redhat.

OpenShift - Access Denied 2018

DevConf, Brno, 2018

Jan Wozniak
Software Engineer

Josef Karasek
Software Engineer

Linux

Docker

File permissions

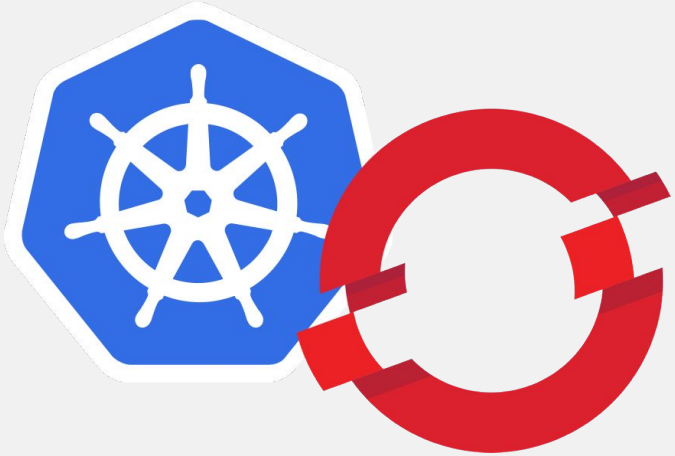
SELinux

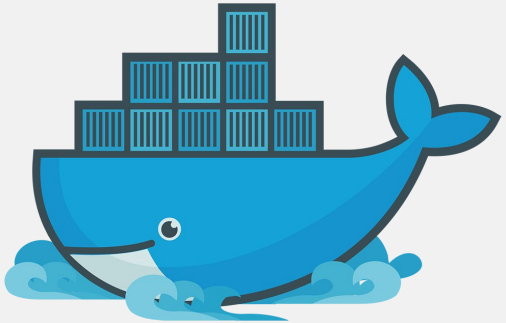
Kubernetes

Security Context Constraints

OpenShift

Persistent Volumes





Docker, Kubernetes and OpenShift

Docker

- Encapsulated app with its dependencies
- Easy to run (container) and build / share (image)

Kubernetes

- Orchestration, span containers across multiple machines

OpenShift

- RedHat flavored Kubernetes with additional features (multitenancy, security ...)

Linux - Users & File Permissions

Host filesystem

```
# id
uid=1009(tst1) gid=100(users) groups=100(users)
# ls -l
dr-xr-x---.   9 root  root   280 Jan 17 13:14 root
lrwxrwxrwx.   1 tst1 users    8 Dec 19 03:25 fldr1 -> xyz/fldr
-rwxr-xr-x.  30 tst2 wheel 1020 Jan 18 18:30 awesome file
```

Container filesystem

```
$ id
uid=1000160000 gid=0(root) groups=0(root)
$ ls -l
dr-xr-x---.   9 root  root   280 Jan 17 13:14 root
lrwxrwxrwx.   1 tst1 users    8 Dec 19 03:25 fldr1 -> xyz/fldr
-rwxr-xr-x.  30 tst2 wheel 1020 Jan 18 18:30 awesome file
```

Security Enhanced Linux

- fine grained control over objects in Linux
- set of rules defining who can access what

```
# ls -lFnZa
-rw-r--r--. usr1 users  system_u:object_r:svirt_sandbox_file_t:s0:c0,c12 f
```

