OpenShift4 Developer On-Boarding System Demo

架构介绍

多租户情况下， OpenShift 需要在集群上加入 developer 团队。从理论上讲，我们需要一个流程，该流程可以自动为团队提供所有必需的资源和授权以开始使用 OpenShift。本文会先进行概念介绍，然后进行实验展现。

此解决方案需要解决三个问题：

- 将公司组织映架构射到 OpenShift
- 将组织结构从目录管理系统（AD、LDAP）同步到 OpenShift
- 创建和配置 namespace

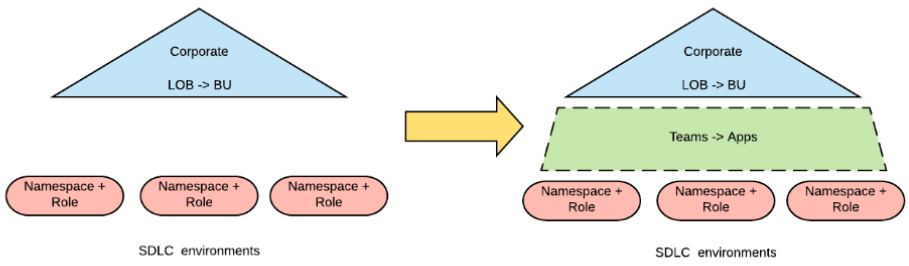
步骤 1：将公司组织映射到 OpenShift

首先，我们需要将公司组织结构映射到 OpenShift，以使与用户相关联的权限与他们当前在组织中拥有的实际角色一致。通常，我们不需要映射整个组织，而只需要映射必须使用 OpenShift 的部分。

通常，公司的组织存储在目录服务（例如 LDAP）。相对于 OpenShift 所需要的架构，我们在这些目录服务中的组织结构通常是分层的且粗粒度的。例如，我们可能具有以下层次结构级别：业务线（LOB）->业务单位（BU）。但这在 LDAP 中可能不存在。

在 OpenShift 端，通过 role、group、namespace（通常代表 SDLC：Software Development Life Cycle 的特定环境）之间的三向绑定来授予权限。Kubernetes 中的命名空间是一种扁平结构，没有层次概念。另外，由于它们代表 SDLC 环境，因此它们的粒度也相对较细。

因此，我们在公司的目录服务中可以找到的公司组织代表与 OpenShift 的需求之间通常存在差距。如下图所示：

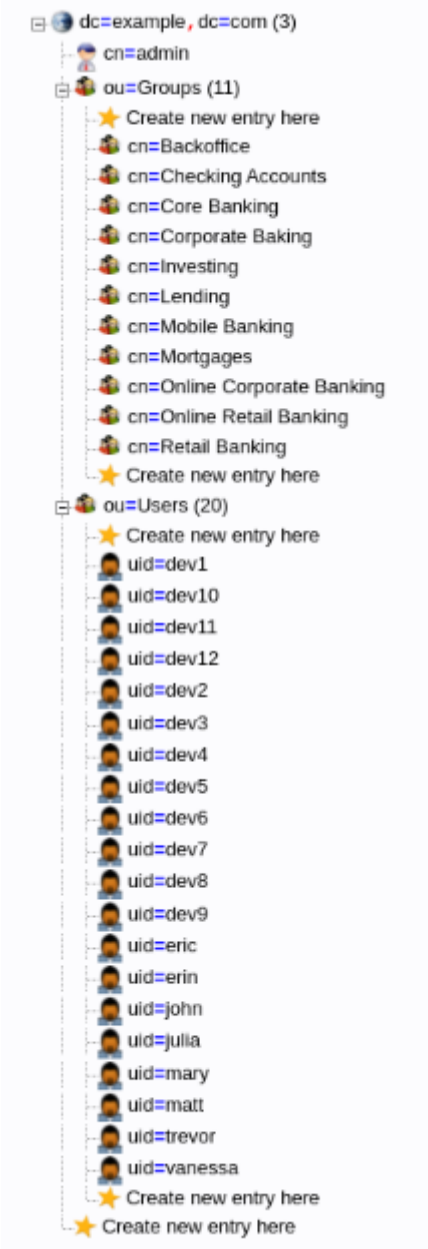


此处的关键是图右侧的绿色层，它代表了 Teams 和 Apps 层对公司 LDAP 组织层次结构的增强。也就是说，增加了一层类似租户的概念。

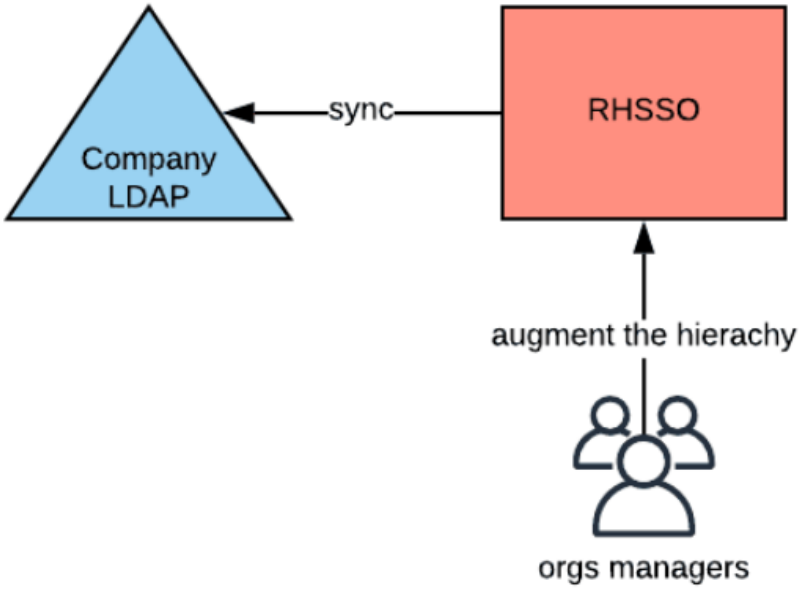
为了实现这些目标，必须使用工具将司层次结构以及达到 OpenShift 所需粒度所需的层次。

此外，这个工具还应该提供一定程度的自我服务，以便团队经理可以自行配置他们在组织中的部分。该组织管理工具可能像 git 储存库中的文件一样简单，该文件通过“拉取请求（PR）”的形式进行审核，一旦更新，就会触发某种下游自动化。

