

VMware Advanced Customer Engagements (ACE) Team

Enterprise PKS Identity Management

Authored by Riaz Mohamed & Raghu Pemmaraju
April 2020

Table of Contents

Introduction	3
Personas and Roles	5
Role Definitions	5
Enterprise PKS roles	5
Kubernetes Roles:	6
PKS RBAC	6
Harbor Roles:	7
Personas	8
Identity Management Endpoints	9
Group Definitions	10
LDAP – As an Identity Endpoint	12
Required LDAP Groups Configuration	13
Setup Users and Groups.....	13
Create Users	16
Assign Users to Groups	18
EPMC LDAP settings	24
Configure Users and Groups from EPMC	26
Map LDAP Group to PKS Admin roles	26
Map LDAP Group to PKS Manage roles	27
Map LDAP Group to PKS Read roles	28
Login to a PKS (Revisit).....	29
NON EPMC DEPLOYMENT	30
PKS LDAP Integration	30
Configuring Groups in UAA	31
OpsMan LDAP Integration.....	32
OpsMan RBAC group.....	32
OpsMan Tile LDAP settings.....	33
Azure Active Directory as a SAML Identity Endpoint	35
Prerequisites	35
Configure SAML in Azure AD.....	35
Claims	39
UAA Scopes for Enterprise PKS Users.....	42
Sign on using Enterprise PKS CLI	43
Troubleshooting:	45
Harbor Identity Management	49
Harbor AD/LDAP as an identity Management end point.....	49
Group definitions	50
Harbor LDAP configuration	51
RBAC Authorization to K8 Clusters	53
Create Bindings.....	53
Roles And Role Bindings.....	54
KubeConfig.....	61

Introduction

In this document, we provide an overview of Enterprise PKS Roles and Responsibility and detail steps to configure the supported Enterprise PKS (PKS) identity providers. It is essential to understand the options available when configuring Identity management for PKS (PKS) to ensure the principle of least privileges are applied, and authorized access is provided to the users to ensure the integrity and security of PKS and PKS managed Kubernetes clusters. The steps provided are based on PKS 1.6 and above. The steps here apply to installing and configuring PKS using PKS Management Console (EPMC) or a standalone installation.

PKS consists of several components that require careful consideration to provide authentication and authorization. PKS users can gain access to PKS Control Plane, and PKS managed Kubernetes clusters using the OpenID Connect provider called User Account and Authentication (UAA). UAA is an OAuth2 provider issues tokens to PKS and Kubernetes cluster administrators and users with the appropriate roles. Roles can be cluster administrator, developer, or someone with custom access.

We discuss all the necessary configuration steps in this document, starting with a table that shows all of the PKS components that require user access. PKS supports three types of identity management: endpoints, local accounts, LDAP and SAML integrations.

Component	User Access	API	User
Enterprise PKS Management Console (EPMC)	Local only (EPMC 1.7 will support LDAP and SAML)		root
OpsMan	Local ¹ or LDAP or SAML		admin (Found in EPMC metadata)
Bosh	Local or LDAP or SAML		admin (Found in EPMC metadata)
PKS	1. Local Account 2. Integrate to LDAP or SAML	pks	LDAP users For local accounts admin (Found in EPMC metadata)
K8 clusters	Integrate to LDAP or SAML	K8 API	LDAP users

¹ Local Only when using EPMC as all metadata is stored in EPMC



Harbor	1. Local Account 2. Integrate to LDAP or SAML	Harbor	LDAP users For local accounts admin (Found in EPMC metadata)
--------	--	--------	---

EMPC - VMware Enterprise PKS Management Console provides a unified installation experience for deploying Enterprise Pivotal Container Service (Enterprise PKS) to vSphere.

OpsMan - Platform provides a set of APIs and a graphical interface to manage the deployment and upgrade of PKS components

Bosh - Unifies release engineering, deployment, and lifecycle management of small and large-scale PKS clusters

PKS - is an enterprise Kubernetes platform, architected for rapid results, scaling, and reliability on any infrastructure

K8 Cluster access - The recommended approach to access Kubernetes clusters provisioned via PKS is to use Role-based access control (RBAC) to regulate access to computer or network resources based on the roles of individual users within an enterprise. More information can be found [here](#).

Harbor - Harbor is an open source container image registry that secures images with role-based access control, scans images for vulnerabilities, and signs images as trusted

Personas and Roles

An organization that deploys PKS should keep in mind three types components and personas:

1. PKS Control Plane
 - a. Administrators
 - b. Users/Developers
2. Kubernetes Clusters
 - a. Administrators
 - b. Developers
 - c. Users
3. Harbor Registry
 - a. Administrators
 - b. Users/Developers

Let us go into the details.

Role Definitions

Roles are a set of defined rights that are assumed by users associated to it to perform specific operations. Role-based access control (RBAC) is a method of regulating access to computer or network resources based on the roles of individual users within your organization.

Enterprise PKS roles

PKS defines a set of roles that are applicable to the PKS control plane. PKS access gives users the ability to deploy and manage Kubernetes clusters. This includes creating clusters, upgrading clusters, deleting clusters, creating network profiles, scaling cluster nodes etc. There are three roles or UAA profiles within PKS:

pks.clusters.manage: Accounts with this scope can create and access their own clusters

pks.clusters.admin: Accounts with this scope can create and access all clusters.

pks.clusters.admin.read: Accounts with this scope can access any information about all clusters except for cluster credentials.

It is straightforward that pks.clusters.admin should be assigned to an individual or group responsible for all of the clusters created by the PKS control plane.

You can assign pks.clusters.manage role that you want them to create clusters, but not let them view clusters they have not created or those created by someone with pks.clusters.admin role. Be careful who is given access to pks.clusters.manage and pks.clusters.admin

pks.clusters.admin.read role is best for a developer. With this role, they can have readonly access to all of the clusters and they can issue 'pks get-kubeconfig' command to get kubeconfig for the Kubernetes cluster they have access.

Kubernetes Roles:

An RBAC Role or ClusterRole contains rules that represent a set of permissions. Permissions are purely additive (there are no "deny" rules).

A Role always sets permissions within a particular namespace; when you create a Role, you have to specify the namespace it belongs in.

ClusterRole, by contrast, is a non-namespaced resource. The resources have different names (Role and ClusterRole) because a Kubernetes object always has to be either namespaced or not namespaced; it can't be both.

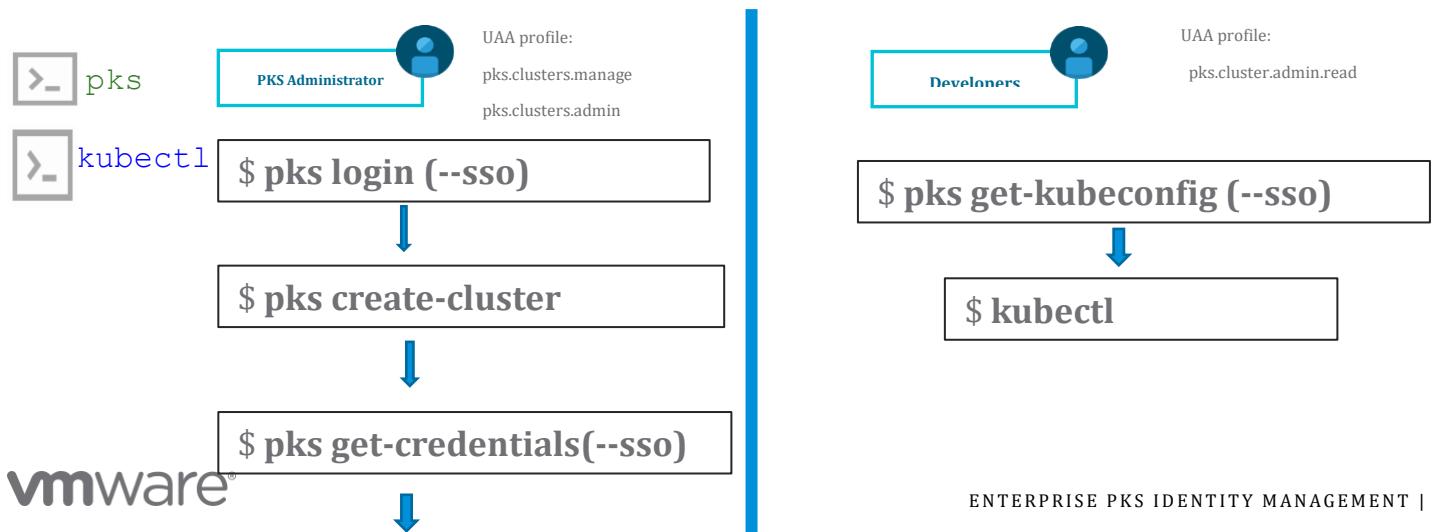
ClusterRoles have several uses. You can use a ClusterRole to:

- define permissions on namespaced resources and be granted within individual namespace(s)
- define permissions on namespaced resources and be granted across all namespaces
- define permissions on cluster-scoped resources

If you want to define a role within a namespace, use a Role; if you want to define a role cluster-wide, use a ClusterRole.

PKS RBAC

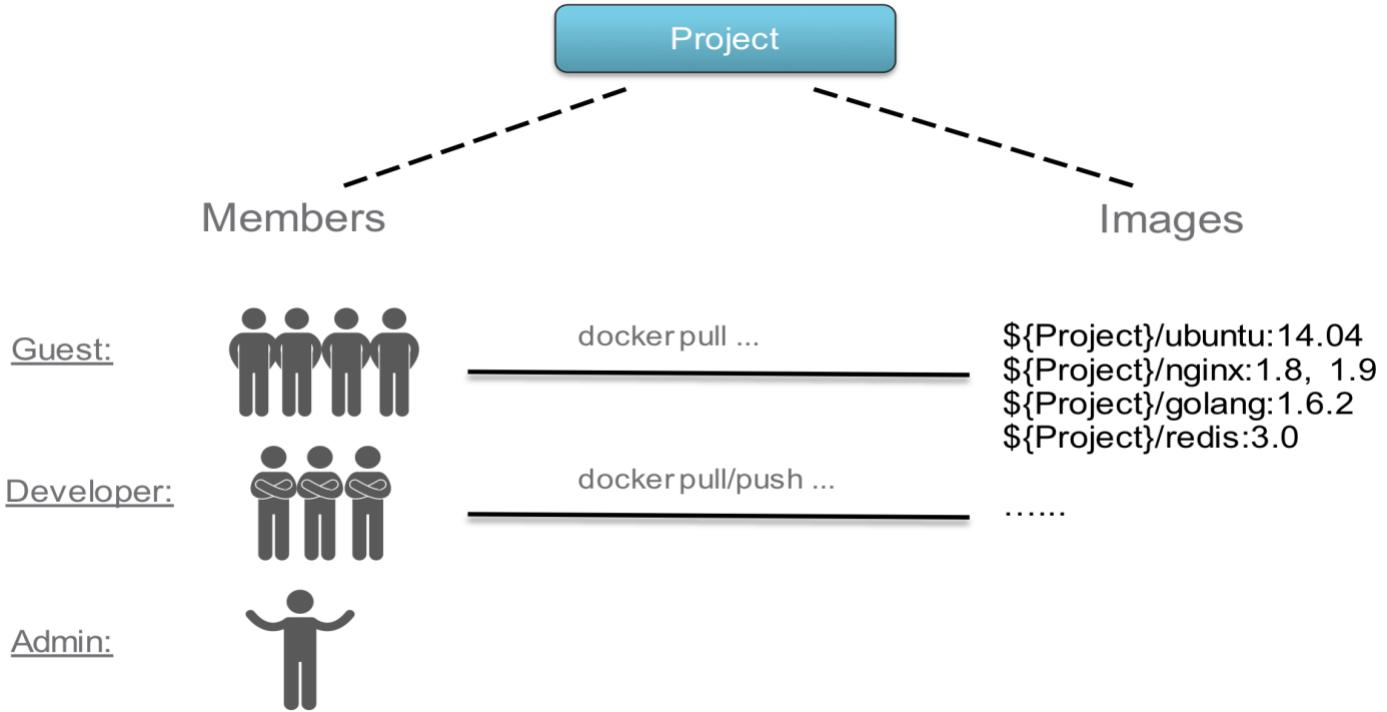
The diagram below shows how different personals gain access to PKS and Kubernetes.



```
$ kubectl
```

Harbor Roles:

Harbor manages images through projects. You provide access to these images to users by including the users in projects and assigning one of the following roles to them.



Limited Guest: A Limited Guest does not have full read privileges for a project. They can pull images but cannot push, and they cannot see logs or the other members of a project. For example, you can create limited guests for users from different organizations who share access to a project.

Guest: Guest has read-only privilege for a specified project. They can pull and retag images, but cannot push.

Developer: Developer has read and write privileges for a project.

Master: Master has elevated permissions beyond those of 'Developer' including the ability to scan images, view replication jobs, and delete images and helm charts.

ProjectAdmin: When creating a new project, you will be assigned the “ProjectAdmin” role to the project. Besides read-write privileges, the “ProjectAdmin” also has some management privileges, such as adding and removing members, starting a vulnerability scan.

Besides the above roles, there are two system-level roles:

Harbor system administrator: “Harbor system administrator” has the most privileges. In addition to the privileges mentioned above, “Harbor system administrator” can also list all projects, set an ordinary user as administrator, delete users and set vulnerability scan policy for all images. The public project “library” is also owned by the administrator.

Anonymous: When a user is not logged in, the user is considered as an “Anonymous” user. An anonymous user has no access to private projects and has read-only access to public projects.

Personas

To help you understand how we plan the various personas found in an organization, we will use the following personas.