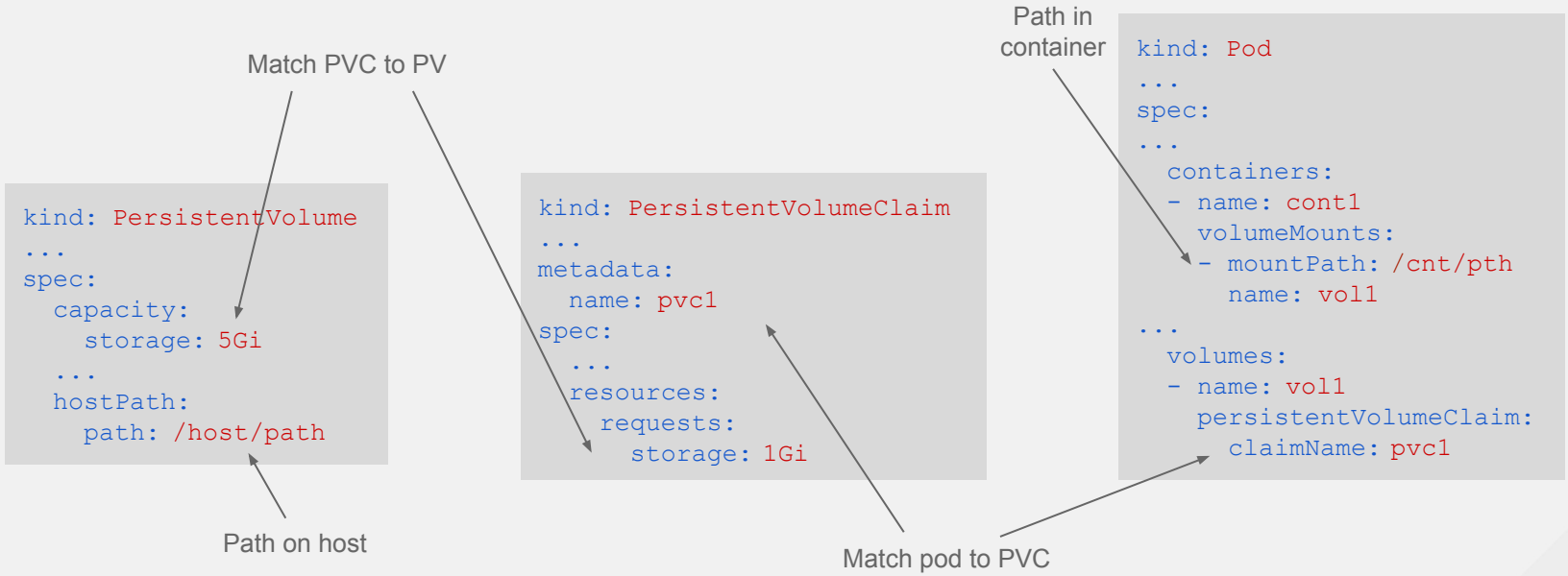
Security Context

1. **User:** Typically available `user_u (logged user)`, `system_u (system process)`, `root`
2. **Role:** Placeholder for files, grouping of policies for processes (RBAC)
3. **Type:** Finer grained control over what processes have access to which objects
4. **Level:** Optional field, higher level dominates lower
5. **Categories**: Optional field, addition for containers. Isolation of objects host shares with specific container

```
# oc get deployment [d] -o yaml
   container:
     securityContext:
       seLinuxOptions:
         level: " s0:c0,c12"
```

Storage

Pods are stateless -> Storage maintains state
Persistent Volume vs. “bare hostPath”



User Control and Management

User vs. Service Account
- pod vs. deployment

```
# oc whoami
# system:serviceaccount: [project]:[name]     # remember 'default' name
```

Security Context Constraints
- additional control to RBAC over actions pod can perform
- "SCCs are also very useful for managing access to persistent storage" - OpenShift documentation

```
# oc get scc
NAME                    SELINUX     RUNASUSER          VOLUMES
hostaccess              MustRunAs    MustRunAsRange     [..., hostPath, ...]
hostmount-anyuid        MustRunAs    RunAsAny           [..., hostPath, ...]
privileged              RunAsAny     RunAsAny           [..., hostPath, ...]
# oc adm policy add-scc-to-user [scc] system:serviceaccount: [pr]:default
```

Useful Commands & Information

```
$ oc get pod
$ oc get pod [pod] -o yaml
$ oc describe pod [pod]
$ oc edit pod [pod]

$ oc get deployment
$ oc get rs/replicaset
$ oc get pv/persistentvolume
$ oc get pvc/persistentvolumeclaim

$ oc logs [pod]
$ oc rsh [pod]
$ oc replace --force -f [file.yaml]

$ oc adm policy add-scc-to-user [scc] [user]
```

```
$ ssh [user]@[host]
$ id

$ check_results
$ retry [task]

$ ls -lFnZa
$ chmod [mod] [file]
$ sudo chcon -u [user] -r [role] -t [type] [file]
$ sudo chcat -- [+/-][cat] [file]

SCC: hostaccess, hostmount-anyuid, privileged
SELinux: system_u:object_r:svirt_sandbox_file_t

SA: system:serviceaccount:[project]:default
```