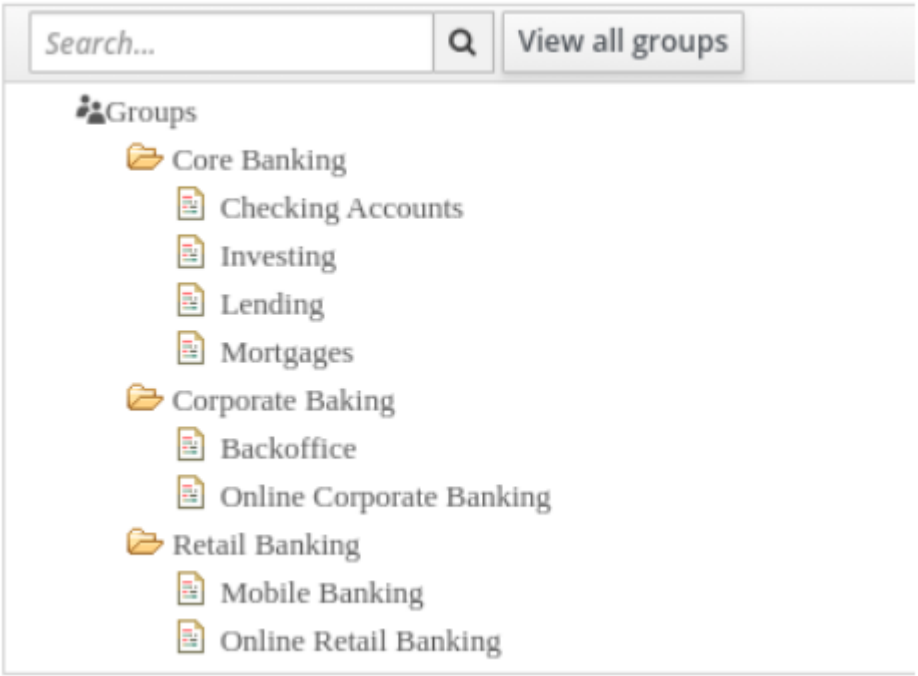
我们使用 RH SSO 作为组织管理工具（社区 keycloack）。我们假设我们有一个具有两个层次结构层的公司 LDAP：业务范围和业务部门。



然后，我们将署 RH SSO 并将其与 LDAP 服务器集成，如下图所示：



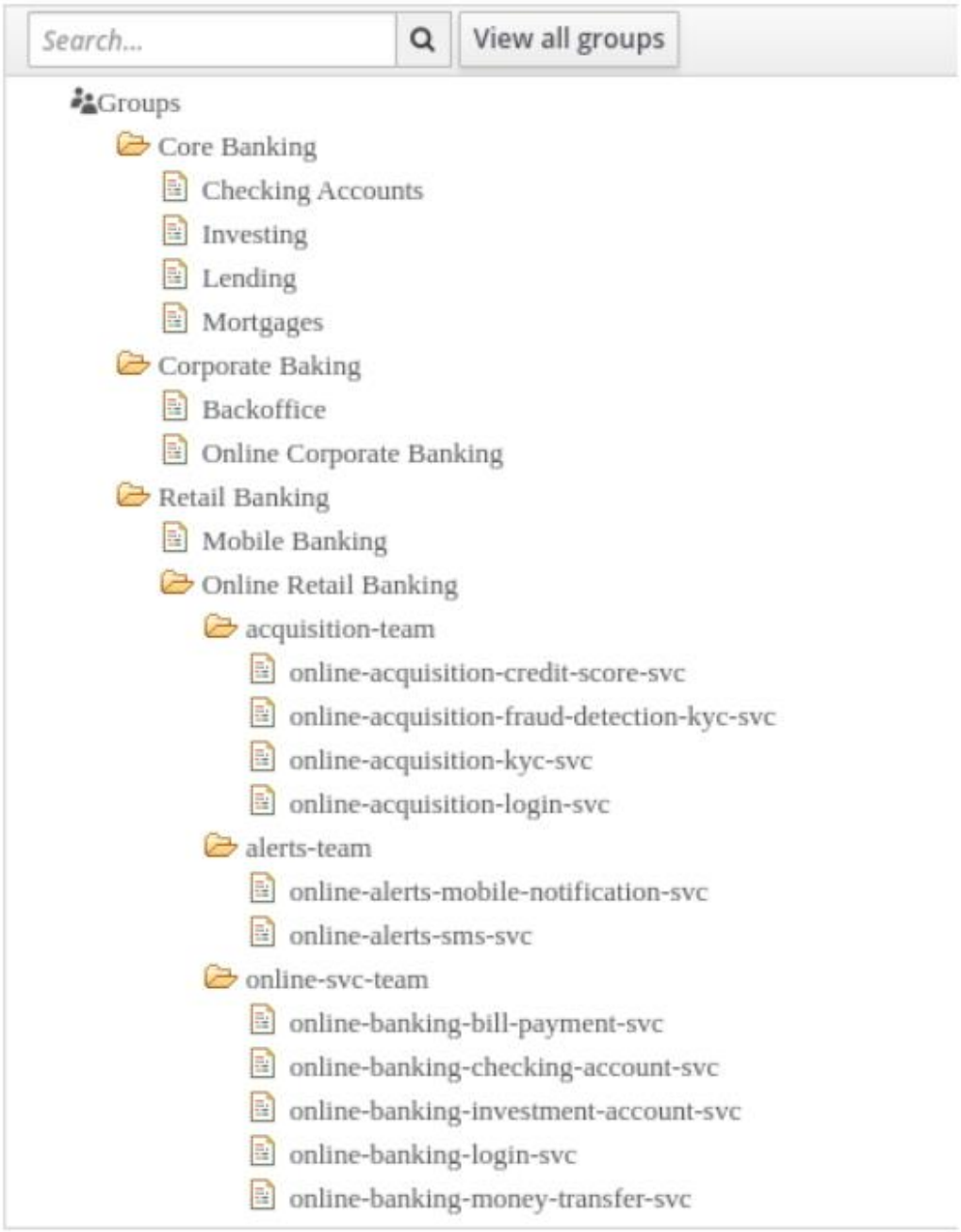
RH SSO 和 LDAP 之间的集成每五分钟同步一次。经过这段时间后，在 RH SSO 中看到导入的层次结构：



如上图所示，此时每个 BU 的 manager 想通过增加 group（team，application）的方式来拓展架构。然就将 employee 指定到各个 group。

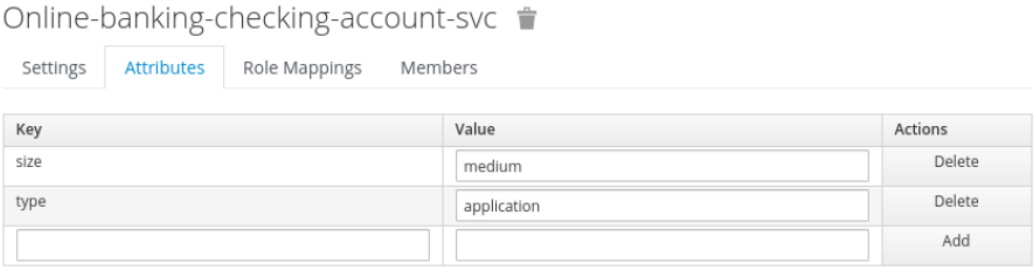
在具有细粒度权限的 RH SSO 中，可以创建一种配置，通过该配置，用户被授予 role 来管理组层次结构的一部分。完成此配置后，该用户将能够将子组添加到层次结构中并管理组成员身份。这样，扩展原始层次结构的职责不必由团队集中负责，而是可以移交给 BU 所有者。