Alana

Alana is a PKS Administrator and Operator. This person is involved in the following aspects of Pivotal Container Service:

- Initial PKS Deployment and Installation activities
- PKS platform upgrades and systems patches
- PKS Tile Administration and the PKS control plans
- Setting PKS Role-Based Access Control for the PKS control plane. This person provides access and authorization to any underlying PKS Cluster Administrators.

Cody

Cody could create a PKS cluster and become its PKS cluster Admin. This person is involved in the following aspects of Pivotal Container Service:

- Can create a PKS cluster through mechanism of pkcs cli: pkcs create-cluster ...
- Can be the PKS Cluster Administrator.
- Sets Kubernetes Role-Based Access Control to a specific PKS Cluster for End-users and Developers within the Kubernetes control plane.
- Assume the role of a Harbor system administrator
- Setup User access to harbor projects

Naomi

Naomi is an end-user or developer on a Kubernetes cluster. This person is involved in the following aspects of Pivotal Container Service:

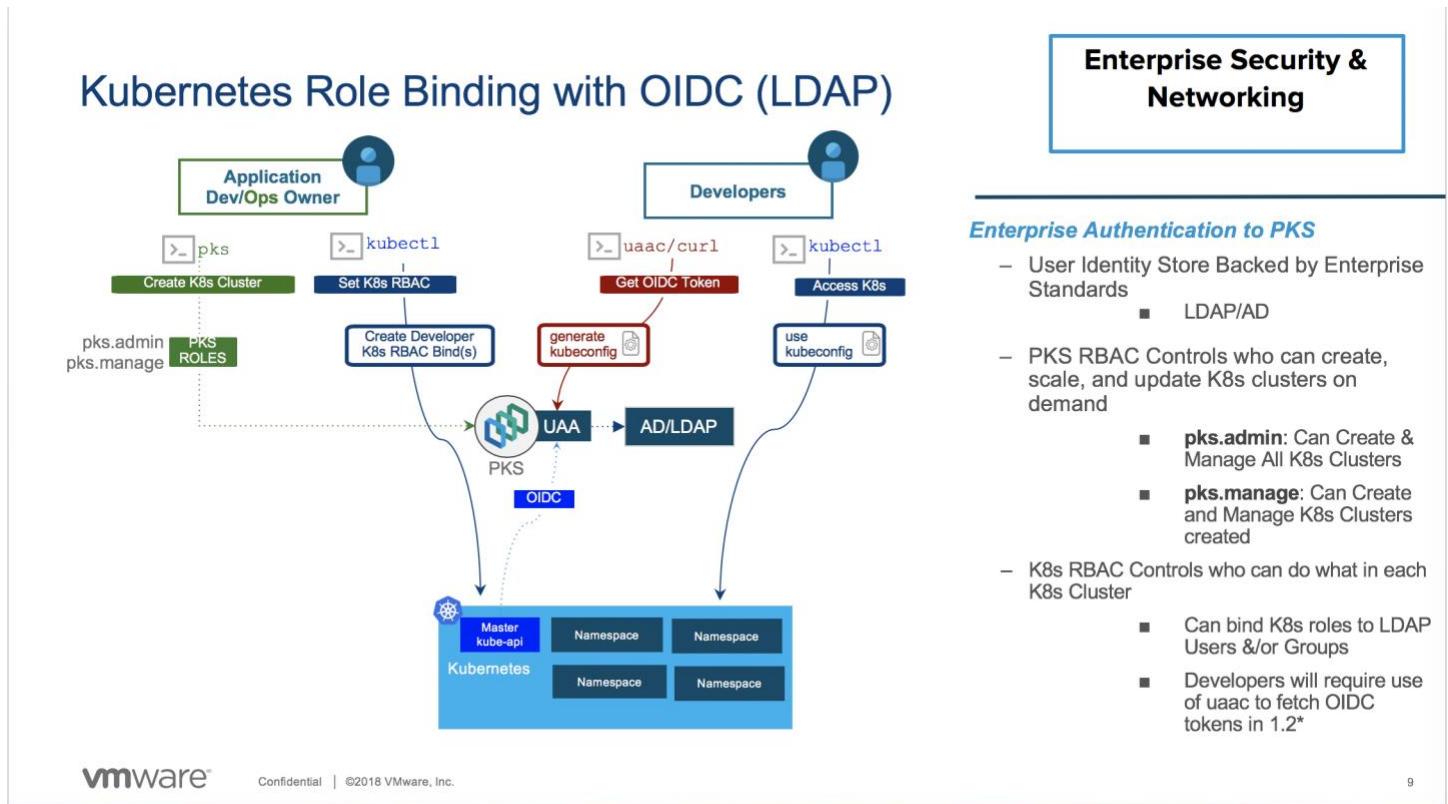
- Access a Kubernetes API for the cluster they have access to
- Can use the `kubectl` cli, *Kubernetes Dashboard*, or other Kubernetes API clients to interact with a specific PKS Cluster
- Harbor Developer access

Scott

Scott is an application developer. This person is involved in the following aspects of Pivotal Container Service:

- Access a Kubernetes API for the cluster they have access to with specific granular roles
- Can use the `kubectl` cli, *Kubernetes Dashboard*, or other Kubernetes API clients to interact with a specific PKS Cluster
- Harbor Developer access

Identity Management Endpoints



PKS supports the following identity management endpoints

- Local User Database – Uses UAA

- AD/LDAP
- SAML Identity provider

Harbor supports the following identity management endpoints

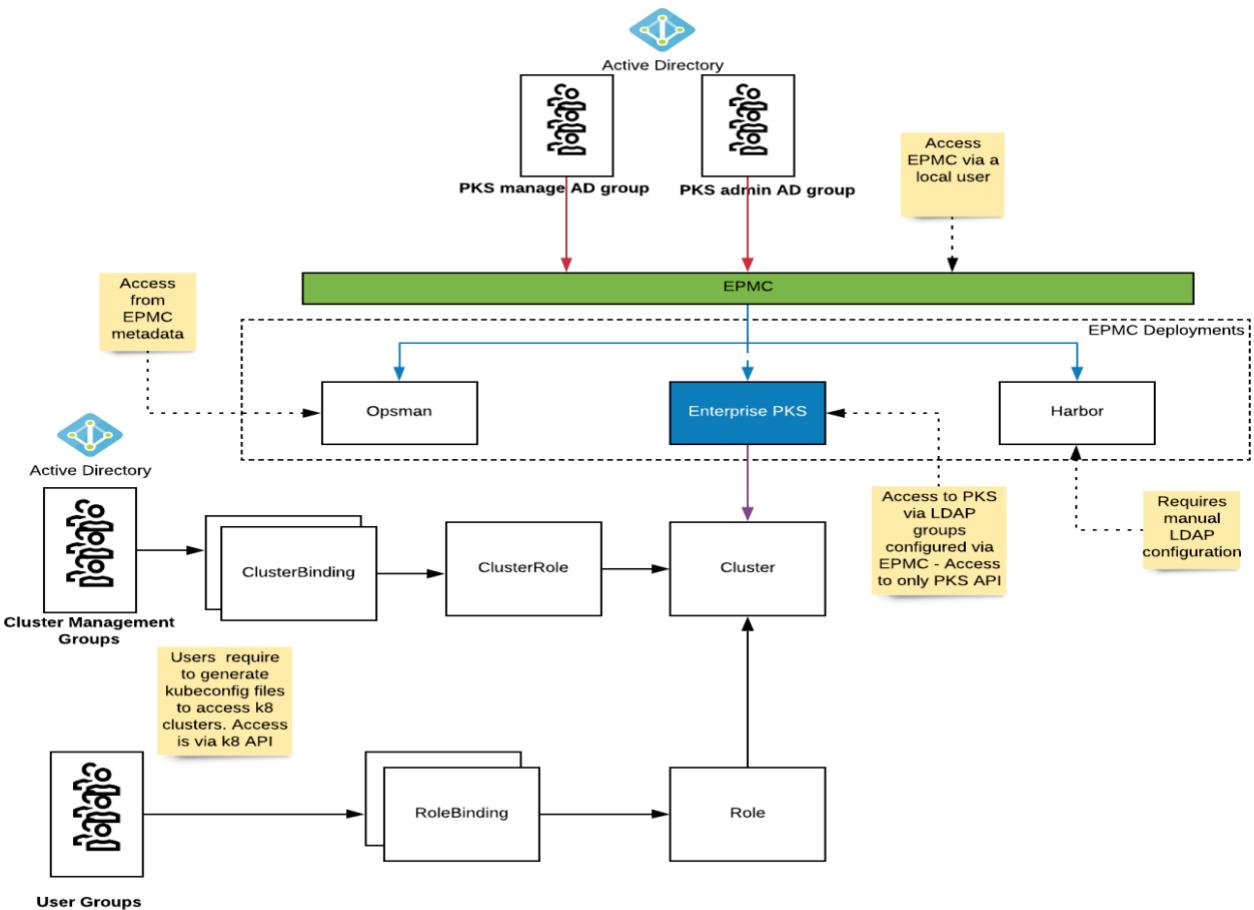
- Harbor internal user management
- AD/LDAP
- UAA in Pivotal Container Service

Group Definitions

This section defines the usergroups we will be creating within the different identity management endpoints to associate the different personas

Group	Persona	Role
pksclusteradmin	alana	pks.cluster.admin
pksclustermanage	cody	pks.cluster.manage
pksclusterread	naomi scott	pks.cluster.read
k8clusteradmins	naomi	cluster-admin
k8developers	scott	create, list , get on pods in namespace dataengg
harboradmins	cody naomi	Harbor system administrator
harborusers	scott	Harbor developer

LDAP – As an Identity Endpoint



This section provides the steps to set up LDAP that we would be using in the later sections to configure PKS components.

For this setup, we would be using Microsoft Active Directory (AD) as the LDAP provider.

The document assumes to have all users under

`CN=Users,DC=corp,DC=local`

The groups definitions are created under

`OU=pks,OU=Infra,DC=corp,DC=local`

NOTE: You could have different OU where your user's exists eg. IT(OU)/Offices(OU)/Boston

Required LDAP Groups Configuration

This section provides an overview on setting up LDAP groups and users that we would be using in the later sections. The LDAP Groups settings will be as follows:

Setup Users and Groups

This section provides the steps to create ou's, groups, users and assign users to groups.

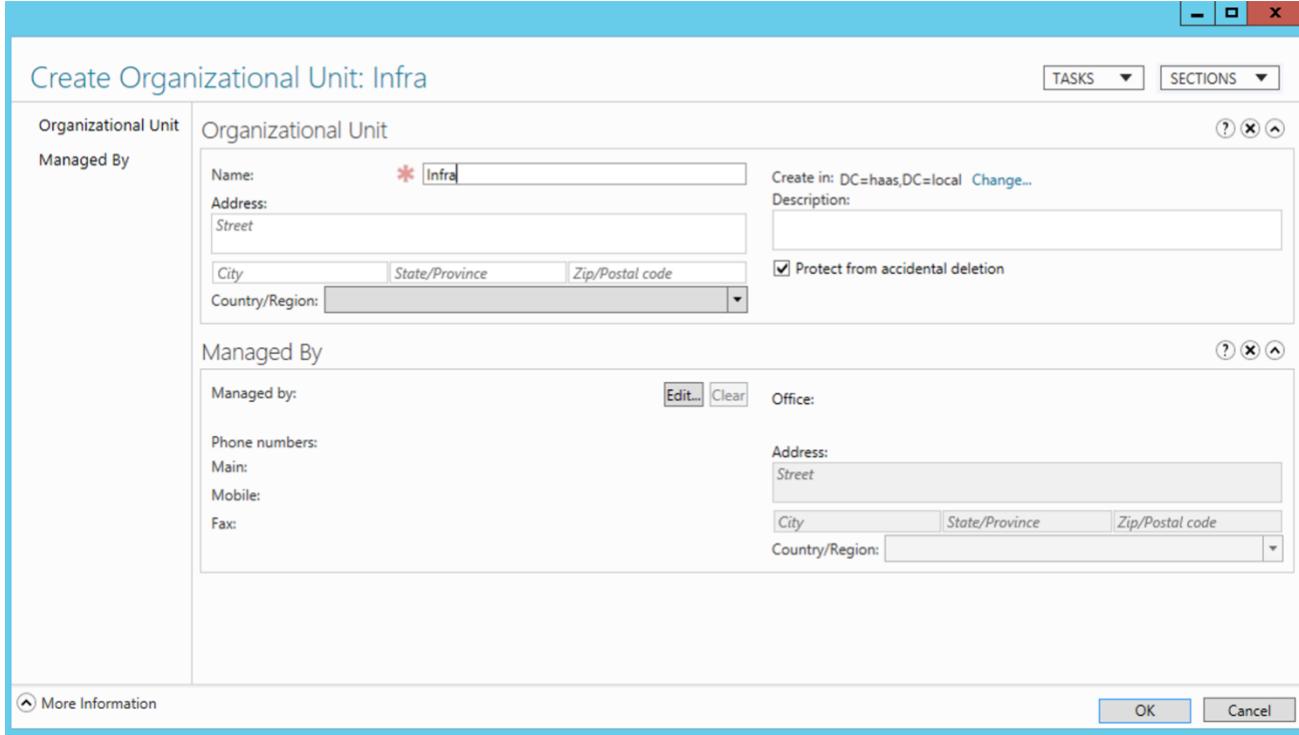
Create Groups

These groups will be used to configure access to different components within the PKS infrastructure.

Step 1: Select New Organizational Unit:

The screenshot shows the Active Directory Administrative Center interface. The left navigation pane is visible with options like Active Directory, Overview, haas (local), Dynamic Access Control, Authentication, and Global Search. The main pane displays a list of objects under 'haas (local)'. A context menu is open over the 'Builtin' object, which is highlighted in blue. The menu tasks include 'New' (selected), 'Delete', 'Properties', and other options like 'Change domain controller', 'Raise the forest functional level...', 'Raise the domain functional le...', and 'Enable Recycle Bin ...'. Below the list, a detailed view of the 'Builtin' object is shown with fields for Object class: builtinDomain, Description:, and Modified: 12/12/2019 6:00 AM. At the bottom right, there is a message to 'Activate Windows' with a link to 'Go to System in Control Panel to activate Windows.'

