Table of Contents

* [Create and Assign Policies and Permissions](https://www.opentlc.com/labs/ocp_advanced_deployment/04_1_Authentication_And_Permissions_Solution_Lab.html#_create_and_assign_policies_and_permissions)
  + [Create Projects](https://www.opentlc.com/labs/ocp_advanced_deployment/04_1_Authentication_And_Permissions_Solution_Lab.html#_create_projects)
  + [Assign Administrative Privileges to Groups](https://www.opentlc.com/labs/ocp_advanced_deployment/04_1_Authentication_And_Permissions_Solution_Lab.html#_assign_administrative_privileges_to_groups)
  + [Configure Platform Administrator Access](https://www.opentlc.com/labs/ocp_advanced_deployment/04_1_Authentication_And_Permissions_Solution_Lab.html#_configure_platform_administrator_access)
* [Explore Docker Registry Security and Image Pull Policies](https://www.opentlc.com/labs/ocp_advanced_deployment/04_1_Authentication_And_Permissions_Solution_Lab.html#_explore_docker_registry_security_and_image_pull_policies)
  + [Deploy to Development Using S2I](https://www.opentlc.com/labs/ocp_advanced_deployment/04_1_Authentication_And_Permissions_Solution_Lab.html#_deploy_to_development_using_s2i)
  + [Deploy Development Image to Test Environment](https://www.opentlc.com/labs/ocp_advanced_deployment/04_1_Authentication_And_Permissions_Solution_Lab.html#_deploy_development_image_to_test_environment)
* [Allow Production Administrators to Run Unsafe Containers](https://www.opentlc.com/labs/ocp_advanced_deployment/04_1_Authentication_And_Permissions_Solution_Lab.html#_allow_production_administrators_to_run_unsafe_containers)
  + [Add Production Service Accounts](https://www.opentlc.com/labs/ocp_advanced_deployment/04_1_Authentication_And_Permissions_Solution_Lab.html#_add_production_service_accounts)
  + [Create Root Docker Image Using S2I](https://www.opentlc.com/labs/ocp_advanced_deployment/04_1_Authentication_And_Permissions_Solution_Lab.html#_create_root_docker_image_using_s2i)
  + [Verify Deployed Root Docker Image](https://www.opentlc.com/labs/ocp_advanced_deployment/04_1_Authentication_And_Permissions_Solution_Lab.html#_verify_deployed_root_docker_image)
* [Disable Project Self-Provisioning](https://www.opentlc.com/labs/ocp_advanced_deployment/04_1_Authentication_And_Permissions_Solution_Lab.html#_disable_project_self_provisioning)
  + [Configure Project Request Message](https://www.opentlc.com/labs/ocp_advanced_deployment/04_1_Authentication_And_Permissions_Solution_Lab.html#_configure_project_request_message)
  + [Remove Permissions](https://www.opentlc.com/labs/ocp_advanced_deployment/04_1_Authentication_And_Permissions_Solution_Lab.html#_remove_permissions)
  + [Allow Production Administrators to Create Projects](https://www.opentlc.com/labs/ocp_advanced_deployment/04_1_Authentication_And_Permissions_Solution_Lab.html#_allow_production_administrators_to_create_projects)
* [Clean Up Environment](https://www.opentlc.com/labs/ocp_advanced_deployment/04_1_Authentication_And_Permissions_Solution_Lab.html#_clean_up_environment)

Create and Assign Policies and Permissions

In this section, you create projects and policies and assign them to different groups. You also explore how to grant cluster administration roles.

You assign roles to groups in projects or the cluster according to these criteria:

| **Group** | **Role** | **Projects/Cluster** |
| --- | --- | --- |
| **portalapp** | **admin** | **portalapp-dev** and **portalapp-test** projects |
| **paymentapp** | **admin** | **paymentapp-dev** and **paymentapp-test** projects |
| **ocp-production** | **admin** | **portalapp-prod** and **paymentapp-prod** projects |
| **ocp-platform** | **cluster-admin** | OpenShift cluster |