在我们的演示中，我们使用脚本模拟了增强过程。执行脚本后，将显示以下内容：



如上图所示，我们扩展了 Online Retail Banking。

，我们已经根据命名约定对 group 进行了标准化。另外，我们还向创建的组添加了元数据（稍后将对此进行详细说明），如下所示：



因此，该特定组具有两部分元数据：type 和 size。 我们将使用 type 来确定应将哪些操作应用于该组。 在这种情况下，type 是 *application*，因此，我们将为其创建 SDLC 名称空间。我们使用 size 来确定要分配给创建的名称空间的配额。

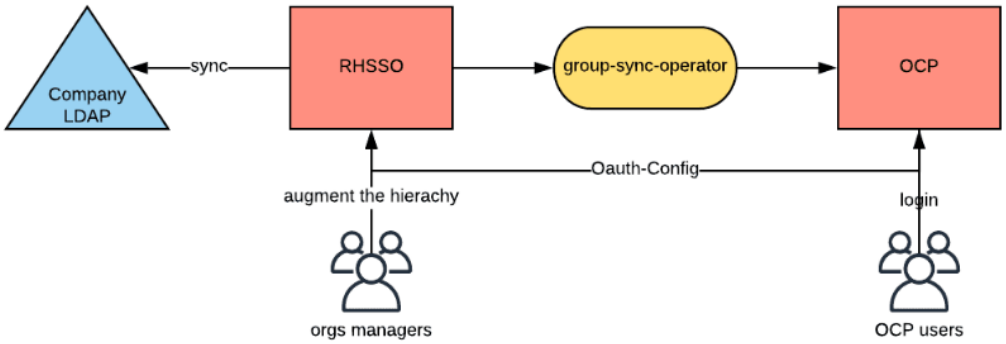
组同步

在 OpenShift 中：组和用户需要存在于 OpenShift 中，以使身份验证和授权按设计运行。User objects 是用户首次登录时创建的，因此其管理相对简单。相反，需要通过一些外部过程或自动化来创建组。在大多数情况下，这意味着需要使用同步机制从记录组系统中导入组。

在以前版本的 OCP 中，OpenShift 仅通过 oc 命令支持与 LDAP 同步组。但是，越来越多的客户最近开始将 OpenShift 与不同的 IDP 提供程序（主要是 OIDC 实现）进行集成。我们最好使用 group-sync-operator 来实现 group sync。

group-sync-operator 可以与多个 IDP 同步组和组成员身份，并且具有插件架构，因此将来可以添加对更多 IDP 的支持。而且，只要 IDP 允许，它就可以管理组元数据，例如层次结构信息和通用组标签。

在我们的演示中，我们将设置 OpenShift 与 RH-SSO 作为身份验证提供程序集成，并使用 group-sync-operator 同步来自 RH-SSO 的组：



完成这些步骤（OCP-RH SSO 集成，组同步部署和配置）之后，您应该能够看到已将组导入到 OpenShift 中。如果检查组，则应该能够看到与它们相关的元数据，最后，您应该能够使用 RH-SSO 登录到 OCP。登录时，此时，您将没有任何权限。

命名空间配置

至此，我们在 OpenShift 中有了所需的组和用户。接下来，我们需要创建具有正确权限和配置的名称空间。更具体地说，除了创建名称空间外，客户通常要求的一些常见配置类型是：

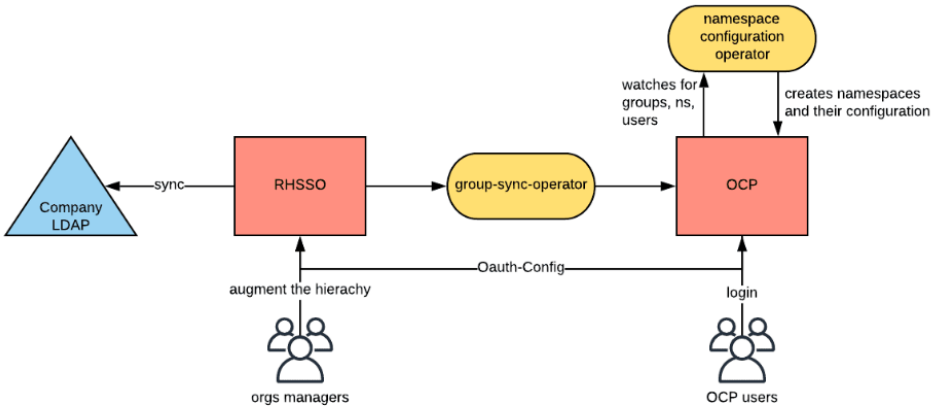
配置角色绑定
配置名称空间配额和多名称空间配额
配置网络策略
配置出口网络策略

namespace-configuration-operator 允许我们定义策略以基于用户、组和命名空间创建和强

制使用任意 Kubernetes 资源。

namespace-configuration-operator 使用标签和注释选择器选择要在哪些对象上应用定义的配置。

我们使用 namespace-configuration-operator 基于定义的组及其批注（组元数据）创建名称空间，此外，我们还将使用前面介绍的一些资源来配置创建的名称空间。。



完成以上操作后（名称空间配置操作员部署和策略定义）之后，我们会看到已将几个名称空间添加到集群中。我们还应该能够看到，如果使用 LDAP 中的用户登录，可以看到多个名称空间，但是只能在其中一些名称空间上操作：

NS ocp-ops-view	Active
NS online-acquisition-credit-score-svc-build	Active
NS online-acquisition-credit-score-svc-dev	Active
NS online-acquisition-credit-score-svc-prod	Active
NS online-acquisition-credit-score-svc-qa	Active
NS online-acquisition-fraud-detection-kyc-svc-build	Active
NS online-acquisition-fraud-detection-kyc-svc-dev	Active
NS online-acquisition-fraud-detection-kyc-svc-prod	Active
NS online-acquisition-fraud-detection-kyc-svc-qa	Active
NS online-acquisition-kyc-svc-build	Active
NS online-acquisition-kyc-svc-dev	Active
NS online-acquisition-kyc-svc-prod	Active

一旦完成此设置，如果在 LDAP 或 RH SSO 中进行了修改，它们将迅速反映在 OCP 上。除了要求被放入正确的组中之外，用户不必请求任何其他配置，并且应该能够立即使用 OpenShift。