Step 2: Enter Name e.g. Infra and click OK. The OU must be created:

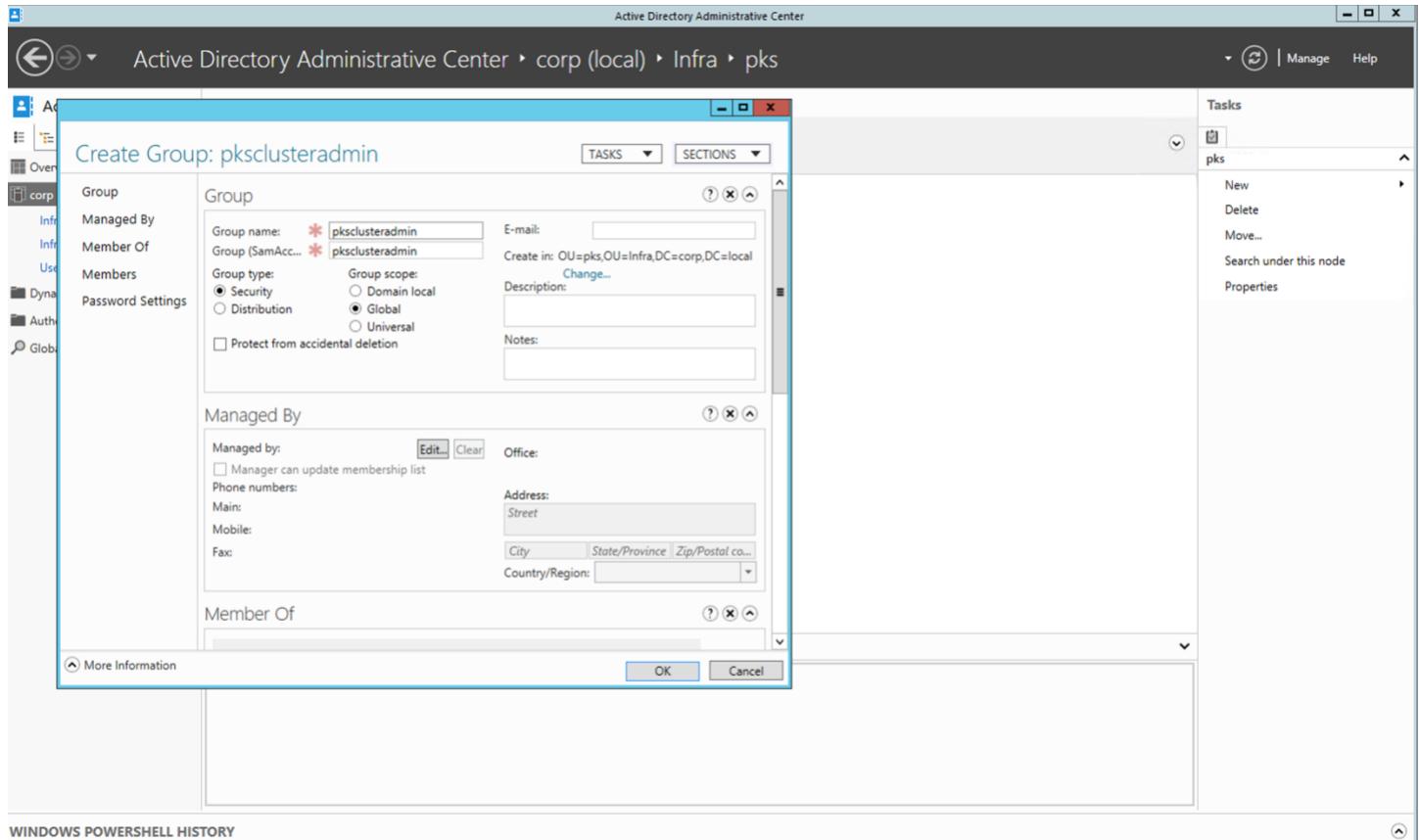


Step 3: Drill down to Infra and create another OU pks:

The screenshot shows the Active Directory Administrative Center. The left navigation pane is expanded to show 'haas (local) > Infra'. The main pane displays a table with one entry: 'pks' (Type: Organizational Unit). On the right, a context menu is open over the 'pks' entry, listing options: Tasks (New, Delete, Move...), Search under this node, Properties, and a separator line followed by 'Infra' (New, Delete, Move..., Search under this node, Properties).

Step 4: Create Groups under pk - Select pk and click on New Group

Step 5: Create the pksclusteradmin group



Step 6: Create the other groups under the pks OU – pksclustermanage, pksclusterread, k8clusteradmins, k8developers, harboradmins, harborusers

The screenshot shows the Active Directory Administrative Center interface. The left navigation pane shows the hierarchy: Active Directory... > corp (local) > Infra > pks. The main pane displays a list of groups under 'pks': harboradmins, harborusers, k8clusteradmins (selected), k8developers, pksclusteradmin, pksclustermanage, and pksclusterreadonly. The right pane shows the 'Tasks' context menu for 'k8clusteradmins', which includes options like 'Add to another group...', 'Delete', 'Move...', and 'Properties'. Below the main pane, a detailed view of 'k8clusteradmins' is shown, including its E-mail, Managed by, Modified date (4/10/2020 10:19 AM), and Description fields. A summary section is also present.

Create Users

Create users under each BU as per the following table

Step 1: Create the following users under CN=Users,DC=corp,DC=local

NOTE: You could create your own org structure for users.

Step 2: Enter 'alana' for both Full Name and User SamAccountName, Select the option of password never expires and enter a password (E.g. <password>)

Create User: alana

Account	Account First name: alana Middle initials: Last name: Full name: alana User UPN logon: User SamAccountName l... corp\alana Password: ***** Confirm password: ***** Create in: CN=Users,DC=corp,DC=local Change... <input type="checkbox"/> Protect from accidental deletion Log on hours... Log on to...	Account expires: <input checked="" type="radio"/> Never <input type="radio"/> End of <input type="text"/> Password options: <input type="radio"/> User must change password at next log on <input checked="" type="radio"/> Other password options <input type="checkbox"/> Smart card is required for interactive log on <input checked="" type="checkbox"/> Password never expires <input type="checkbox"/> User cannot change password Encryption options: Other options:
Organization	Organization Display name: alana Office: E-mail: Web page: Other web pages...	Job title: PKS Administrator Department: Company: Manager: Edit... Clear Direct reports: Add

[More Information](#)

OK Cancel

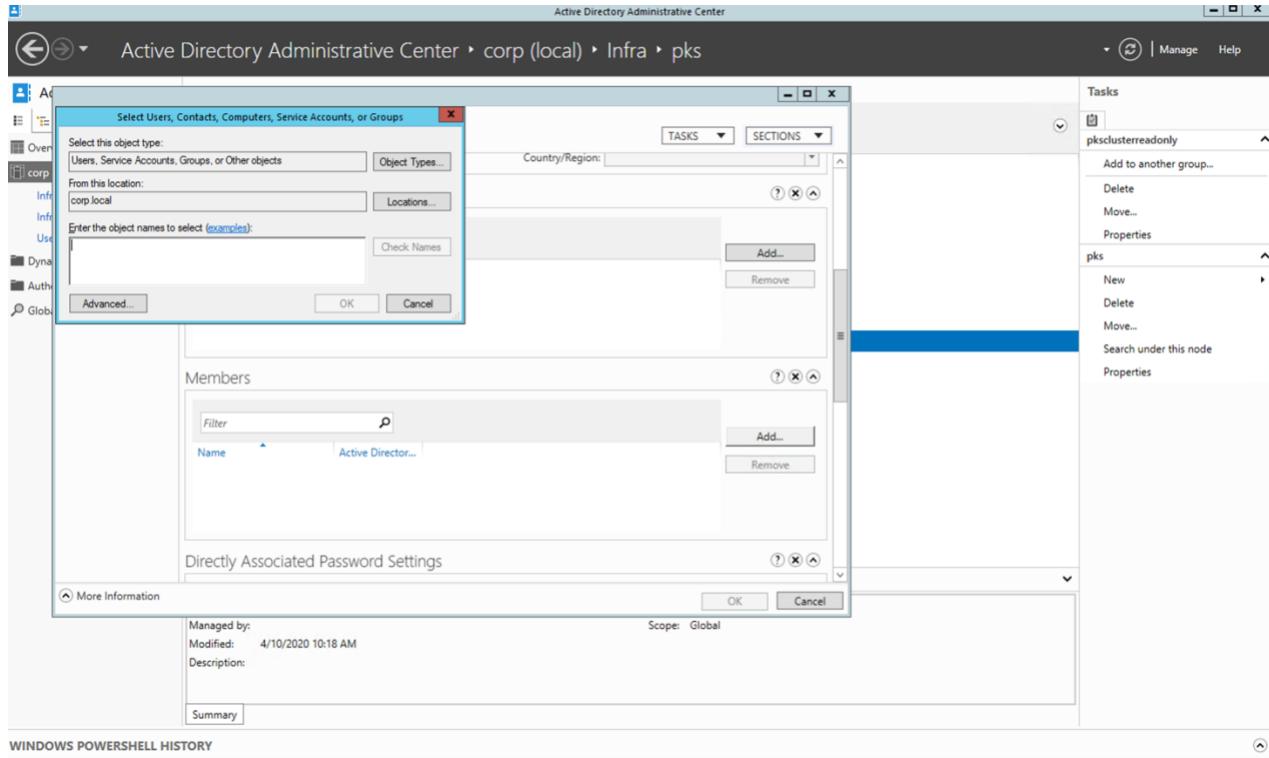
Step 3: Repeat the above steps to create the other personas cody, naomi, scott,

Assign Users to Groups

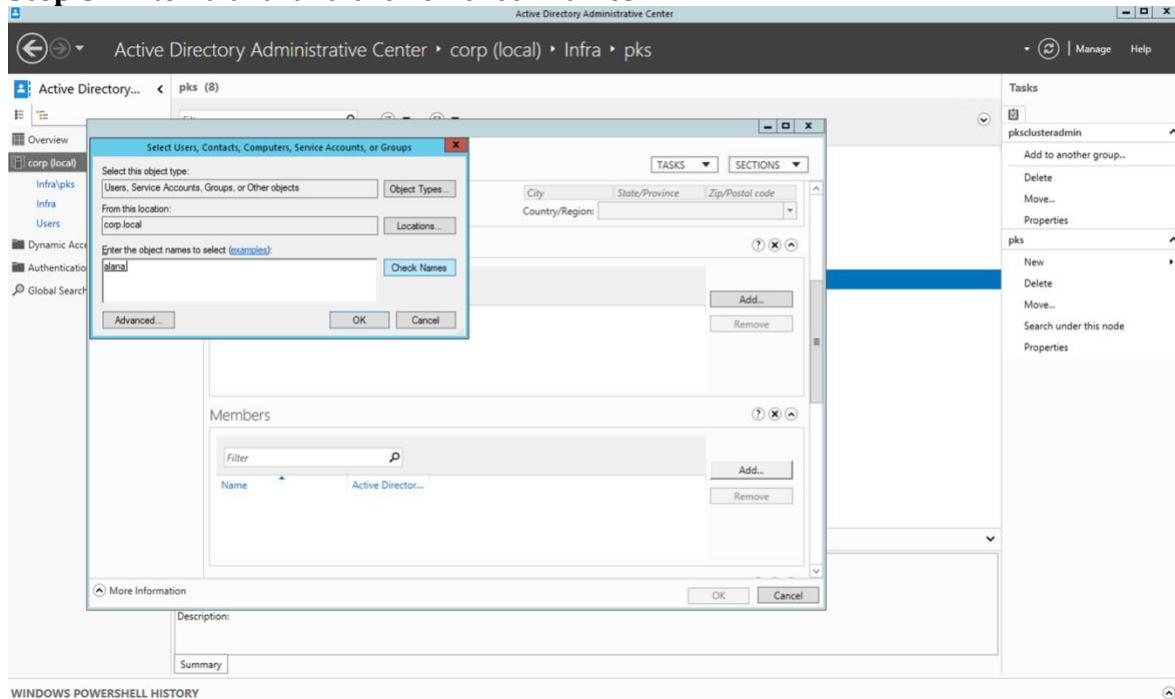
Step 1: Drill down to Infra/pks/pksclusteradmin group

The screenshot shows the 'pksclusteradmin' group configuration page. The left sidebar lists navigation options: Group, Managed By, Member Of, Members, Password Settings, and Extensions. The main content area is divided into sections: 'Group', 'Managed By', and 'Member Of'. The 'Group' section contains fields for Group name (pksclusteradmin), Group (SamAccountName) (pksclusteradmin), E-mail, Description, and Notes. It also includes Group type (Security, selected), Group scope (Global, selected), and a checkbox for Protect from accidental deletion. The 'Managed By' section includes fields for Managed by, Office, Address (Street), City, State/Province, Zip/Postal code, and Country/Region. The 'Member Of' section features a search bar with Filter and Add... buttons.

Step 2: Under members click on Add



Step 3: Enter 'alana' and click on check names



Step 4: Click on OK

Step 5: Select the appropriate group and add users as per the table

Group	Persona
pksclusteradmin	alana
pksclustermanage	cody
pksclusterread	naomi scott
k8clusteradmins	naomi
k8developers	scott
harboradmins	cody
harborusers	scott

PKS LDAP Attributes

This section covers the basic LDAP attributes required to configure PKS for LDAP integration. Use an LDAP Browser like LDAPSoft AD Browser if not familiar with how to get values for some of these attributes.