Create Projects

1. Create the projects for the **portalapp** and **paymentapp** applications:
2. export APPNAME=portalapp
3. export APPTEXT="Portal App"
4. oc new-project ${APPNAME}-dev --display-name="${APPTEXT} Development"
5. oc new-project ${APPNAME}-test --display-name="${APPTEXT} Testing"
6. oc new-project ${APPNAME}-prod --display-name="${APPTEXT} Production"
7. export APPNAME=paymentapp
8. export APPTEXT="Payment App"
9. oc new-project ${APPNAME}-dev --display-name="${APPTEXT} Development"
10. oc new-project ${APPNAME}-test --display-name="${APPTEXT} Testing"

oc new-project ${APPNAME}-prod --display-name="${APPTEXT} Production"

Assign Administrative Privileges to Groups

RHOCP is hooked in to IPA, but only one user has logged in, so there is only one account.

1. Retrieve the list of users to verify that there is only one account:

oc get users

1. Assign administrative privileges to the developer groups for their respective projects—in this case, using the default **admin** role that comes with RHOCP.
2. oc adm policy add-role-to-group admin portalapp -n portalapp-dev
3. oc adm policy add-role-to-group admin portalapp -n portalapp-test
4. oc adm policy add-role-to-group admin paymentapp -n paymentapp-dev

oc adm policy add-role-to-group admin paymentapp -n paymentapp-test

1. Examine the policy bindings for either of these projects to verify success:

oc describe rolebinding.rbac -n paymentapp-dev

**Sample Output**

Name: admin

Labels: <none>

Annotations: <none>

Role:

Kind: ClusterRole

Name: admin

Subjects:

Kind Name Namespace

---- ---- ---------

User system:admin

Name: admin-0

Labels: <none>

Annotations: <none>

Role:

Kind: ClusterRole

Name: admin

Subjects:

Kind Name Namespace

---- ---- ---------

Group paymentapp

[...]

1. Verify that the groups are given the **admin** role in their respective projects.
2. Assign administrators to production projects.
3. oc adm policy add-role-to-group admin ocp-production -n portalapp-prod

oc adm policy add-role-to-group admin ocp-production -n paymentapp-prod

Configure Platform Administrator Access

1. Add the **cluster-admin** role to the **ocp-platform** group.

oc adm policy add-cluster-role-to-group cluster-admin ocp-platform

1. Log out of the web console and log in again as one of the platform administrators—for example, **david**.
   * Expect to see all of the projects, including the OpenShift system projects such as the **default** project.

Explore Docker Registry Security and Image Pull Policies

In this section, you explore how to set policies allowing one project to view and pull images from another project. You allow service accounts from the **paymentapp-prod** and **paymentapp-test** to pull images created in the **paymentapp-dev** project.

1. Grant image pull rights on the **paymentapp-dev** project to the service accounts in the **paymentapp-prod** and **paymentapp-test**projects.
2. oc login -u system:admin
3. oc policy add-role-to-group system:image-puller system:serviceaccounts:paymentapp-prod -n paymentapp-dev

oc policy add-role-to-group system:image-puller system:serviceaccounts:paymentapp-test -n paymentapp-dev

1. Assign the **registry-viewer** role to the **ocp-production** group so that the production administrators can see the image streams.
2. oc policy add-role-to-group registry-viewer ocp-production -n paymentapp-dev

oc policy add-role-to-group registry-viewer ocp-production -n paymentapp-test

Deploy to Development Using S2I

In this section, you deploy the **https://github.com/openshift/sinatra-example** Ruby application in **paymentapp-dev** and tag the resulting image as **sinatra:test**.

1. As the **marina** user, use **oc new-app** to build the **sinatra** example:
2. oc login -u marina --password 'r3dh4t1!'
3. oc project paymentapp-dev

oc new-app ruby~https://github.com/openshift/sinatra-example --name=sinatra -n paymentapp-dev

1. Wait for the build to complete:

oc logs -f build/sinatra-1 -n paymentapp-dev

**Sample Output**

Cloning "https://github.com/openshift/sinatra-example" ...

Commit: ff65a82271fffc60d4129bccde9c42ded49a199d (Merge pull request #11 from corey112358/patch-1)

Author: Ben Parees <bparees@users.noreply.github.com>

Date: Wed Jul 22 00:20:36 2015 -0400

[...]

Pushing image docker-registry.default.svc:5000/paymentapp-dev/sinatra:latest ...