实验验证

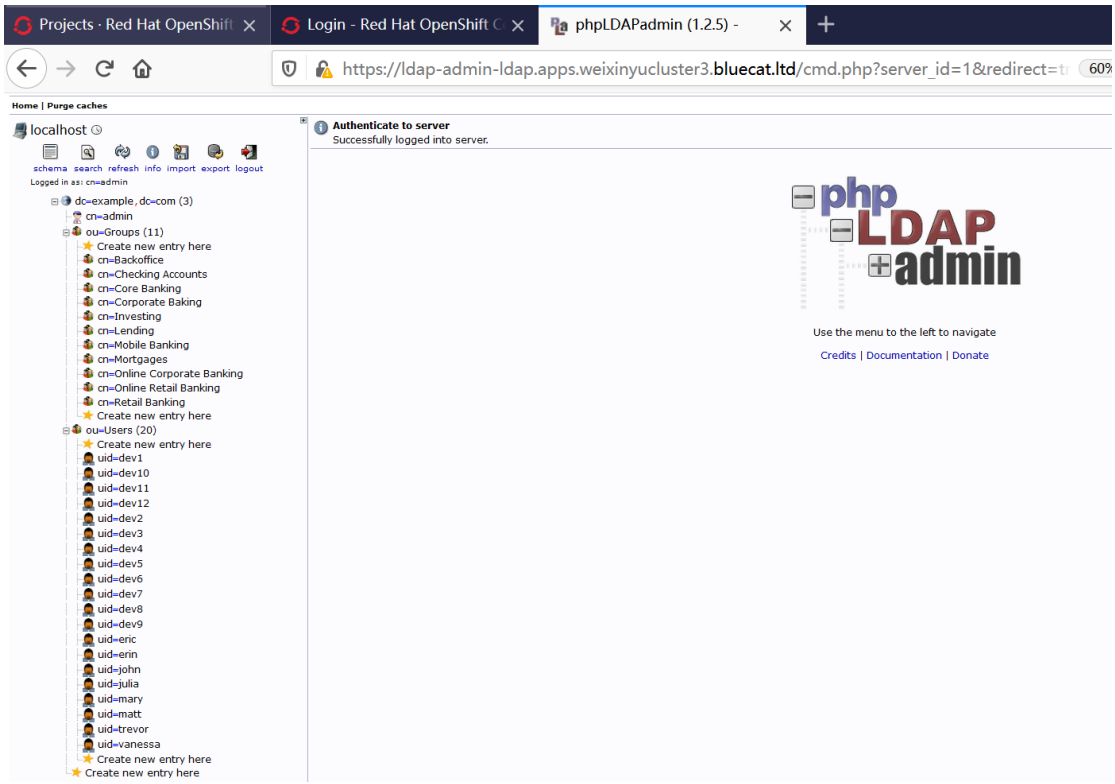
git clone https://github.com/raffaelespazzoli/orgs-management-ocp.git
首先在 OCP 中创建一个 ldap

```
oc new-project ldap

oc adm policy add-scc-to-user anyuid -z default -n ldap

oc apply -f ./ldap -n ldap
```

完成此步骤后，您应该能够连接到 ldap admin UI:

```
[root@lb.weixinyucluster3 ~/orgs-management-ocp]# echo https://$(oc get route ldap-admin -n ldap -o jsonpath='{.spec.host}')

https://ldap-admin-ldap.apps.weixinyucluster3.bluecat.ltd
```

使用 cn=admin,dc=example,dc=com/admin 认证登录，如下图所示:



这代表了我们在企业 LDAP 中可能发现的情况，并且是我们演示的起点。

所有具有人名的用户都将 admin 分配为密码。 以 dev 开头的所有用户都将 dev 作为密码。

安装 RH-SSO

接下来，安装 RH-SSO（安装之前需要保证 OCP 中有一个可用的 pv，数据库需要）

```
oc new-project keycloak-operator

oc apply -f ./keycloak/operator.yaml -n keycloak-operator

oc apply -f ./keycloak/keycloak.yaml -n keycloak-operator

oc create route reencrypt keycloak --port 8443 --service keycloak -n keycloak-operator
```

```
[root@lb.weixinyucluster3 ~/orgs-management-ocp]# oc get pods
NAME                                  READY   STATUS    RESTARTS   AGE
keycloak-0                            1/1     Running   0          2m2s
keycloak-operator-5658d655bf-knhhf    1/1     Running   0          2m38s
keycloak-postgresql-575599fbf6-mkg25  1/1     Running   0          112s
```

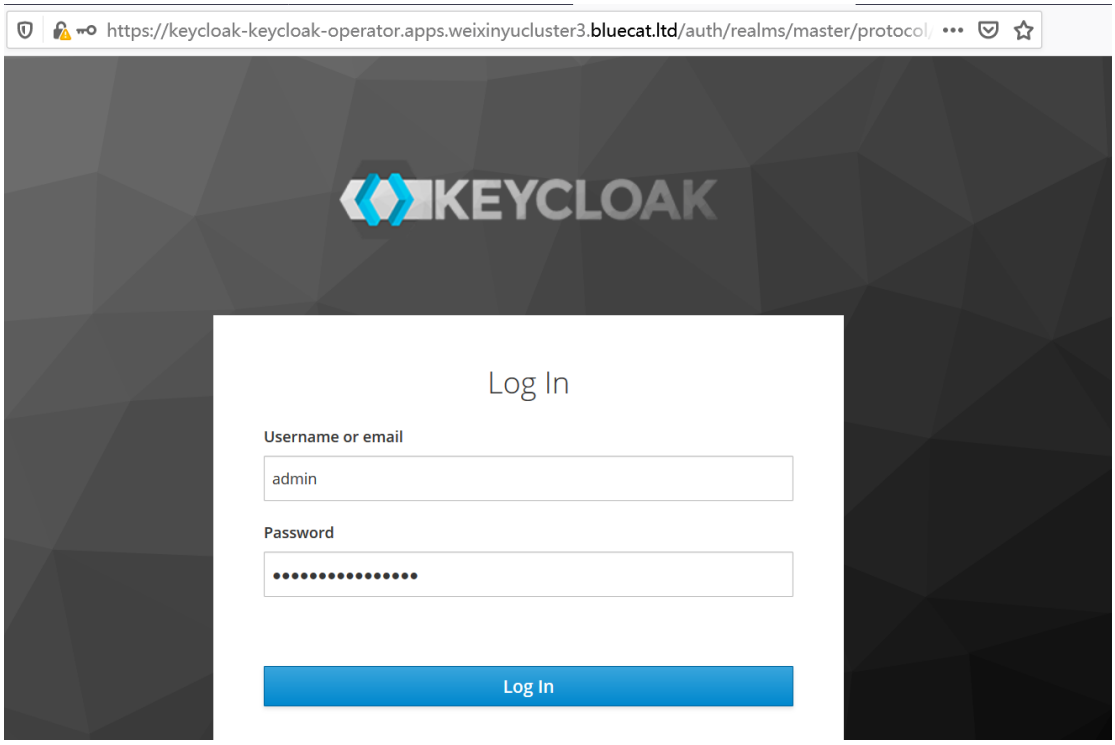
将 RH-SSO 与 LDAP 集成：

```
export admin_password=$(oc get secret credential-ocp-keycloak -n keycloak-operator -o jsonpath='{.data.ADMIN_PASSWORD}' | base64 -d)

oc exec -n keycloak-operator keycloak-0 -- /opt/jboss/keycloak/bin/kcadm.sh config credentials --server http://localhost:8080/auth --realm master --user admin --password ${admin_password} --config /tmp/kcadm.config

export ldap_integration_id=$(cat ./keycloak/ldap-federation.json | envsubst | oc exec -i -n keycloak-operator keycloak-0 -- /opt/jboss/keycloak/bin/kcadm.sh create components --config /tmp/kcadm.config -r ocp -f - -i)

echo created ldap integration $ldap_integration_id

cat ./keycloak/role-mapper.json | envsubst | oc exec -i -n keycloak-operator keycloak-0 -- /opt/jboss/keycloak/bin/kcadm.sh create components --config /tmp/kcadm.config -r ocp -f -
```