LDAP Username : CN=Administrator,CN=Users,DC=corp,DC=local

User Search Base:

Select 'Users', right click and Copy DN
CN=Users,DC=corp,DC=local

Enterprise PKS Identity Management

Screenshot of the Enterprise PKS Identity Management interface showing the LDAP browser and attribute editor.

Left Panel (LDAP Browser):

- Path: DC=corp,DC=local
 - CN=Builtin
 - CN=Computers
 - CN=ForeignSecurityPrincipals
 - CN=Infrastructure
 - CN=LostAndFound
 - CN=Managed Service Accounts
 - CN=NTDS Quotas
 - CN=Program Data
 - CN=System
 - CN=TPM Devices
 - CN=Users
 - OU=Do... (highlighted)
 - OU=Inf...

Right Panel (Attribute Editor):

Attribute Name	Value	Size	Type/Editor	Required
objectClass	top	3	ObjectClass	Y
objectClass	container	9	ObjectClass	Y
cn	Users	5	Text	Y
instanceType	4	1	Integer	Y
objectCategory	CN=Container,CN=Schema,CN=Configuration,DC=corp,DC=local	56	DN	Y
nTSecurityDescriptor		0	Text	Y
createTimeStamp	20140130205609.0Z (Thu Jan 30 2014 12:56:09 GMT-0800)	17	Operational	N
description	Default container for upgraded user accounts	44	Text	N
distinguishedName	CN=Users,DC=corp,DC=local	25	DN	N
dScorePropagationData	20200410171704.0Z (Fri Apr 10 2020 10:17:04 GMT-0700)	17	Generalized Time	N
dScorePropagationData	20140130205845.0Z (Thu Jan 30 2014 12:58:45 GMT-0800)	17	Generalized Time	N
dScorePropagationData	16010101000417.0Z (Sun Dec 31 1600 16:04:17 GMT-0800)	17	Generalized Time	N
isCriticalSystemObject	TRUE	4	Boolean	N
modifyTimeStamp	20140130205609.0Z (Thu Jan 30 2014 12:56:09 GMT-0800)	17	Operational	N
name	Users	5	Text	N
objectGUID	(E105BD04-7446-42D4-9D19-2CA5652A9C13)	16	objectGUID	N
showInAdvancedViewOnly	FALSE	5	Boolean	N
structuralObjectClass	top	3	Operational	N
structuralObjectClass	container	9	Operational	N
subSchemaSubEntry	CN=Aggregate,CN=Schema,CN=Configuration,DC=corp,DC=local	56	Operational	N
systemFlags	-1946157056	11	Integer	N
uSNChanged	5821	4	uSNChanged	N
uSNCreated	5821	4	uSNCreated	N
whenChanged	20140130205609.0Z (Thu Jan 30 2014 12:56:09 GMT-0800)	17	Generalized Time	N
whenCreated	20140130205609.0Z (Thu Jan 30 2014 12:56:09 GMT-0800)	17	Generalized Time	N
adminDescription		0	Text	N
adminDisplayName		0	Text	N
allowedAttributes		0	OID	N
allowedAttributesEffective		0	OID	N
allowedChildClasses		0	OID	N
allowedChildClassesEffective		0	OID	N
bridgeheadServerListBL		0	DN	N
canonicalName		0	Text	N
defaultClassStore		0	DN	N
directReports		0	DN	N
displayName		0	Text	N
displayNamePrintable		0	Text	N
dSAcceptableSignature		0	Text	N
extensionName		0	Text	N
flags		0	Integer	N
fromEntry		0	Boolean	N
frsComputerReferenceBL		0	DN	N

Group Search Base

The groups definitions are created under

OU=pks,OU=Infra,DC=corp,DC=local

	Attribute Name	Value	Size	Type/Editor	Required
DC=corp,DC=local	objectClass	top	3	ObjectClass	Y
CNs=Builtin	objectClass	organizationalUnit	18	ObjectClass	Y
CNs=Computers	instanceType	4	1	Integer	Y
CNs=ForeignSecurityPrincipals	objectCategory	CN=Organizational-Unit,CN=Schema,CN=Configuration,DC=corp,DC=local	66	DN	Y
CNs=Infrastructure	ou	pks	3	Text	Y
CNs=LostAndFound	nTSecurityDescriptor		0	Text	Y
CNs=Managed Service Accounts	createTimeStamp	20200410171716.0Z (Fri Apr 10 2020 10:17:16 GMT-0700)	17	Operational	N
CNs=NTDS Quotas	distinguishedName	OU=pks,OU=Infra,DC=corp,DC=local	32	DN	N
CNs=Program Data	dsCorePropagationData	20200410171716.0Z (Fri Apr 10 2020 10:17:16 GMT-0700)	17	Generalized Time	N
CNs=System	dsCorePropagationData	16010101000000.0Z (Sun Dec 31 1600 16:00:00 GMT-0800)	17	Generalized Time	N
CNs=TPM Devices	modifyTimeStamp	20200410171716.0Z (Fri Apr 10 2020 10:17:16 GMT-0700)	17	Operational	N
CNs=Users	name	pks	3	Text	N
OU=Domain Controllers	objectGUID	(0EA19B38-7322-40BC-B0B8-C2155C1BF948)	16	objectGUID	N
OU=Infra	structuralObjectClass	top	3	Operational	N
OU=pks	structuralObjectClass	organizationalUnit	18	Operational	N
CN=harboradmins	subSchemaSubEntry	CN=Aggregate,CN=Schema,CN=Configuration,DC=corp,DC=local	56	Operational	N
CN=harboursers	uSNChanged	997875	6	uSNChanged	N
CN=k8sclusteramins	uSNCreated	997873	6	uSNCreated	N
CN=k8sdevelopers	whenChanged	20200410171716.0Z (Fri Apr 10 2020 10:17:16 GMT-0700)	17	Generalized Time	N
CN=PKS Dev	whenCreated	20200410171716.0Z (Fri Apr 10 2020 10:17:16 GMT-0700)	17	Generalized Time	N
CN=pksclusteradmin	whenCreated	20200410171716.0Z (Fri Apr 10 2020 10:17:16 GMT-0700)	17	Generalized Time	N
CN=pksclustermanage	adminDescription		0	Text	N
CN=pksclusterreadonly	adminDisplayName		0	Text	N
	allowedAttributes		0	OID	N
	allowedAttributesEffective		0	OID	N
	allowedChildClasses		0	OID	N
	allowedChildClassesEffective		0	OID	N
	bridgeheadServerListBL		0	DN	N
	businessCategory		0	Text	N
	c		0	Text	N
	canonicalName		0	Text	N
	cn		0	Text	N
	co		0	Text	N
	countryCode		0	Integer	N
	defaultGroup		0	DN	N
	description		0	Text	N
	desktopProfile		0	Text	N
	destinationIndicator		0	Text	N
	directReports		0	DN	N
	displayName		0	Text	N
	displayNamePrintable		0	Text	N
	dsSignature		0	Text	N
	extensionName		0	Text	N

EPMC LDAP settings

This section shows the configuration settings on EPMC to install PKS. To configure LDAP through EPMC, select AD/LDAP as the Identity management Service during PKS installation through EPMC.

NOTE: Since EPMC is the management console, we would require to only configure LDAP for PKS. In ‘traditional’ (tiles based) deployments LDAP would need to be configured for OpsMan as well.

Example LDAP settings provided to EPMC. Please update with your values and use during installation.

Description	Value
Server URL (Only takes an IP)	ldap://192.168.110.10
LDAP Username	CN=Administrator,CN=Users,DC=corp,DC=local
LDAP Password	<password>
User Search Base	CN=Users,DC=corp,DC=local
User Search Filter	sAMAccountName={0}
Group Search Base	OU=pk,OU=Infra,DC=corp,DC=local
Email	mail

- Configure Created Clusters to use UAA as the OIDC provider – Enabled
- Once LDAP is configured generate configuration and apply to deploy and configure PKS with LDAP

Enterprise PKS Identity Management

For Harbor LDAP integration on the Harbor section of EPMC, select Log in Harbor with LDAP users.

The screenshot shows the VMware Enterprise PKS Management Console interface. On the left, there's a navigation sidebar with sections like Administration, PKS Configuration, PKS Instance Upgrade, and PKS Component Patch. The main area is titled 'PKS Configuration' and shows a 'WIZARD' tab selected. Step 8, '8. Harbor', is currently being configured. It includes fields for 'Enable Harbor', 'Harbor FQDN' (set to 'pksharbor.acelab.local'), 'Authentication' (with a dropdown menu where 'Log in Harbor with LDAP users' is highlighted), 'Password', 'Confirm Password', 'HTTP Proxy' (optional), and 'HTTPS Proxy' (optional). Below these, there are sections for 'Resources' (VM Type for Harbor-App set to 'medium.disk (cpu: 2, ram: 4 GB, disk: 32 GB)'), 'Disk Size for Harbor-App' (set to '30 GB'), and 'Clair Settings' (Install Clair and Update Interval both set to 0). At the bottom right, there's a link to 'Activate Windows'.

NOTE: If this is not selected in the initial install, the Harbor tile will need to be deleted through Opsman and redeployed with the LDAP integration enabled.

A UAA admin user can assign the following UAA scopes to Enterprise PKS users

pks.clusters.manage: Accounts with this scope can create and access their own clusters.

pks.clusters.admin: Accounts with this scope can create and access all clusters.

pks.clusters.admin.read: Accounts with this scope can access any information about all clusters except for cluster credentials.

Configure Users and Groups from EPMC

Once the deployment is complete, click on Identity Management on the EPMC Console and select the groups tab

The screenshot shows the VMware Enterprise PKS Management Console interface. The left sidebar has sections for Administration (Identity Management, Deployment Metadata, Configuration) and Enterprise PKS. The main area is titled 'Identity Management' and shows the 'Groups' tab selected. It includes buttons for 'ADD GROUP' and 'REMOVE GROUP'. A table lists groups with columns for Name and Role, and a search/filter icon. At the bottom, it shows 'Groups per page' set to 10, and '1 - 10 of 0 Groups'.

Map LDAP Group to PKS Admin roles

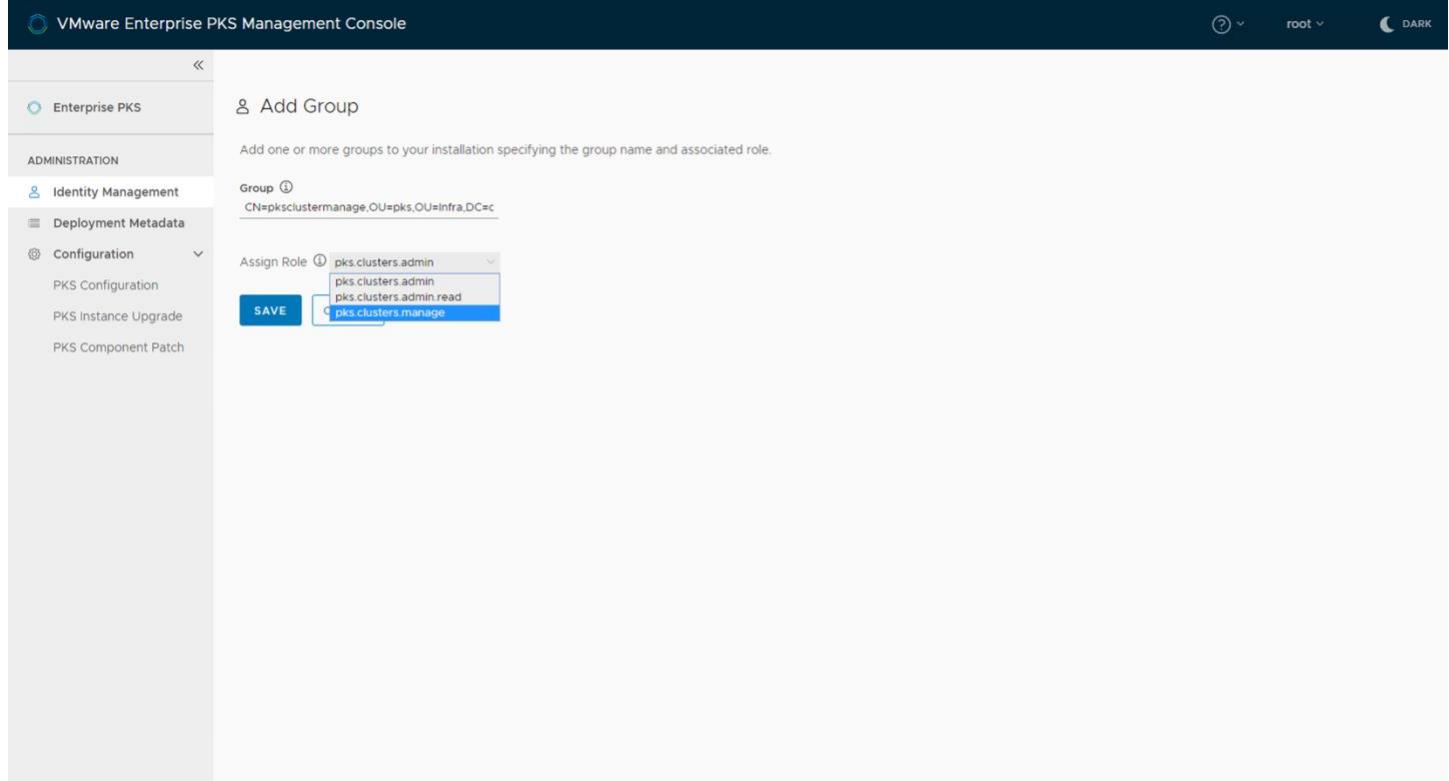
Click on Add Group

The screenshot shows the 'Add Group' dialog in the VMware Enterprise PKS Management Console. The left sidebar is identical to the previous screenshot. The main area is titled 'Add Group' and contains instructions: 'Add one or more groups to your installation specifying the group name and associated role.' It has fields for 'Group' (containing 'CN=pksclusteradmin,OU=pksc,OU=Infra,DC=corp') and 'Assign Role' (containing 'pks.clusters.admin'). There are 'SAVE' and 'CANCEL' buttons at the bottom.