Pushed 0/6 layers, 8% complete

Pushed 1/6 layers, 38% complete

Pushed 2/6 layers, 47% complete

Pushed 3/6 layers, 57% complete

Pushed 4/6 layers, 72% complete

Pushed 5/6 layers, 99% complete

Pushed 6/6 layers, 100% complete

Push successful

* + The image is placed in the **paymentapp-dev** path in the registry.

1. Once the application has finished building, examine the tags:

oc describe imagestream sinatra -n paymentapp-dev

**Sample Output**

Name: sinatra

Namespace: paymentapp-dev

Created: About a minute ago

Labels: app=sinatra

Annotations: openshift.io/generated-by=OpenShiftNewApp

Docker Pull Spec: docker-registry.default.svc:5000/paymentapp-dev/sinatra

Image Lookup: local=false

Unique Images: 1

Tags: 1

latest

no spec tag

\* docker-registry.default.svc:5000/paymentapp-dev/sinatra@sha256:101f923bfd9e2239a6b6347f8088121ea11d8325c9e92ba708808450ba5ff0a3

28 seconds ago

1. Tag the **latest** image as **test**.

oc tag sinatra:latest sinatra:test

**Sample Output**

Tag sinatra:test set to sinatra@sha256:101f923bfd9e2239a6b6347f8088121ea11d8325c9e92ba708808450ba5ff0a3.

* + In the next section, you use the **test** tag to deploy to the test project.

Deploy Development Image to Test Environment

1. Log in as **marina** and switch to the **portalapp-test** project:
2. oc login -u marina --password 'r3dh4t1!'

oc project paymentapp-test

1. Use **oc new-app** to deploy the **paymentapp-dev** project’s **sinatra** image with the **test** tag:

oc new-app paymentapp-dev/sinatra:test -n paymentapp-test

|  |  |
| --- | --- |
|  | Do not forget the **test** tag—without it, you get **latest**. |

1. Check that the deployment is successful by watching the pods:

oc get pods

**Sample Output**

NAME READY STATUS RESTARTS AGE

sinatra-1-hfqp5 1/1 Running 0 42s

1. If the deployment is successful, imagine that you tested the application, then tag the image as **sinatra:prod** (still as the **marina**user) and deploy it to the **paymentapp-prod** project as the **david** user.
   1. Tag the image:

oc tag sinatra:test sinatra:prod -n paymentapp-dev

**Sample Output**

Tag sinatra:prod set to sinatra@sha256:101f923bfd9e2239a6b6347f8088121ea11d8325c9e92ba708808450ba5ff0a3.

* 1. Deploy to production:
  2. oc login -u david -p 'r3dh4t1!'
  3. oc project paymentapp-prod
  4. oc new-app paymentapp-dev/sinatra:prod -n paymentapp-prod
  5. sleep 10

oc get pod

Allow Production Administrators to Run Unsafe Containers

In this section, you allow one of the projects to create and deploy an S2I-built image with **root** permissions—in other words, to run privileged containers.

Users generally do not create pods directly. They create a deployment configuration or a replication controller to launch the pods. Therefore, it is the service account in the project that needs **root** permissions.

RHOCP comes with a number of security context constraints (SCCs), and the **anyuid** SCC does what you want—it allows you to run containers as any UID, specifically **root**. Since only production administrators have access to the production projects, you can simply allow the service accounts for the production projects to run containers as any UID.

Add Production Service Accounts

In this step, you use SCCs to allow service accounts in the **paymentapp-prod** project to run images/containers running with the **root**user.

1. Modify the SCCs to allow **anyuid** privilege for the service account in the **paymentapp-prod** project.
2. oc login -u system:admin

oc adm policy add-scc-to-group anyuid system:serviceaccounts:paymentapp-prod

Create Root Docker Image Using S2I

1. Log in as **prod1** and switch to the **paymentapp-prod** project:
2. oc login -u prod1 -p 'r3dh4t1!'

oc project paymentapp-prod

1. Build a **root** Docker image using S2I:

oc new-app --strategy=docker https://github.com/thoraxe/centos-root-docker-example