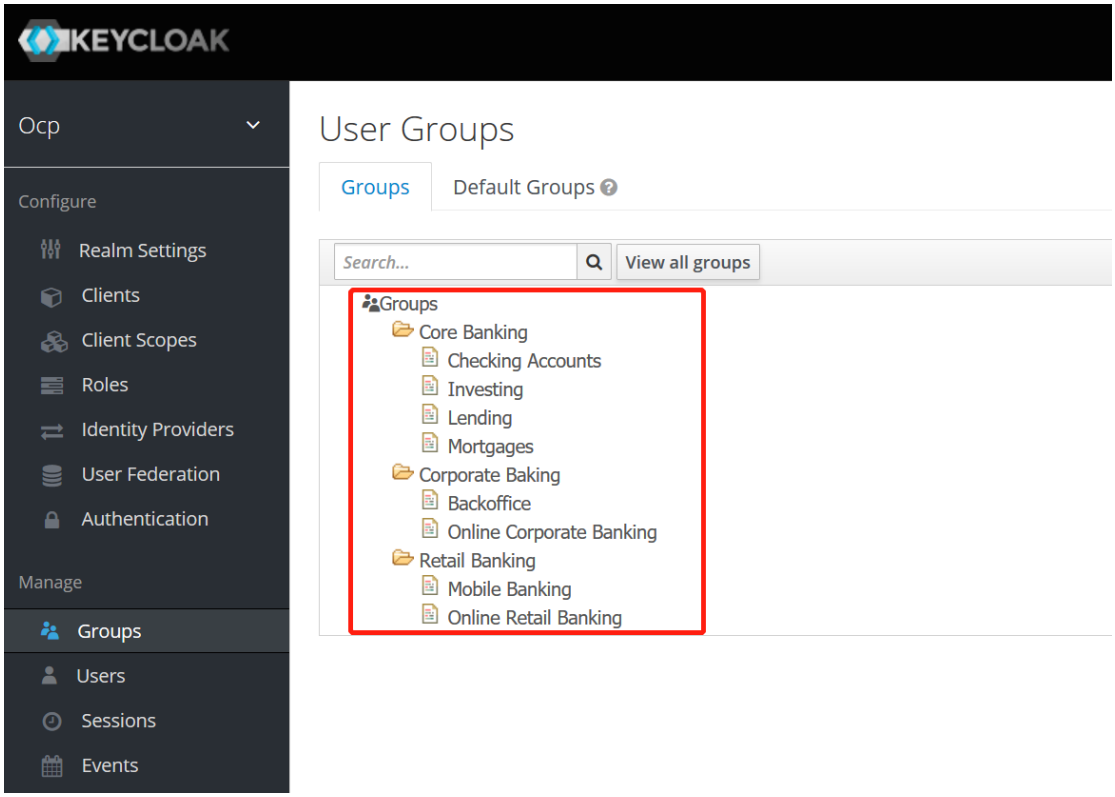
查看 RH-SSO 的登录方式：

```
echo https://$(oc get route keycloak -n keycloak-operator -o jsonpath='{.spec.host}')

echo admin/${admin_password}
```

```
[root@lb.weixinyucluster3 ~/orgs-management-ocp]# echo https://$(oc get route keycloak -n keycloak-operator -o jsonpath='{.spec.host}')
https://keycloak-keycloak-operator.apps.weixinyucluster3.bluecat.ltd
[root@lb.weixinyucluster3 ~/orgs-management-ocp]# echo admin/${admin_password}
admin/4k-tnNdLXYc1cQ==
```



从 LDAP 到 RH-SSO 的同步时间需要 5 分钟，同步后在 RH-SSO 中可以看到 LDAP 中同步过来的组：



RH-SSO 组扩充

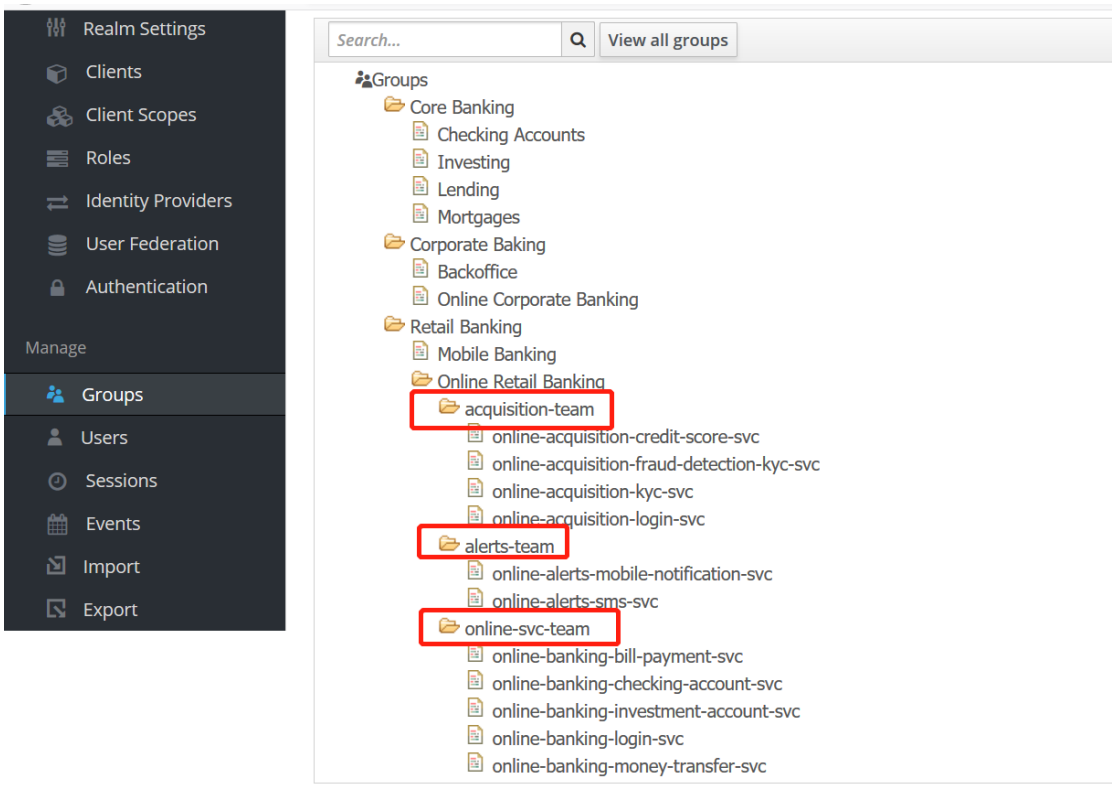
在本节中，我们模拟连接到 RH-SSO 的组织所有者，并通过添加开发团队和应用程序层来组织他们在组织层次结构中的部分。