Group Name - DN of the group that need to be added with pks admin roles -
CN=pksclusteradmin,OU=pksc,OU=Infra,DC=corp,DC=local

Map LDAP Group to PKS Manage roles

Click on Add Group, select pks.cluster.manage as role



The screenshot shows the 'Add Group' page in the VMware Enterprise PKS Management Console. The left sidebar is titled 'ADMINISTRATION' and includes sections for 'Identity Management', 'Deployment Metadata', and 'Configuration' (with sub-options: PKS Configuration, PKS Instance Upgrade, PKS Component Patch). The main form has a title 'Add Group' and a subtitle 'Add one or more groups to your installation specifying the group name and associated role.' A 'Group' input field contains the value 'CN=pksclustermanage,OU=pks,OU=Infra,DC=corp,DC=local'. Below it, a 'Assign Role' dropdown menu is open, showing three options: 'pks.clusters.admin', 'pks.clusters.admin.read', and 'pks.clusters.manage'. The 'pks.clusters.manage' option is highlighted with a blue selection bar. A 'SAVE' button is visible at the bottom left of the form.

Group Name - DN of the group that need to be added with pks manage roles. -
CN=pksclustermanage,OU=pks,OU=Infra,DC=corp,DC=local

Map LDAP Group to PKS Read roles

Click on Add Group, select pks.cluster.read as role

VMware Enterprise PKS Management Console

Add Group

Add one or more groups to your installation specifying the group name and associated role.

Group: CN=pksclustermanage,OU=pks,OU=Infra,DC=corp,DC=local

Assign Role:

- pks.clusters.admin
- pks.clusters.admin.read**
- pks.clusters.manage

SAVE

Group Name - DN of the group that need to be added with pks manage roles. -
CN=pksclusterreadonly,OU=pks,OU=Infra,DC=corp,DC=local

VMware Enterprise PKS Management Console

Identity Management

Users Groups

ADD GROUP REMOVE GROUP

Name	Role
cn=pksclusteradmin,ou=pks,ou=infra,dc=corp,dc=local	pks.clusters.admin
cn=pksclustermanage,ou=pks,ou=infra,dc=corp,dc=local	pks.clusters.manage
cn=pksclusterreadonly,ou=pks,ou=infra,dc=corp,dc=local	pks.clusters.admin.read

Groups per page: 10 | 1-3 of 3 Groups

Login to a PKS (Revisit)

```
pks login -a pks.corp.local -u <user-in-ldap> --skip-ssl-validation
```

Login to pks as alana and create a cluster, this should be possible

```
pks login -a pks.corp.local -u alana -k
```

Login to pks as cody and create a cluster, should be possible but cody

```
pks login -a pks.corp.local -u cody -k
```

Login to pks as scott or naomi. The k8 developers will only be able to view clusters in the system but not do any admin tasks on the clusters.

NON EPMC DEPLOYMENT

In a non EPMC deployment each product on the installation requires to be separately configured to use LDAP.

PKS LDAP Integration

The following section covers LDAP integration on the PKS tile:

- Log into OpsMan and select the Enterprise PKS Tile
- Select the UAA section and configure the following

Note: Enable UAA as OIDC Provider

The screenshot shows the configuration interface for the Enterprise PKS Tile. On the left, there is a sidebar with various sections: Plan 4, Plan 5, Plan 6, Plan 7, Plan 8, Plan 9, Plan 10, Kubernetes Cloud Provider, Logging, Networking, UAA (which is highlighted with a grey bar), Monitoring, Usage Data, Errands, and Resource Config. The main area is titled "Configure your UAA user account store with either internal or external authentication mechanisms". It includes the following fields:

- Server URL:** ldap://controlcenter.corp.local
- LDAP Credentials:** CN=Administrator,CN=Users,DC=corp,DC=local (username) and a password field (password).
- User Search Base:** OU=Offices,OU=IT,DC=corp,DC=local
- User Search Filter:** sAMAccountName={0}
- Group Search Base:** OU=pk,OU=Infra,DC=corp,DC=local
- Group Search Filter:** member={0}
- Server SSL Cert:** A placeholder for a certificate file.

 At the bottom, there is a note: "Required only for ldaps://. This is the server certificate if using a Self Signed Certificate. If using a CA certificate, this must be the root certificate from your CA." and a link "Go to System in Control Panel to activate Windows."

Example LDAP settings provided to EPMC. Please update with your values and use during installation.

Description	Values
Current Decryption Password	<password>
Server URL	ldap://controlcenter.corp.local
LDAP Username	CN=Administrator,CN=Users,DC=corp,DC=local
LDAP Password	<password>
User Search Base	CN=Users,DC=corp,DC=local
User Search Filter	sAMAccountName={0}
Group Search Base	OU=pk,OU=Infra,DC=corp,DC=local

Group Search Filter	Member{0}
---------------------	-----------

Configuring Groups in UAA

A UAA admin user can assign the following UAA scopes to Enterprise PKS users

pks.clusters.manage: Accounts with this scope can create and access their own clusters.

pks.clusters.admin: Accounts with this scope can create and access all clusters.

pks.clusters.admin.read: Accounts with this scope can access any information about all clusters except for cluster credentials.

To grant Enterprise PKS access to an external LDAP group, perform the following steps:

1. Log in as the UAA admin using the procedure in [Log in as a UAA Admin](#).
2. To assign the `pks.clusters.manage` scope to all users in an LDAP group, run the following command:

```
uaac group map --name pks.clusters.manage GROUP-DISTINGUISHED-NAME
```

Where `GROUP-DISTINGUISHED-NAME` is the LDAP Distinguished Name (DN) for the group.

For example:

```
uaac group map --name pks.clusters.manage cn=pksclustermanage,ou=pks,ou=infra,dc=corp,dc=local
```

For more information about LDAP DNs, see the [LDAP DNs and RDNs](#) in the LDAP documentation.

3. To assign the `pks.clusters.admin` scope to all users in an LDAP group, run the following command:

```
uaac group map --name pks.clusters.admin GROUP-DISTINGUISHED-NAME
```

Where `GROUP-DISTINGUISHED-NAME` is the LDAP Distinguished Name (DN) for the group.

For example:

```
uaac group map --name pks.clusters.admin cn=pksclusteradmin,ou=pks,ou=infra,dc=corp,dc=local
```

4. To assign the `pks.clusters.read` scope to all users in an LDAP group, run the following command:

```
uaac group map --name pks.clusters.read GROUP-DISTINGUISHED-NAME
```

Where `GROUP-DISTINGUISHED-NAME` is the LDAP Distinguished Name (DN) for the group.

For example:

```
uaac group map --name pks.clusters.read cn=pksclusterreadonly,ou=pks,ou=infra,dc=corp,dc=local
```

OpsMan LDAP Integration

OpsMan LDAP integration requires an OpsMan RBAC group, which we had configured when setting up LDAP. Copy the DN to this group similar to what was done for the 'pksadmin' group.

OpsMan RBAC group

OpsMan Tile LDAP settings

Access the OpsMan configuration pane via Admin setting that can be found on the right corner

The screenshot shows the 'PCF Ops Manager' interface with the 'admin' user logged in. The top navigation bar includes links for 'INSTALLATION DASHBOARD', 'STEMCELL LIBRARY', and 'CHANGE LOG'. On the left, a sidebar lists various settings categories: 'Change Decryption Passphrase', 'SAML Settings', 'LDAP Settings' (which is currently selected and highlighted in grey), 'SSL Certificate', 'Pivotal Network Settings', 'Proxy Settings', 'Custom Banner', 'Export Installation Settings', 'Syslog', and 'Advanced Options'. The main content area is titled 'LDAP Settings' and contains the following fields:

- Current Decryption Passphrase***: A redacted password field.
- Server URL***: A text input field containing 'ldap://controlcenter.corp.local'.
- LDAP Username***: A text input field containing 'CN=Administrator,CN=Users,DC=corp,DC=local'.
- LDAP Password***: A redacted password field.
- User Search Base***: A text input field containing 'OU=Offices,OU=IT,DC=corp,DC=local'.
- User Search Filter**: A text input field containing 'sAMAccountName=[0]'.
- User Search Filter***: A text input field containing 'sAMAccountName=[0]'.
- Group Search Base***: A text input field containing 'OU=pks,OU=Infra,DC=corp,DC=local'.
- Group Search Filter***: A text input field containing 'member=[0]'.
- Group Max Search Depth (min: 1, max: 10)**: A text input field containing '10'.
- LDAP RBAC Admin Group Name***: A text input field containing 'CN=opsmanage,OU=pks,OU=Infra,DC=corp,'.
- Email Attribute***: A text input field containing 'mail'.
- LDAP Referrals***: A dropdown menu set to 'Follow any referrals automatically'.
- Server SSL Cert**: A file upload input field.

At the bottom of the form, there is a checked checkbox labeled 'Provision an Admin Client in the BOSH UAA' and a blue button labeled 'ENABLE LDAP AUTHENTICATION'.

Description	Values
Current Decryption Password	<password>
Server URL	ldap://controlcenter.corp.local
LDAP Username	CN=Administrator,CN=Users,DC=corp,DC=local
LDAP Password	<password>
User Search Base	OU=Offices,OU=IT,DC=corp,DC=local
User Search Filter	sAMAccountName={0}
Group Search Base	OU=pks,OU=Infra,DC=corp,DC=local
Group Search Filter	Member{0}
Group Max Search Depth	10
LDAP RBAC Admin Group	CN=opsmanage,OU=pks,OU=Infra,DC=corp,DC=local The group where the opsman users are present

Azure Active Directory as a SAML Identity Endpoint

Given below are the detailed steps of configuring single-sign-on (SSO) between Azure Active Directory and VMware Enterprise PKS

These steps are found [here](#)

Prerequisites

To configure Azure AD to designate Enterprise PKS as a service provider, you must have an Azure AD Global Administrator account.

Configure SAML in Azure AD

To configure Azure AD as a SAML identity provider for Enterprise PKS, do the following:

- Log in to Azure AD as a Global Administrator.
- Navigate to **Azure Active Directory**.
- Under **Create**, click **Enterprise application**.

Create



User



Guest user



Group



Enterprise application



App registration

- Under Add your own app, select Non-gallery application.
- Enter a Name and click Add.
- Navigate to Azure Active Directory > Enterprise applications.



Click your app and then click Single sign-on.



- Under Select a single sign-on method, select SAML.



SAML

Rich and secure authentication to applications using the SAML (Security Assertion Markup Language) protocol.

- Under Set up Single Sign-On with SAML, click the pencil icon for Basic SAML Configuration.

Set up Single Sign-On with SAML

Read the [configuration guide](#) for help integrating Test.

1

Basic SAML Configuration



Identifier (Entity ID)	Required
Reply URL (Assertion Consumer Service URL)	Required
Sign on URL	<i>Optional</i>
Relay State	<i>Optional</i>
Logout Url	<i>Optional</i>

Configure as per table below

Description	Values
Identifier (Entity ID)	Enter PKS-API:8443. For example: api.pks.example.com:8443
Reply URL	Enter https://PKS-API:8443/saml/SSO/alias/PKS-API:8443. For example: https://api.pks.example.com:8443/saml/SSO/alias/api.pks.example.com:8443
Sign on URL	Enter https://PKS-API:8443/saml/SSO/alias/PKS-API:8443. For example: https://api.pks.example.com:8443/saml/SSO/alias/api.pks.example.com:8443
Identifier (Entity ID)	Enter PKS-API:8443. For example: api.pks.example.com:8443
Reply URL	Enter https://PKS-API:8443/saml/SSO/alias/PKS-API:8443. For example: https://api.pks.example.com:8443/saml/SSO/alias/api.pks.example.com:8443
Sign on URL	Enter https://PKS-API:8443/saml/SSO/alias/PKS-API:8443. For example: https://api.pks.example.com:8443/saml/SSO/alias/api.pks.example.com:8443

Note: VMware recommends that you use the default settings for the fields that are not referenced in the above table.

- Click the pencil icon for User Attributes &

2 User Attributes & Claims

Givenname	user.givenname
Surname	user.surname
Emailaddress	user.mail
Name	user.userprincipalname
Unique User Identifier	user.userprincipalname

Claims

- Configure your user attributes and claims by doing the procedures in

How to: Customize claims issued in the SAML token for enterprise applications in the Microsoft Azure documentation. By default, Enterprise PKS uses the EmailAddress name identifier format.

The screenshot shows the 'User Attributes & Claims' section in the Microsoft Azure portal. On the left, there's a table for 'Required claim' with one row: 'Unique User Identifier (Name ID)' set to 'user.mail [nameid-formattemailAd...]' with a '...' button. Below it is a table for 'Additional claims' with several rows mapping schema names to user attributes like 'user.groups', 'user.mail', etc. On the right, the 'Group Claims (Preview)' pane is open, showing options for selecting groups ('None', 'All groups', 'Security groups', 'Directory roles', 'Groups assigned to the application') and setting the 'Source attribute' to 'Group ID'. Advanced options include customizing the group claim name to 'groups' and emitting groups as role claims.

As shown above, do the following:

- Click “Add a group claim”
- Choose the option for Group Claims as “Groups assigned to the application”
- If you chose “Source Attribute” as GroupID, then you will need to map GroupID(s) of the group(s) to the right objectID (as shown later).
- Click on “Advanced Options” and “Customize the name of the group claim” with the Name “groups”.
- Save.

