshift/Day3/security

Security Best Practices

- RBAC - Role Based Access Control

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: pod-reader
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["list"]
- apiGroups: ["apps"]
  resources: ["deployments"]
  verbs: ["list"]
```

Verbs:
- get
- list
- create
- update

See commands.txt:

```
oc create serviceaccount pod-reader

oc apply -f role.yaml -n NAMESPACE

oc create rolebinding pod-reader-binding \
  --role=pod-reader \
  --serviceaccount=NAMESPACE:pod-reader

oc create token pod-reader -n NAMESPACE
oc login --token=$TOKEN
```

- NetworkPolicies (Advanced)

NobleProg

Lunch Break: 30 minutes

cd openshift/Day3/capstone

Capstone Project Lab

- App & Database Deployment
- Create new Workflow -> copy&paste from openshift/Day3/capstone/workflow/openshift.yaml
- Goal of Capstone
  - Kustomize name of deployments, labels, service and route
  - Kustomize ENV-Var of container
  - Activate Canary Route

NobleProg

cd openshift/Day3/capstone

Capstone Project Lab

- Run canary-test:
  copy the canary route „URL“ from Openshift and place in test.sh

  ```
  $ cd Day3/capstone/canary
  $ nano test.sh
  ```

  Paste the URL on line 3
  Save with [Ctrl+o] and close with [Ctrl+x]

  Run the script:
  ```
  $ ./test.sh
  ```

Capstone Project

- What would be the difference by „Blue/Green“ implementation?

NobleProg

Course Wrap-Up & Feedback

NobleProg

Thank you!

Trainer: Alexander Kolin

E-Mail: a@kolin.pro

LinkedIn: https://www.linkedin.com/in/alexander-kolin/

https://www.nobleprog.de/

NobleProg