./keycloak/group-config.sh

```
[root@lb.weixinyucluster3 ~/orgs-management-ocp]# ./keycloak/group-config.sh
Logging into http://localhost:8080/auth as user admin of realm master
```

```
[root@lb.weixinyucluster3 ~/orgs-management-ocp]# cat ./keycloak/group-config.sh
```

命令执行成功后，查看 RH-SSO，增加了 dev team 和 application layers



OCP - RH-SSO 的集成

首先安装 yq: https://kislyuk.github.io/yq/

yum -y install python-pip

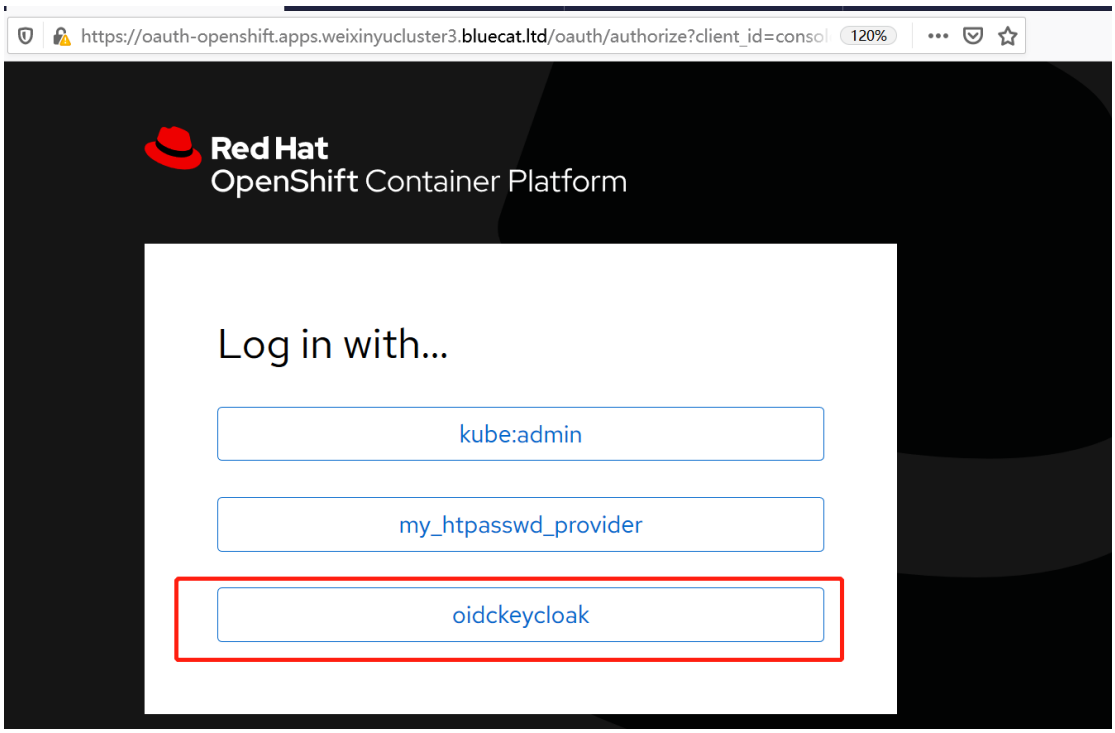
pip install yq

```
export keycloak_route=$(oc get route keycloak -n keycloak-operator -o jsonpath='{.spec.host}')

export auth_callback=$(oc get route oauth-openshift -n openshift-authentication -o jsonpath='{.spec.host}')/oauth2callback

cat ./ocp-auth/keycloak-client.yaml | envsubst | oc apply -f - -n keycloak-operator
```

```
oc apply -f ./ocp-auth/secret.yaml

oc get secrets -n openshift-ingress-operator router-ca -o jsonpath='{.data.tls\.crt}' | base64 -d > /tmp/ca.crt

oc -n openshift-config create configmap ocp-ca-bundle --from-file=/tmp/ca.crt

export oauth_patch=$(cat ./ocp-auth/oauth.yaml | envsubst | yq .)

oc patch OAuth.config.openshift.io cluster -p '[{"op": "add", "path": "/spec/identityProviders/-", "value": '"${oauth_patch}" }]'

--type json
```

```
[root@lb.weixinyucluster3 ~/orgs-management-ocp]# oc describe oauth.config.openshift.io/cluster
Name:         cluster
Namespace:
Labels:       <none>
Annotations:  kubectl.kubernetes.io/last-applied-configuration:
                {"apiVersion":"config.openshift.io/v1","kind":"OAuth","metadata":{"annotations":{},"name":"cluster"},"spec
":{"identityProviders":[{"htpass...
              release.openshift.io/create-only: true
API Version:  config.openshift.io/v1
Kind:         OAuth
Metadata:
  Creation Timestamp:  2020-07-11T04:16:49Z
  Generation:          3
  Resource Version:    7641821
  Self Link:           /apis/config.openshift.io/v1/oauths/cluster
  UID:                 bc00cb8b-9f00-47f9-98ee-5312070dfdee
Spec:
  Identity Providers:
    Htpasswd:
      File Data:
        Name:        htpass-secret
    Mapping Method:  claim
    Name:            my_htpasswd_provider
    Type:            HTPasswd
    Mapping Method:  claim
    Name:            oidckeycloak
    Open ID:
      Ca:
        Name:  ocp-ca-bundle
      Claims:
        Email:
          email
        Name:
          name
        Preferred Username:
          preferred_username
      Client ID:  ocp-client
      Client Secret:
        Name:  ocp-secret
      Issuer:  https://keycloak-keycloak-operator.apps.weixinyucluster3.bluecat.ltd/auth/realms/ocp
    Type:      OpenID
Events:        <none>
```