- Under SAML Signing Certificate, copy and save the link address for App Federation Metadata Url or download Federation Metadata XML. You use the Azure AD metadata to configure SAML in the Enterprise PKS tile. For more information, see the Configure SAML as an Identity Provider section of Installing Enterprise PKS on

3

SAML Signing Certificate	
Status	Active
Thumbprint	41800B385144B1426B29D22B17BE35C443D3AA38
Expiration	7/22/2022, 3:16:44 PM
Notification Email	
App Federation Metadata Url	https://login.microsoftonline.com/c5e8ffb0-eef...
Certificate (Base64)	Download
Certificate (Raw)	Download
Federation Metadata XML	Download

Azure.

- If using Enterprise PKS Management Console or Opsman's PKS Title, use the following options to configure SAML Identity provider. Make sure that "External Group Atribute" is set to "groups" as configured in Azure. Once you save this, you will need to update PKS.

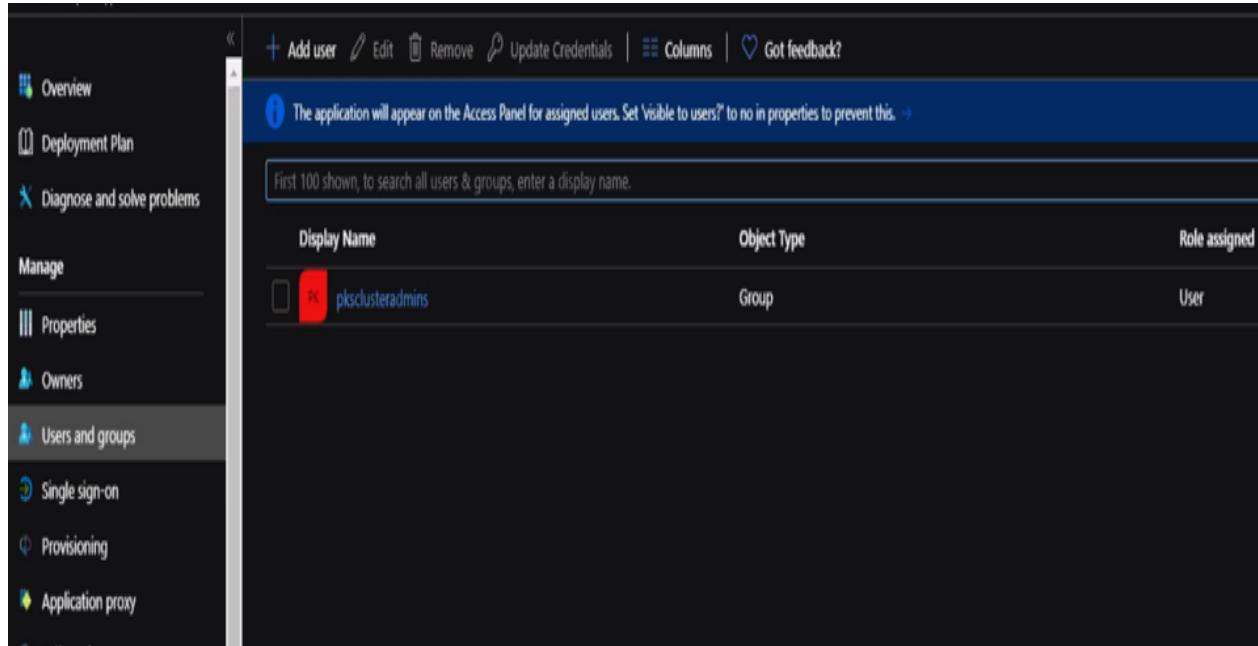
The screenshot shows the VMware Enterprise PKS Management Console interface. On the left, there is a navigation sidebar with sections like Administration, PKS Configuration, and PKS Instance Upgrade. The main content area has a breadcrumb navigation path: 2. Networking > 3. Identity. The title of the page is "Provide the identity management endpoint settings for Kubernetes clusters".

The configuration form for "Identity Management Source" is displayed. It includes the following fields:

- Provider Name:** azure
- Display Name:** Login with azure
- Provider Metadata (or use metadata URL):** A text input field containing XML metadata, with a preview pane below it showing the XML content.
- Provider Metadata URL:** https://login.microsoftonline...
- Name ID Format:** Email Address
- First Name Attribute:** givenname
- Last Name Attribute:** surname
- Email Attribute:** Optional
- External Groups Attribute:** groups
- Sign Authentication Requests:** (checkbox checked)
- Required Signed Assertions:** (checkbox checked)
- Signature Algorithm:** SHA256

- Configure your group attributes and claims by doing the procedures in the Configure group claims for SAML applications using SSO configuration section of Configure group claims for applications with Azure Active Directory (Public Preview) in the Microsoft Azure documentation.

In the example below, we're associating all PKS Admins to the AD group, "pksclusteradmins"



The screenshot shows the 'Users and groups' blade in the Microsoft Azure portal. The left sidebar has 'Manage' selected, with 'Users and groups' highlighted. The main area displays a table with one row:

Display Name	Object Type	Role assigned
pksclusteradmins	Group	User

Below are details of pksclusteradmins attributes. This includes the Object ID, that you will need to map in UAA to the appropriate group (see below).

Enterprise PKS Identity Management

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes 'Search resources, services, and docs (G+ /)'. Below it, the breadcrumb trail is 'Dashboard > Enterprise applications | All applications > Pivotal PKS | Users and groups > pksclusteradmins'. The main content area displays the 'pksclusteradmins' group details. On the left, a sidebar menu lists 'Overview', 'Diagnose and solve problems', 'Manage' (Properties, Members, Owners, Group memberships, Applications, Licenses, Azure role assignments), 'Activity' (Access reviews, Audit logs, Bulk operation results (Preview)), 'Troubleshooting + Support' (New support request), and 'Feedback' (Delete, Got feedback?). The main panel shows the group name 'pksclusteradmins' with a large red 'PK' icon. It includes fields for 'Membership type' (Assigned), 'Source' (Cloud), 'Type' (Security), 'Object Id' (4c68800b-6282-4e3d-8127-7fc8b8bbb96), and 'Creation date' (3/23/2020, 5:43:24 PM). Below these are sections for 'Direct members' (2 User(s), 0 Group(s), 0 Device(s), 0 Other(s)), 'Group memberships' (0), and 'Owners' (0).

UAA Scopes for Enterprise PKS Users

Now that SAML configuration is complete in PKS and Azure, we need to map the UAA scope to the right Azure AD Group. A UAA admin user can assign the following UAA scopes to Enterprise PKS users:

pks.clusters.manage: Accounts with this scope can create and access their own clusters.

pks.clusters.admin: Accounts with this scope can create and access all clusters.

pks.clusters.admin.read: Accounts with this scope can access any information about all clusters except for cluster credentials.

To grant Enterprise PKS access to a SAML group, perform the following steps:

5. Log in as the UAA admin using the procedure in [Log in as a UAA Admin](#).
6. Assign a PKS cluster scope to all users in a SAML group by running the following command:
 - `uaac group map --name UAA-SCOPE SAML-GROUP --origin SAML-ORIGIN`

Where:

- UAA-SCOPE is one of the UAA scopes described above
- SAML-GROUP is name of your SAML identity provider group. In our example it is the object ID for pksclusteradmins which is 4809484d-dc00-419a-83f4-f1301ae0765b
- SAML-ORIGIN is the domain name for your SAML identity provider. To find SAML-ORIGIN, click on the PKS tile, select Settings > UAA > SAML, and locate the Provider Name. In our example, it is azure (see above)

For example:

```
$ uaac group map --name pks.clusters.admin 4809484d-dc00-419a-83f4-f1301ae0765b --origin azure
```

Sign on using Enterprise PKS CLI

Now you can issue the following command to sign on using SAML Single Sign-on

```
pkcs login -a {pksapi.example.com} --sso
```

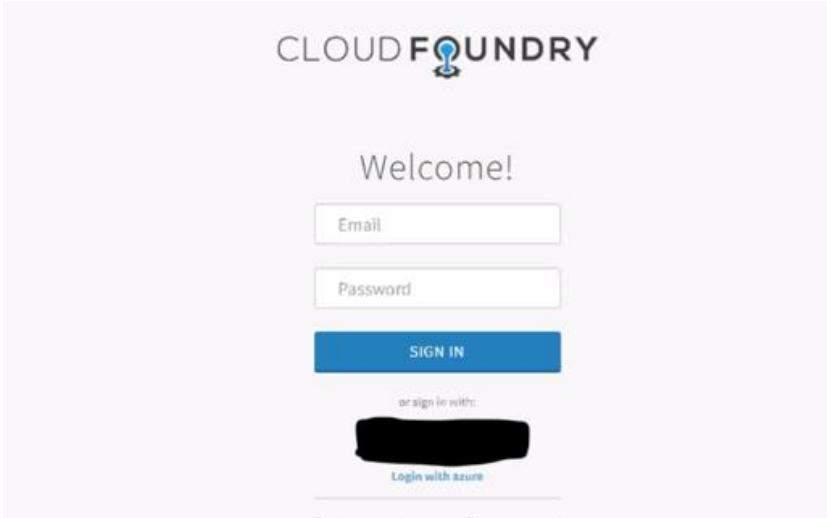
```
Examples:
  pkcs login -a <API> -u <USERNAME> -p <PASSWORD> [--ca-cert <PATH TO CERT> | -k]
  pkcs login -a <API> --client-name <CLIENT NAME> --client-secret <CLIENT SECRET> [--ca-cert <PATH TO CERT> | -k]

Flags:
  -a, --api string          The PKS API server URI
  --ca-cert string          Path to CA Cert for PKS API
  --client-name string      Client name
  --client-secret string    Client secret
  -h, --help                help for login
  -p, --password string     Password
  -K, --skip-ssl-validation Skip SSL Validation
  --skip-ssl-verification  Skip SSL Verification (DEPRECATED: use --skip-ssl-validation)
  --sso                      Prompt for a one-time passcode to do Single sign-on
  --sso-auto                 Auto launch local browser to do Single sign-on
  --sso-passcode string     Single sign-on with one-time passcode
  -u, --username string     Username

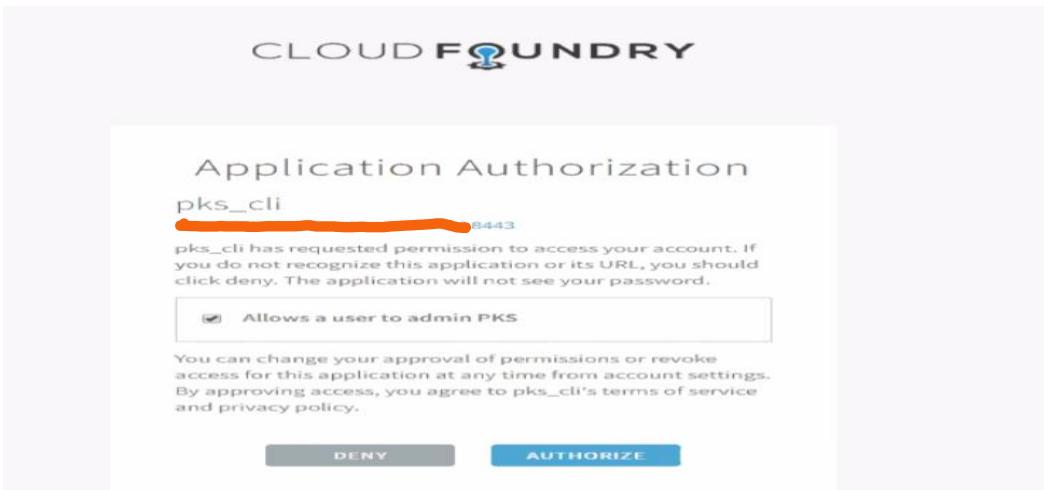
c:\>pkcs Pksapi.example.com gmt.local -k --sso
One Time Code ( Open A Web Browser to the following URL to get a Code: Pksapi.example.com gmt.local:8443/oauth/authorize?response_type=code&client_id=pks_cli&redirect_uri=Pksapi.example.com gmt.local:8443 )

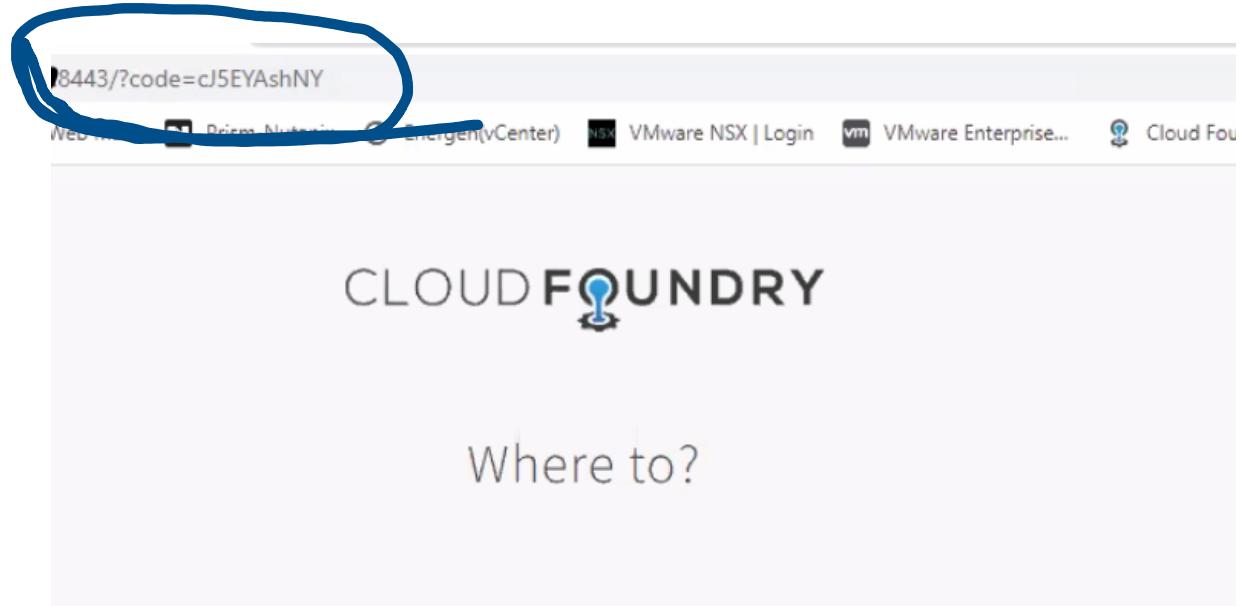
```

Use a web browser and copy the URL to get the code as following and click on “login with azure”



The first time you sign on you will get the following message:





Use the code above to sign on...