Task 1 - hint

```
# oc logs task1
/bin/sh: /pv/file: Permission denied
# oc rsh task1
$ id
uid=10000X0000 gid=0(root) groups=0(root),10000X0000
$ ls -la /pv/
drwxr-xr-x.    2 root   root    18 Jan 24 13:14 .
-rw-r--r--.   30 100Y   root     0 Jan 24 13:14 file
```

Task 1 - solution

```
# oc logs task1
/bin/sh: /pv/file: Permission denied
# oc rsh task1
$ id
uid=10000X0000 gid=0(root) groups=0(root),10000X0000
$ ls -la /pv/
drwxr-xr-x.    2 root   root    18 Jan 24 13:14 .
-rw-r--r--.   30 100Y   root     0 Jan 24 13:14 file
```

```
# chmod 664 ~/pv/1/file
# retry task1
# check_results
```

Task 2 - hint

```
# oc describe replicaset task2-*
...
unable to validate against any security context constraint:
...
"hostPath": hostPath volumes are not allowed to be used
# oc get scc
NAME            ...           VOLUMES
...
hostaccess      ...           [... hostPath ...]
```

Task 2 - solution

```
# oc describe replicaset task2-*
...
unable to validate against any security context constraint:
...
"hostPath": hostPath volumes are not allowed to be used
# oc get scc
NAME              ...             VOLUMES
...
hostaccess        ...             [... hostPath ...]
```

```
# oc adm policy add-scc-to-user hostmount-anyuid \
    system:serviceaccount:${user}-project:default
# retry task2
# check_results
```

Task 3 - hint

```
# retry task3
# oc logs task3-*
# ls -lFnZa ~/pv/3/
-rw-rw-r--. system_u:object_r:svirt_sandbox_file_t:s0 file
drwxr-xr-x. system_u:object_r:svirt_sandbox_file_t:s0:c0,c5 ./
# oc get deployment -o yaml
...
securityContext: {}
```

Task 3 - solution

```
# retry task3
# oc logs task3-*
# ls -lFnZa ~/pv/3/
-rw-rw-r--. system_u:object_r:svirt_sandbox_file_t:s0 file
drwxr-xr-x. system_u:object_r:svirt_sandbox_file_t:s0:c0,c5 ./
# oc get deployment -o yaml
...
securityContext: {}
```

```
# oc edit deployment task3
   securityContext:
     seLinuxOptions:
       level: "s0:c0,c5"
# oc adm policy add-scc-to-user privileged \
    system:serviceaccount:${user}-project:default
```

Task 4 - hint

```
# ls -ltrFnZa ~/pv/4/
-rw-rw-r--. 1002 0 unconfined_u:object_r:user_home_t:s0 file
drwxr-xr-x.    0 0 unconfined_u:object_r:user_home_t:s0 ./
```

Task 4 - solution

```
# ls -ltrFnZa ~/pv/4/
-rw-rw-r--. 1002 0 unconfined_u:object_r:user_home_t:s0 file
drwxr-xr-x.    0 0 unconfined_u:object_r:user_home_t:s0 ./
```

```
sudo chcon -u system_u -t svirt_sandbox_file_t ~/pv/4
sudo chcon -u system_u -t svirt_sandbox_file_t ~/pv/4/file
```

Task 5 - hint

```
# oc describe pod task5
...
Unable to mount volumes for pod … timeout expired … list of
unattached/unmounted volumes=[v1]
# oc get pod -o yaml
...
volumes:
  persistentVolumeClaim:
# oc get pv pv-${user}
NAME      CAPACITY   ...
px-ux     1Mi        ...
# oc get pvc -o yaml
    resources:
      requests:
        storage: 1Gi
```

Task 5 - solution

```
# oc get pv pv-${user}
NAME      CAPACITY   ...
px-ux     1Mi        ...
# oc get pvc -o yaml
    resources:
      requests:
        storage: 1Gi
```

```
# oc get pvc -o yaml > pvc.yaml
# vim pvc.yaml
    resources:
      requests:
        storage: 1Mi
# oc replace --force -f pvc.yaml
```