此时再登录 OCP，就会出现 RH-SSO 的认证入口：



OCP - RH-SSO Group Sync

部署 group sync operator

oc new-project group-sync-operator

oc apply -f ./group-sync/operator.yaml -n group-sync-operator

```
[root@lb.weixinyucluster3 ~/orgs-management-ocp]# oc get pods
NAME                                   READY   STATUS    RESTARTS   AGE
group-sync-operator-65dd87c549-bc6d4   1/1     Running   0          22s
[root@lb.weixinyucluster3 ~/orgs-management-ocp]#
```

部署 group sync logic

export keycloak_route=$(oc get route keycloak -n keycloak-operator -o jsonpath='{.spec.host}')

export admin_password=$(oc get secret credential-ocp-keycloak -n keycloak-operator -o jsonpath='{.data.ADMIN_PASSWORD}' | base64 -d)

oc create secret generic keycloak-group-sync --from-literal=username=admin --from-literal=password=${admin_password} -n keycloak-operator

oc create secret generic keycloak-group-sync --from-literal=username=admin --from-literal=password=${admin_password} -n group-sync-operator

[root@lb.weixinyucluster3 ~/orgs-management-ocp]# cat 2.yaml

apiVersion: redhatcop.redhat.io/v1alpha1

kind: GroupSync

metadata:

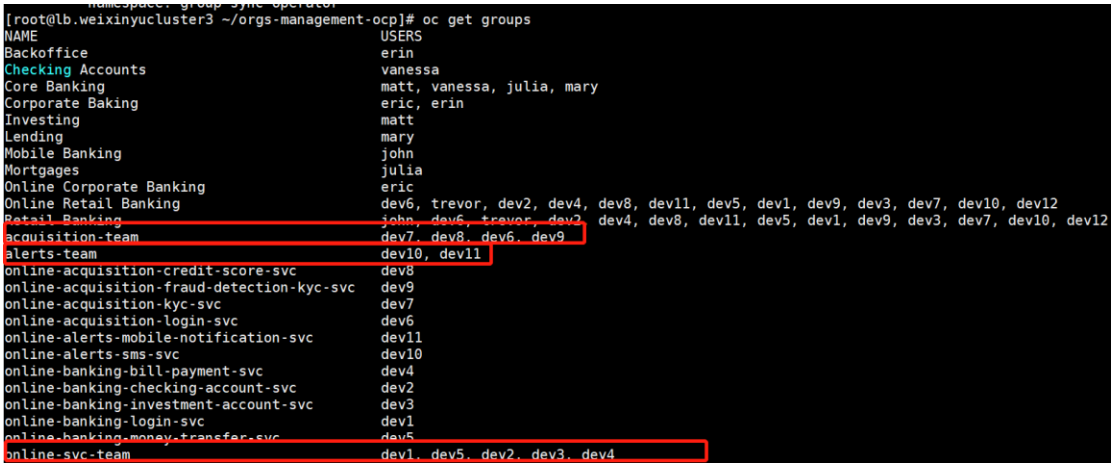
  name: keycloak-groupsync

```
spec:
  providers:
    - name: keycloak
      syncPeriodMinutes: "0 5 * * *"
      keycloak:
        insecure: true
        realm: ocp
        url: https://keycloak-keycloak-operator.apps.weixinyucluster3.bluecat.ltd
        credentialsSecret:
          name: keycloak-group-sync
          namespace: group-sync-operator
```

```
oc apply -f 2.yaml
```

同步后，我们可以看到：

oc get groups



从上图我们可以看到我们增加的三个“租户”。如 acquisition-team、alerts-team、online-svc-team。他们都属于 Online Retail Banking 这个组织：

部署 Deploy namespace-configuration-operator

```
oc new-project namespace-configuration-operator
```

```
oc apply -f ./namespace-configuration/operator.yaml -n namespace-configuration-operator
```

```
[root@lb.weixinyucluster3 ~/orgs-management-ocp]# oc get pods
NAME                                                READY   STATUS    RESTARTS   AGE
namespace-configuration-operator-54595ccbdf-2cwpg   1/1     Running   0          13s
```

部署 namespace configuration

```
oc apply -f ./namespace-configuration/admin-no-build-role.yaml
```

```
oc apply -f ./namespace-configuration/app-namespace-groupconfig.yaml
```

```
oc apply -f ./namespace-configuration/multiproject-quota-groupconfig.yaml
```

```
oc apply -f ./namespace-configuration/role-binding-groupconfig.yaml
```

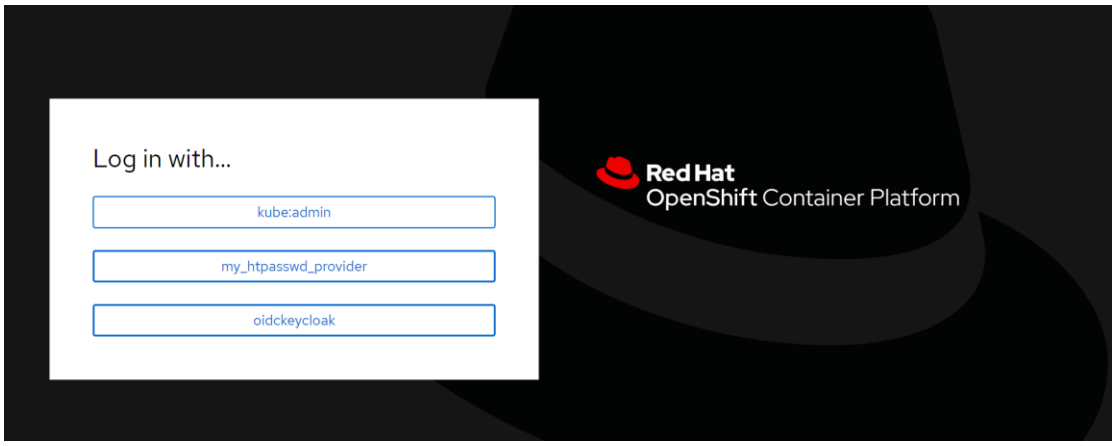
```
oc apply -f ./namespace-configuration/egress-networkpolicy-namespaceconfig.yaml
oc apply -f ./namespace-configuration/networkpolicy-namespaceconfig.yaml
oc apply -f ./namespace-configuration/quota-namespaceconfig.yaml
```