1. Wait for the build to complete:

oc logs -f build/centos-root-docker-example-1 -n paymentapp-prod

Verify Deployed Root Docker Image

1. Once the build completes, verify that a deployed pod called **centos-root-docker-example** results:

oc get pod

**Sample Output**

centos-root-docker-example-1-9x2lr 1/1 Running 0 12s

1. Use **oc rsh** (or the web UI) to connect to the pod and type **whoami** to verify that you are **root**:
2. oc rsh centos-root-docker-example-1-9x2lr
3. sh-4.2*# whoami*
4. root

sh-4.2*# exit*

Disable Project Self-Provisioning

In this exercise, you remove the user’s default permission to create their own projects and allow only production administrators to create projects. To complete this exercise, you do the following:

* Make sure that users cannot create projects.
* Allow users from the **ocp-production** group to create their own projects.
* Configure the message for users who attempt to create projects: "Please create project using the portal or contact Karla at [karla@example.com](mailto:karla@example.com)".

Configure Project Request Message

In this section, you use the installer to configure the project request message. This requires rerunning the installer and takes about 30 minutes.

1. Edit **/etc/ansible/hosts** on your **bastion** host and add the project request message.
   * Make sure that the **[OSEv3:vars]** section contains the following:
   * # Project Configuration

osm\_project\_request\_message='Please create project using the portal http://portal.$GUID.internal/provision or contact Karla at karla@example.com'

1. On the **bastion** server, run the installer again:

ansible-playbook -f 20 /usr/share/ansible/openshift-ansible/playbooks/deploy\_cluster.yml

* + This takes about 30 minutes.

|  |  |
| --- | --- |
|  | Alternatively, you can edit the **/etc/origin/master/master-config.yaml** file on all masters and just restart the master API service, which is much quicker.  To do this, you need the following in the **master-config.yaml** file:  projectConfig:  projectRequestMessage: Please create project using the portal http://portal.$GUID.internal/provision  or contact Karla at karla@example.com |

Remove Permissions

1. Log in as **system:admin** and set the project to **default**:
2. oc login -u system:admin

oc project default

1. Disable self-provisioning for the **system:authenticated** group by editing the cluster roles.
2. oc adm policy remove-cluster-role-from-group self-provisioner system:authenticated

oc adm policy remove-cluster-role-from-group self-provisioner system:authenticated:oauth

1. Examine the cluster roles:

oc describe clusterrolebinding.rbac -n default

1. In the lengthy output, look for this section:

**Sample Output**

[...]

Name: self-provisioner

Labels: <none>

Annotations: <none>

Role:

Kind: ClusterRole

Name: self-provisioner

Subjects:

Kind Name Namespace

---- ---- ---------

ServiceAccount management-admin management-infra

[...]

1. Log in as an authenticated user—not as **system:admin**:

oc login -u payment1 -p r3dh4t1!

1. Verify that the updated project request message appears when trying to create a project:

oc new-project thiswillnotwork

**Sample Output**

Error from server (Forbidden): Please create project using the portal http://portal.$GUID.internal/provision or contact Karla at karla@example.com

Allow Production Administrators to Create Projects

In this section, you configure the platform administrator group you previously created so that its members can create projects for everyone.

1. Log in as **system:admin** and select the **default** project:
2. oc login -u system:admin

oc project default

1. Use **oc adm policy** again, but this time add the cluster role of **self-provisioner** to the **ocp-production** group.

oc adm policy add-cluster-role-to-group self-provisioner ocp-production

1. Log in as one of the **prod1** or **prod2** production administrators:

oc login -u prod1 -p r3dh4t1!

1. Create a new project and verify that it works:

oc new-project thiswillwork

**Sample Output**

Now using project "thiswillwork" on server "https://loadbalancer1.c3fc.internal:443".

You can add applications to this project with the 'new-app' command. For example, try:

oc new-app centos/ruby-22-centos7~https://github.com/openshift/ruby-ex.git

to build a new example application in Ruby.

Clean Up Environment

1. Delete the projects you created in this lab:
2. oc login -u system:admin
3. oc delete project thiswillwork
4. oc delete project paymentapp-dev
5. oc delete project paymentapp-test
6. oc delete project paymentapp-prod
7. oc delete project portalapp-dev
8. oc delete project portalapp-test

oc delete project portalapp-prod

Build Version: c3147ce9f77191e30b447cc423f2f68a0c40fc03 : Last updated 2018-07-31 01:29:02 EDT