Troubleshooting:

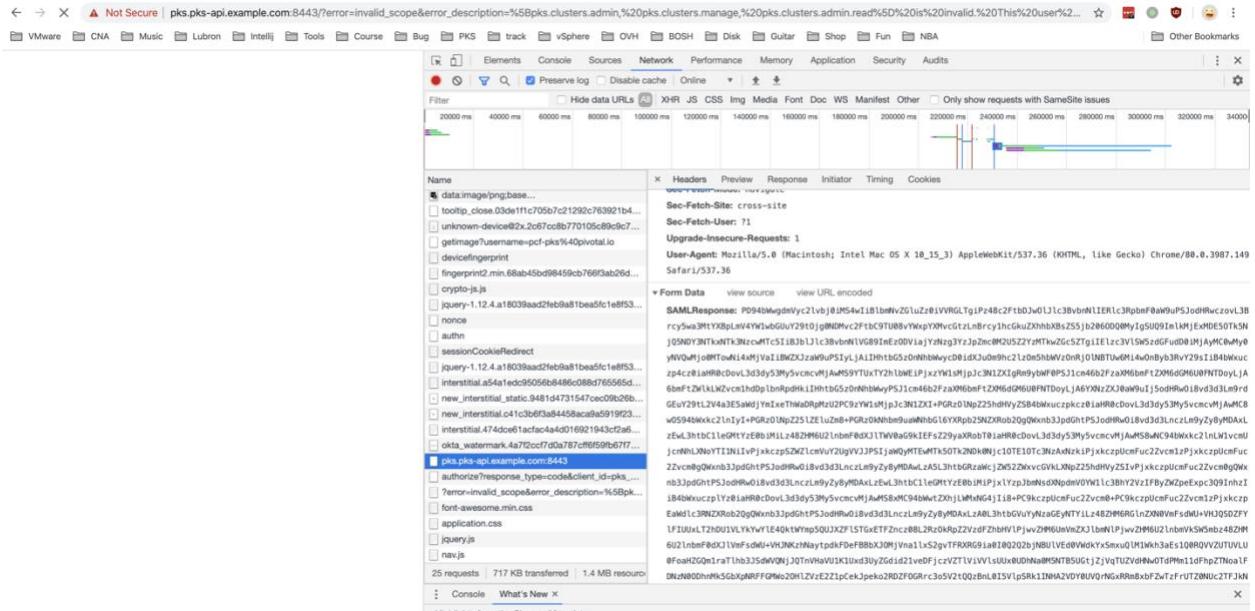
If you see the following error message, the mapping for UAA is not correct or what is coming back in the SAML token is not correct.

```
https://pksapi.example.com:8443/?error=invalid_scope&error_description=%5Bpks.clusters.admin,%20pks.clusters.admin.read,%20pks.clusters.manage%5D%20is%20invalid.%20This%20user%20is%20not%20allowed%20any%20of%20the%20requested%20scopes
```

```
error=invalid_scope
error_description=pks.clusters.admin,pks.clusters.admin.read,pks.clusters.manage
Userid is invalid. This user is not allowed any of the requested scopes
```

The following steps can help determine how to resolve this:

Sign on using Chrome. Before hitting enter, right click and click on ‘inspect’ on the page.



Copy the SAML Response and go to samltool.com/decode.php

Enterprise PKS Identity Management

The screenshot shows the onelogin SAML Developer Tools interface. The left sidebar contains various tools: X.509 CERTS, CODE/DECODE (Base64, Zip, URL Encode/Decode, Deflate + Base64 Encode, Base 64 Decode + Inflate), ENCRYPT / DECRYPT, SIGN, VALIDATE, ATTRIBUTE EXTRACTOR, XML PRETTY PRINT, BUILD METADATA, EXAMPLES, and EXTERNAL SAML TOOLS. The main content area is titled "Base64 Decode + Inflate". It has a sub-instruction: "Use this tool to base64 decode and inflate an Intercepted SAML Message. Paste a deflated base64 encoded SAML Message and obtain its plain-text version." Below this is a "CLEAR FORM FIELDS" button. A large text input box contains a long, base64-encoded SAML message. To the right of the input box is a "DECODE AND INFLATE XML" button. Below the input box is another section titled "Deflated XML" containing the same SAML message.

Here you can see the attributes that are being returned including the groups being returned and evaluated.

```
<Attribute  
Name="http://schemas.microsoft.com/ws/2008/06/identity/claims/groups"><AttributeValue>4c68800b-  
6282-4e3d-8127-7fcd8b8bbb96</AttributeValue></Attribute>
```

For example, In this case we noticed the groups attribute value is being returned as ObjectID instead of the Name of the group. Given this information, we were able to change the mapping to use the Object ID instead of Name.

Harbor Identity Management

Harbor supports the following identity management endpoints

- Harbor internal user management
- AD/LDAP
- UAA in Pivotal Container Service

Harbor AD/LDAP as an identity Management end point

Harbor Authentication mode needs to be enabled when installing PKS.

EPMC configuration, select Login harbor with LDAP users

The screenshot shows the VMware Enterprise PKS Management Console interface. The left sidebar has sections for Administration (Identity Management, Deployment Metadata, Configuration), PKS Configuration, PKS Instance Upgrade, and PKS Component Patch. The main area is titled 'PKS Configuration' and shows a 'WIZARD' tab is active. Step 8, 'Harbor', is currently selected. The configuration details for step 8 include:

- Harbor FQDN: harbor.corp.local
- Authentication: Log in Harbor with LDAP users (selected)
- HTTP Proxy: Optional
- HTTPS Proxy: Optional

Below the configuration fields, there is a note about Container Registry Storage Configuration and a Local File System option. At the bottom, there is a 'Resources' section.

If not using EPMC, change the Authentication mode to LDAP : Login harbor with LDAP users in the authentication section of the Harbor Registry tile within opsman.

Group definitions

This section defines the usergroups we will be creating within the different identity management endpoints to associate the different personas to

Group	Persona	Role
harboradmins	cody naomi	Harbor system administrator
harborusers	scott	Harbor developer

The groups are created under OU=pks,OU=Infra,DC=corp,DC=local

Harbor LDAP configuration

- Login to harbor as the admin user.
- Navigate to the Configuration section
- Select Authentication and the Authentication Mode as LDAP

The screenshot shows the Harbor web interface with the URL controlcenter.corp.local. The top navigation bar includes links for Home, Projects, Logs, Administration, Tasks, Configuration, and API Explorer. The Configuration section is selected. Under the Authentication tab, the Auth Mode is set to LDAP. The LDAP configuration details are as follows:

Setting	Value
LDAP URL	controlcenter.corp.local
LDAP Search DN	CN=Administrator,CN=Users,DC=corp,DC=local
LDAP Search Password	*****
LDAP Base DN	CN=Users,DC=corp,DC=local
LDAP Filter	(&(objectCategory=Person)(sAMAccountName
LDAP UID	sAMAccountName
LDAP Scope	Subtree
LDAP Group Base DN	OU=pks,OU=Infra,DC=corp,DC=local
LDAP Group Filter	
LDAP Group GID	cn
LDAP Group Admin DN	cn=harboradmins,ou=pks,ou=infra,dc=corp,dc=local
LDAP Group Membership	memberOf
LDAP Group Scope	Subtree

- Configure LDAP as per the below table

Description	Values	Description
LDAP URL	controlcenter.corp.local	LDAP hostname
LDAP Search DN	CN=Administrator,CN=Users,DC=corp,DC=local	Service account that has read rights to LDAP
LDAP Search Password	<password>	Service account password
LDAP Base DN	CN=Users,DC=corp,DC=local	DN where Users exist
LDAP Filter*	(&(objectCategory=Person)(sAMAccountName=*)((memberOf=CN=harboradmins,OU=pks,OU=Infra,DC=corp,DC=local)(memberOf=CN=harborusers,OU=pks,OU=Infra,DC=corp,DC=local)))	Add a filter that only queries users that belong to specific groups. Eg. harborusers and harboradmins
LDAP UID	sAMAccountName	Attribute name used to search user
LDAP Scope	Subtree	
LDAP Group Base DN	OU=pks,OU=Infra,DC=corp,DC=local	Base DN under which the harbor usergroups exist
LDAP Group Filter		Filter to look up specific LDAP groups
LDAP Group GID	cn	Attribute used in a search to match users , can be sAMAccountName
LDAP Group Admin DN	cn=harboradmins,ou=pks,ou=infra,dc=corp,dc=local	LDAP group that will have admin rights to Harbor
LDAP Group Membership	memberOf	All members of this group will have admin access to harbor
LDAP Group Scope	Subtree	

- Save and Test LDAP server
- Logout of harbor and try logging in as cody and scott

*for a full list of LDAP Filter go to <http://www.ldapexplorer.com/en/manual/109010000-ldap-filter-syntax.htm>

RBAC Authorization to K8 Clusters

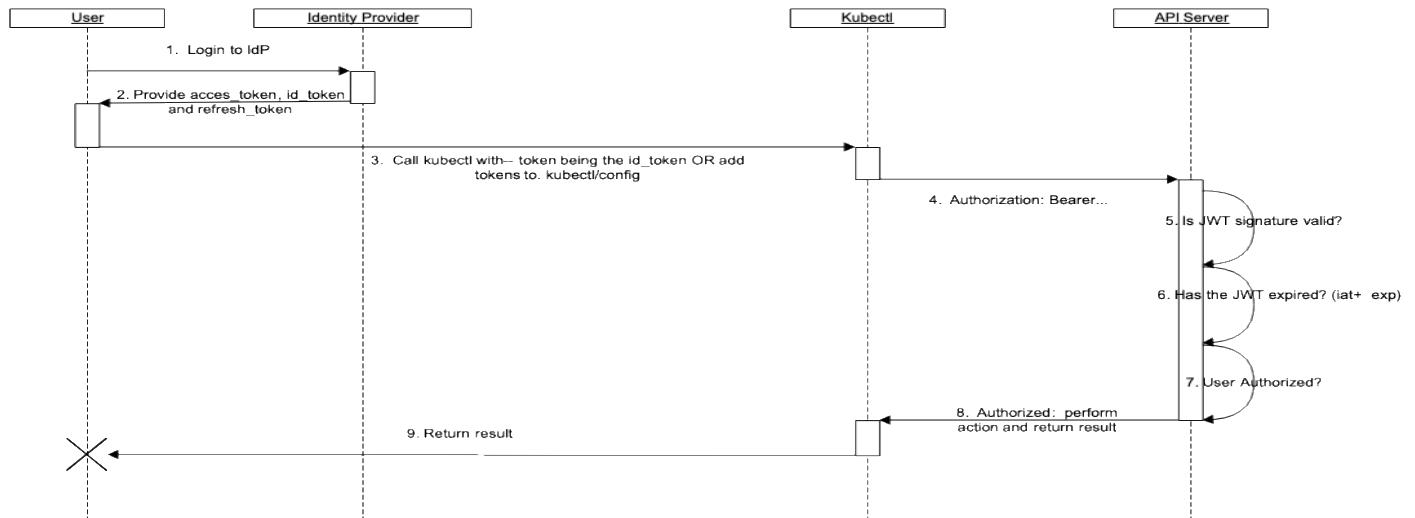
This section describes the steps to configure PKS to perform authentication against LDAP server so Developer can access their K8s clusters (globally or via namespace).

PKS UAA integration with LDAP (Windows Active Directory) in order to generate token-id (that's why the functionality is called UAA as OIDC provider).

This token-id then allows the developer to access K8s Cluster using kubectl.

PKS Control Plane VM is interacting with LDAP server here (K8s cluster don't interact with LDAP server at all)

NOTE: Make sure that UAA as an OIDC provider is enabled



Create Bindings

NOTE: Change the values as per the environment

- Login to the PKS API using the following command. Login as alana/cody who are pks api admins

```
pks login -a pks.corp.local -u cody -k
```

- Get credentials to the cluster

```
pks get-credentials my-cluster
```

- Create a namespace called 'dataengg' and run 5 replicas of nginx in it

```
kubectl create ns dataengg  
kubectl run nginx --image=nginx --replicas=5 --namespace=dataengg
```

Roles and Role Bindings

A Role contains rules that represent a set of permissions. Permissions are purely additive (there are no “deny” rules). A Role can be defined within a namespace with a Role, or cluster-wide with a ClusterRole.

A Role can only be used to grant access to resources within a single namespace.

A Role binding grants the permissions defined in a Role to a user or set of users. It holds a list of subjects (users, groups, or service accounts), and a reference to the Role being granted. Permissions can be granted within a namespace with a RoleBinding, or cluster-wide with a ClusterRoleBinding.

Cluster Roles

- [Create a cluster role](#) for certain resources with get, list and watch permissions

The following is an example of a cluster role for with permissions to all resources
PKS by default has a cluster-admin cluster role configured, this could be used instead.