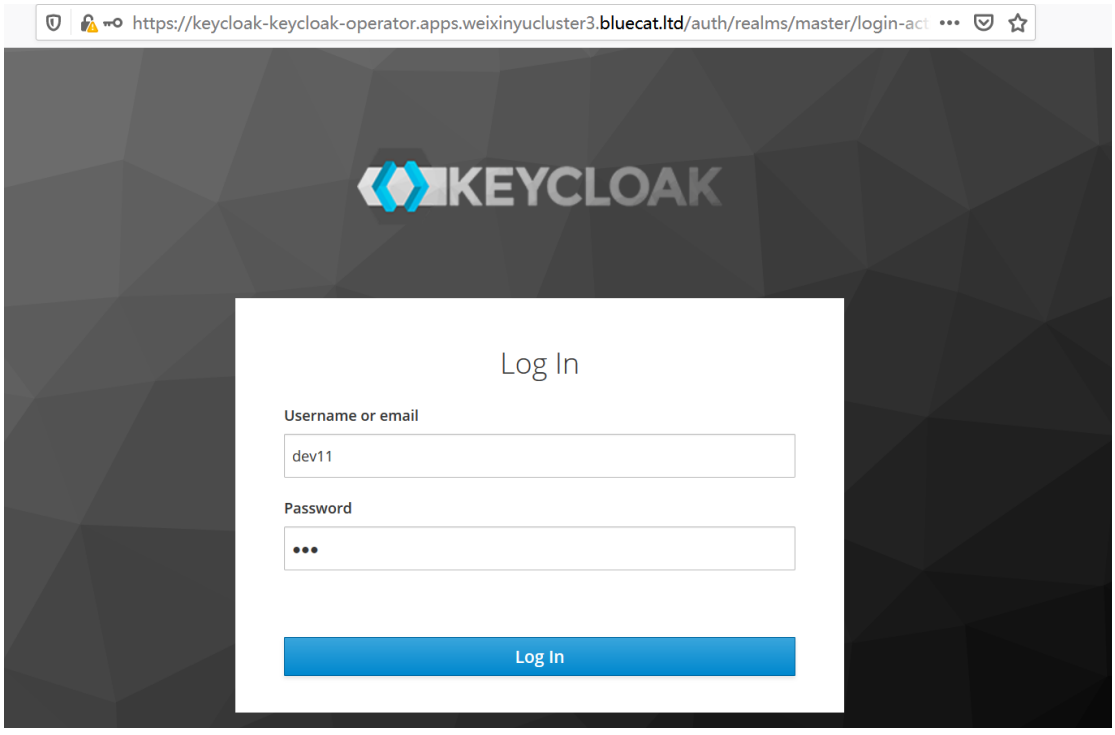
此时，使用集群管理员，我们可以看到所有新创建的项目：

```
[root@lb.weixinyucluster3 ~/orgs-management-ocp]# oc projects |grep -i online
    online-acquisition-credit-score-svc-build
    online-acquisition-credit-score-svc-dev
    online-acquisition-credit-score-svc-prod
    online-acquisition-credit-score-svc-qa
    online-acquisition-fraud-detection-kyc-svc-build
    online-acquisition-fraud-detection-kyc-svc-dev
    online-acquisition-fraud-detection-kyc-svc-prod
    online-acquisition-fraud-detection-kyc-svc-qa
    online-acquisition-kyc-svc-build
    online-acquisition-kyc-svc-dev
    online-acquisition-kyc-svc-prod
    online-acquisition-kyc-svc-qa
    online-acquisition-login-svc-build
    online-acquisition-login-svc-dev
    online-acquisition-login-svc-prod
    online-acquisition-login-svc-qa
    online-alerts-mobile-notification-svc-build
    online-alerts-mobile-notification-svc-dev
    online-alerts-mobile-notification-svc-prod
    online-alerts-mobile-notification-svc-qa
    online-alerts-sms-svc-build
    online-alerts-sms-svc-dev
    online-alerts-sms-svc-prod
    online-alerts-sms-svc-qa
    online-banking-bill-payment-svc-build
    online-banking-bill-payment-svc-dev
    online-banking-bill-payment-svc-prod
    online-banking-bill-payment-svc-qa
    online-banking-checking-account-svc-build
    online-banking-checking-account-svc-dev
    online-banking-checking-account-svc-prod
    online-banking-checking-account-svc-qa
    online-banking-investment-account-svc-build
    online-banking-investment-account-svc-dev
    online-banking-investment-account-svc-prod
    online-banking-investment-account-svc-qa
    online-banking-login-svc-build
```

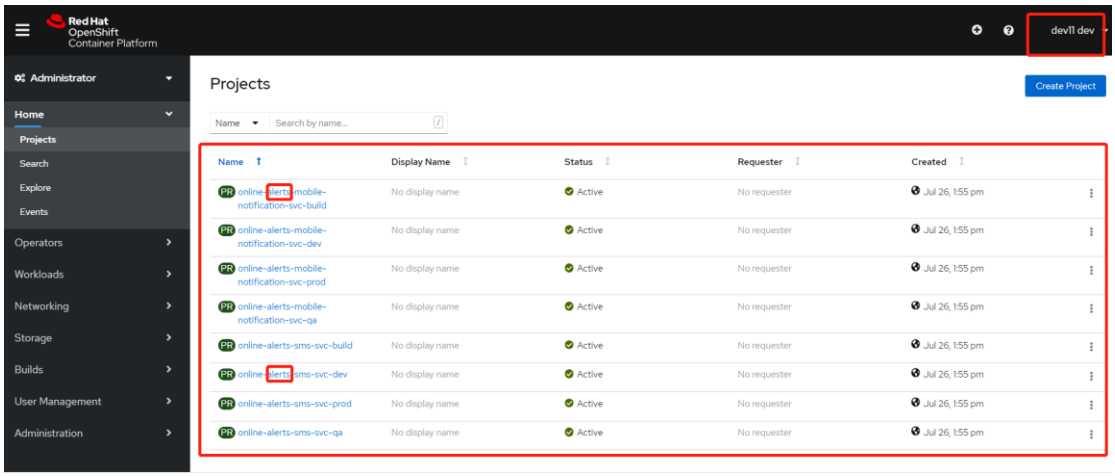
```
online-banking-login-svc-dev
online-banking-login-svc-prod
online-banking-login-svc-qa
online-banking-money-transfer-svc-build
online-banking-money-transfer-svc-dev
online-banking-money-transfer-svc-prod
online-banking-money-transfer-svc-qa
```

此时，如果我们以开发人员之一的身份进行连接，则应该只看到租户所管辖的项目，并且应该能够开始工作。我们使用 alerts-team 的 dev10 登录：





Dev11 只能看到 alerts-team 下属的项目：



https://www.openshift.com/blog/orgs-management-and-team-onboarding-in-openshift-a-fully-automated-approach