Copy the contents below to a file cluster-all-role.yaml

```
kubectl apply -f cluster-all-role.yaml
```

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: cluster-all-admin
rules:
- apiGroups:
  - '*'
  resources:
  - '*'
  verbs:
  - '*'
- nonResourceURLs:
  - '*'
  verbs:
  - '*'
```

The following is an example of a cluster role for certain resources with get, list and watch permissions

Copy the contents below to a file cluster-reader-role.yaml

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: reader
rules:
- apiGroups: []
  resources: ["deployments", "configmaps", "pods", "secrets", "services"]
  verbs: ["get", "list", "watch"]
```

kubectl apply -f cluster-reader-role.yaml

Cluster Rolebinding

- Create a [cluster role binding](#) for k8 cluster administrators. The persona associated with this role is naomi.

The cluster role referred here is cluster-admin that exists in pks by default.

Copy the contents to a file admin-cluster-binding.yaml

```
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: dev-cluster-role-binding
subjects:
- kind: Group
  name: oidc:k8clusteradmins # Name is case sensitive
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: cluster-admin
  apiGroup: rbac.authorization.k8s.io
```

NOTE: We are assigning the group 'k8clusteradmins' that is present in our LDAP configuration.

- Logout of the pks using the pks cli

pks logout

```
ubuntu@cli-vm:~/roles$ pks logout
```

```
You have successfully logged out
```

- Retrieve the kube config for user

To obtain kubeconfig for a selected cluster and a user, run the following command:

```
pks get-kubeconfig <CLUSTER-NAME> -u <USER> -p <password> -a <PKS API Server> -k
```

Eg. `pks get-kubeconfig my-cluster -u naomi -a pks.corp.local -k`

Test OIDC based access for 'k8administrators'

- Retrieve the kubeconfig for persona naomi

```
pks get-kubeconfig my-cluster -u naomi -a pks.corp.local -k
```

- Set context

```
kubectl config set-context my-cluster --user=naomi  
kubectl get po --all-namespaces
```

All pods running in the cluster should be visible

```

riazm-a01:roles riazm$ kubectl config set-context riaz-cluster-sd --user=thor
Context "riaz-cluster-sd" modified.
riazm-a01:roles riazm$ kubectl get po --all-namespaces
NAMESPACE      NAME                                         READY   STATUS    RESTARTS   AGE
dataengg       nginx-7bb7cd8db5-8mpmc                   1/1     Running   0          14h
dataengg       nginx-7bb7cd8db5-fhjpc                   1/1     Running   0          14h
dataengg       nginx-7bb7cd8db5-g65lw                   1/1     Running   0          14h
dataengg       nginx-7bb7cd8db5-ltpc7                   1/1     Running   0          14h
dataengg       nginx-7bb7cd8db5-rs9sd                   1/1     Running   0          14h
kube-system   coredns-6f9bcd8956-7tzlz                  1/1     Running   0          16h
kube-system   coredns-6f9bcd8956-djcr2                  1/1     Running   0          16h
kube-system   coredns-6f9bcd8956-zjwbk                  1/1     Running   0          16h
kube-system   kubernetes-dashboard-5fc4ccc79f-9fzz6    1/1     Running   0          16h
kube-system   metrics-server-7f85c59675-xdrbx        1/1     Running   0          16h
pks-system    cert-generator-11b35c51b71ea3086396a780dbf20b5cd695b25d-rwxh7  0/1     Completed  0          16h
pks-system    event-controller-7b96987577-m58k5        2/2     Running   0          16h
pks-system    fluent-bit-684p                       2/2     Running   0          16h
pks-system    fluent-bit-129p6                      2/2     Running   0          16h
pks-system    fluent-bit-z4mzz                     2/2     Running   0          16h
pks-system    kube-state-metrics-65bcd5f775-bd4pv    2/2     Running   0          16h
pks-system    metric-controller-66b8b66498-jbrkk      1/1     Running   0          16h
pks-system    observability-manager-7dd6c4c6d-fw8ql    1/1     Running   0          16h
pks-system    sink-controller-5d76d8d546-cphp8        1/1     Running   0          16h
pks-system    telegraf-kf5g2                        1/1     Running   0          16h
pks-system    telegraf-ndj49                      1/1     Running   0          16h
pks-system    telegraf-tzlgn                      1/1     Running   0          16h
pks-system    telemetry-agent-58797bf64d-whrfm      2/2     Running   0          16h
pks-system    validator-847cb99cc-hpllj           1/1     Running   0          16h
pks-system    vrops-cadvisor-jvx29                 1/1     Running   0          16h
pks-system    vrops-cadvisor-m969s                 1/1     Running   0          16h
pks-system    vrops-cadvisor-wsf2h                 1/1     Running   0          16h
pks-system    waveform-collector-5c9fcfff8-k5cmw    1/1     Running   0          16h
pks-system    waveform-proxy-66d6445f6c-s258c       1/1     Running   0          16h

```

Create Roles

Create a Role on the namespace dataengg which has reader permissions for pods and pod-logs.(get and list permissions). Copy the contents to a file role-dataengg.yaml

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  namespace: dataengg
  name: dataengg-pod-and-pod-logs-reader
rules:
- apiGroups: [""]
  resources: ["pods", "pods/log"]
  verbs: ["get", "list"]
```

```
kubectl apply -f role-dataengg.yaml
```

Role binding for the Developer Role

- [Create a Role binding for the pksdev group](#) to access the Roles defined in [role-dataengg.yaml](#) for the namespace dataengg

Copy the contents to a file dataengg-role-binding.yaml

```
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: dataengg-pod-RoleBinding
  namespace: dataengg
subjects:
- kind: Group
  name: oidc:k8developers
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: dataengg-pod-and-pod-logs-reader
  apiGroup: rbac.authorization.k8s.io
```

NOTE: We are assigning the group ‘pksdev’ that is present in our LDAP configuration

```
kubectl apply -f dataengg-role-binding.yaml
```

Test OIDC based access for 'k8developers'

- Retrieve the kubeconfig for persona naomi

```
pks get-kubeconfig my-cluster -u naomi -a pks.corp.local -k
```

- Set context

```
kubectl config set-context my-cluster --user=scott
```

- List all pods running

```
kubectl get po --all-namespaces
```

This will result in an error since 'scott' is not an admin and has specific permissions to on namespace dataengg

Error from server (Forbidden): pods is forbidden:

```
riazm-a01:roles riazm$ kubectl get po --all-namespaces
Error from server (Forbidden): pods is forbidden: User "oidc:visiondev" cannot list resource "pods" in API group "" at the cluster scope
```

- List pods running in the dataengg namespace

```
kubectl get po --namespace dataengg
```

```
riazm-a01:roles riazm$ kubectl get po --namespace dataengg
NAME           READY   STATUS    RESTARTS   AGE
nginx-7bb7cd8db5-8mpmc  1/1     Running   0          14h
nginx-7bb7cd8db5-fhjpc  1/1     Running   0          14h
nginx-7bb7cd8db5-g65lw  1/1     Running   0          14h
nginx-7bb7cd8db5-ltpc7  1/1     Running   0          14h
nginx-7bb7cd8db5-rs9sd  1/1     Running   0          14h
```

KubeConfig

SOLUTION 1: (Recommended) Retrieve kubeconfig file

In the LDAP configuration we have ‘visiondev’ assigned to the ‘pksdev’ group with a password <password>

To obtain kubeconfig for a selected cluster and a user, run the following command:

```
pks get-kubeconfig <CLUSTER-NAME> -u <USER> -p <password> -a <PKS API Server> -k
```

NOTE: -k option used to avoid “missing certificate” warning

E.G. `pks get-kubeconfig my-cluster -u naoimi -p <password> -a pks.corp.local -k`

This will generate kubeconfig file which can be distributed to Developer to access the cluster.

Admin users can review user related context in the kubeconfig file using **kubectl config get-contexts** command like shown below:

```
kubectl config get-contexts
CURRENT      NAME                               CLUSTER                         AUTHINFO
NAMESPACE
-----
riaz-cluster-vra8_1          riaz-cluster-vra8_1           0dfc190f-0a6e-45c8-afde-
9dbd098d4deb
raghu-cluster-ace1           raghu-cluster-ace1            87930898-554f-
49fb-a689-0d6148b6fc51
raghu-cluster-vra8           raghu-cluster-vra8            34ad0f04-5e2d-
4bafe-902b-11df2d7c10ca
*   riaz-cluster-sd             riaz-cluster-sd             visiondev

Currently active context is marked with '*'
```

END SOLUTION 1

SOLUTION 2:

Retrieve token for Developer

In the LDAP configuration we have visiondev assigned to the pksdev group with a password <password>

Copy the contents to a file getToken.sh

```
curl 'https://pks-ace-api.eng.vmware.com:8443/oauth/token' -k -XPOST -H 'Accept: application/json' -d "client_id=pks_cluster_client&client_secret=""&grant_type=password&username=visiondev&password=<password>&response_type=id_token"
```

Run getToken.sh

[./getToken.sh](#)

```
This will result in both a id token and a refresh token
```

```
E.g.  
{"access_token":"eyJhbGciOiJSUzI1NiIsImprdBzOj8vcGtzLWFjZS1hcGkuZW5nLnZtd2FyZS5jb  
.....  
cn6Bxg","expires_in":599,"scope":"openid roles","jti":"a7c42e36061b44ab809920016237f3
```

Create kubeconfig File

[Create the kube config file](#) or Update an existing one with the id token and refresh token.
(visiondev). If visiondev does not exist in your existing config file, add a new entry.

Note : config file can be found under `~/.kube/config`

If it does not exist create one

```

apiVersion: v1
clusters:
- cluster:
  certificate-authority-data: Ci0tLS0tQkVHSU4gQ0VSVE1GSUNBVEUtLS0tLQpNSU1DK3pDQ0F1T2dBd01CQWdJVWEvZjdGWWVUS1VFTE9
...
uEU0MTlkSG9YUH1Ja3NiUWNjMEVIMVI4dm5tUzVsYmZZdCtBREE9Ci0tLS0tRU5EiENFU1RJRK1DQRFLS0tLS0K
  server: https://my-cluster.corp.local:8443
  name: my-cluster
- name: thor
  user:
    auth-provider:
      config:
        client-id: pks_cluster_client
        client-secret: ""
        id-token:
eyJhbGciOiJSUzI1NiIsImprdsI6Imh0dHBz0i8vcGtzLWFjZS1hcGkuZW5nLnZtd2FyZS5jb2060DQ0My90b2t1b19rZXlziwia2lkIjoia2V5LTEiLCJ
0eXAiOijKV1QifQ.
.....
of_0bK2-RJN9LeGsnnuBgD8qfZfUZoiV2EkySf20nEhXXSSB8JiH4oDvhEPA
  idp-certificate-authority-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURqRENDQW5T0F3SUJBZ01WQ
....
hdC8vMk41UXg0dGkvWE4rSE4vUDMKLS0tLS1FTkQgQ0VSVE1GSUNBVEUtLS0tLQo=
  idp-issuer-url: https://pks-ace-api.eng.vmware.com:8443/oauth/token
  refresh-token: eyJhbGciOiJSUzI1NiIsImprdsI6Imh0dHBz0i8vcGtzLWFjZS1hcGkuZW5nLnZtd2FyZS5jb2
....
u0zlQBv6Dm2fw-MMP3IDBKuuA
  name: oidc
- name: visiondev
  user:
    auth-provider:
      config:
        client-id: pks_cluster_client
        client-secret: ""
        id-token:
eyJhbGciOiJSUzI1NiIsImprdsI6Imh0dHBz0i8vcGtzLWFjZS1hcGkuZW5nLnZtd2FyZS5jb2060DQ0My90b2t1b19rZXlziwia2lkIjoia2V5LTEiLCJ
0eXAiOijKV1QifQ.
.....
zu107ERplGSp4zw6syn52439L700b4ZjnK7c8VvOXNuw9Q5NT2GAJou3tNBgfQjlxUAglLxx7H0Rg
  idp-certificate-authority-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURqRENDQW5T0F3SUJBZ01WQ
....
THBuTFhXChN0VnAxVVI3UkZqS3NoOUJ4VnhMRldJanV3Zj1NdGhMQ210ZEJnd3Q2aFZhdC8vMk41UXg0dGkvWE4rSE4vUDMKLS0tLS1FTkQgQ0VSVE1GSUN
BVEUtLS0tLQo=
  idp-issuer-url: https://pks-ace-api.eng.vmware.com:8443/oauth/token
  refresh-token:
eyJhbGciOiJSUzI1NiIsImprdsI6Imh0dHBz0i8vcGtzLWFjZS1hcGkuZW5nLnZtd2FyZS5jb2060DQ0My90b2t1b19rZXlziwia2lkIjoia2V5LTEiLCJ
0eXAiOijKV1QifQ.
.....
_0e2LzMVT04uF9V11vwT74jjssmpXo1MsdJvWs50g9qMVKouvsBIHPY15xWVh0GCFkTvaqKamr2qLZg2mHz53uzrpzB18Vc
zd3BKDO6ysDM_Wni_JjoNZKHHI6xI-NTe8Xxcyy2n9a7d0lpr7Ai6V-im0sfy8fdvh5RKwuq-
  name: oidc

```

END SOLUTION 2



VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com.

Copyright © 2019 VMware, Inc. All rights reserved. This product is protected by U.S. and international copyright and intellectual property laws. VMware products are covered by one or more patents listed at vmware.com/go/patents. VMware is a registered trademark or trademark of VMware, Inc. and its subsidiaries in the United States and other jurisdictions. All other marks and names mentioned herein may be trademarks of their respective companies. Item No: vmw-wp-tech-temp-word-102-proof 5/19