Identity and Access Management

Level 200

KD Singh
Changbin Gong
Oracle Cloud Infrastructure
October 2019

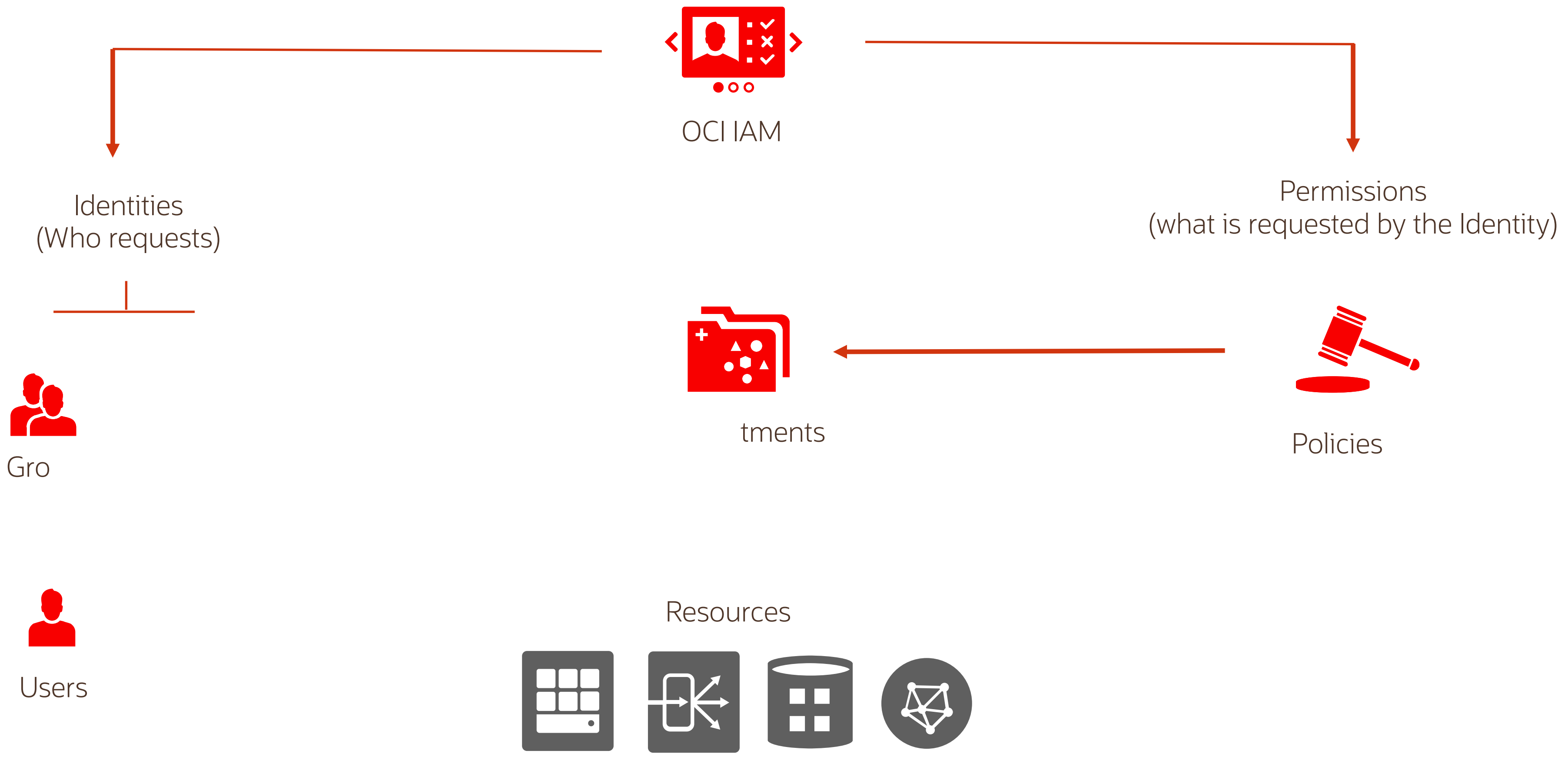
Safe harbor statement

Objectives

After completing this lesson, you should be able to:

- Create Instance Principals
- Understand Multi-Factor Authentication (MFA)
- Write advanced Policies
- Federate OCI with Oracle Identity Cloud Service (IDCS)
- Federate OCI with Microsoft Active Directory
- Federate OCI with Microsoft Azure Active Directory
- Design reference IAM Model for an Enterprise
- Real life story for IAM compartment and policy design

Basics of Identity and Access Management



Part I: Instance Principals and Dynamic Groups

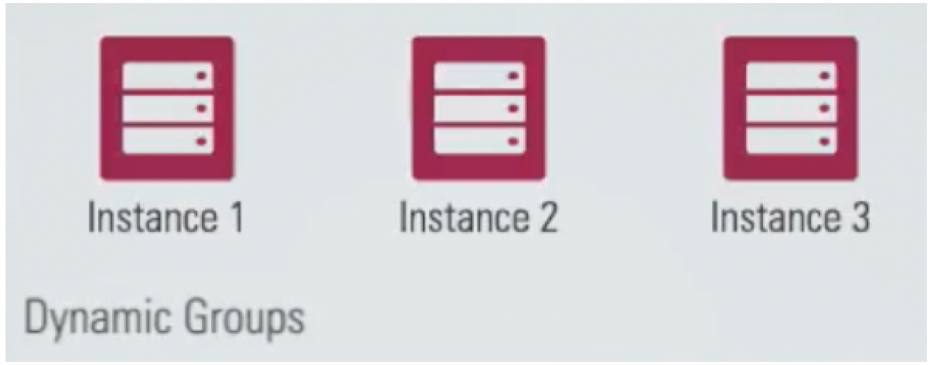
Instance Principals

Instance Principals lets instances (and applications) to make API calls against other OCI services removing the need to configure user credentials or a configuration file

- Current problem
  - Storing API credentials on each instance
  - Credential rotation
  - Audits at instance level are impossible since credentials are same across hosts
- How does Instance principals solve the problem?
  - Instance Principals gives instances their own identity, instances become a new type of Principal (in addition to OCI IAM users/groups)
  - Dynamic groups allows policy to be defined on instances
  - In the Audit, you will see the instance Id making the API call

Steps to create an Instance Principal

01: Create a Dynamic Group that matches a set of instances

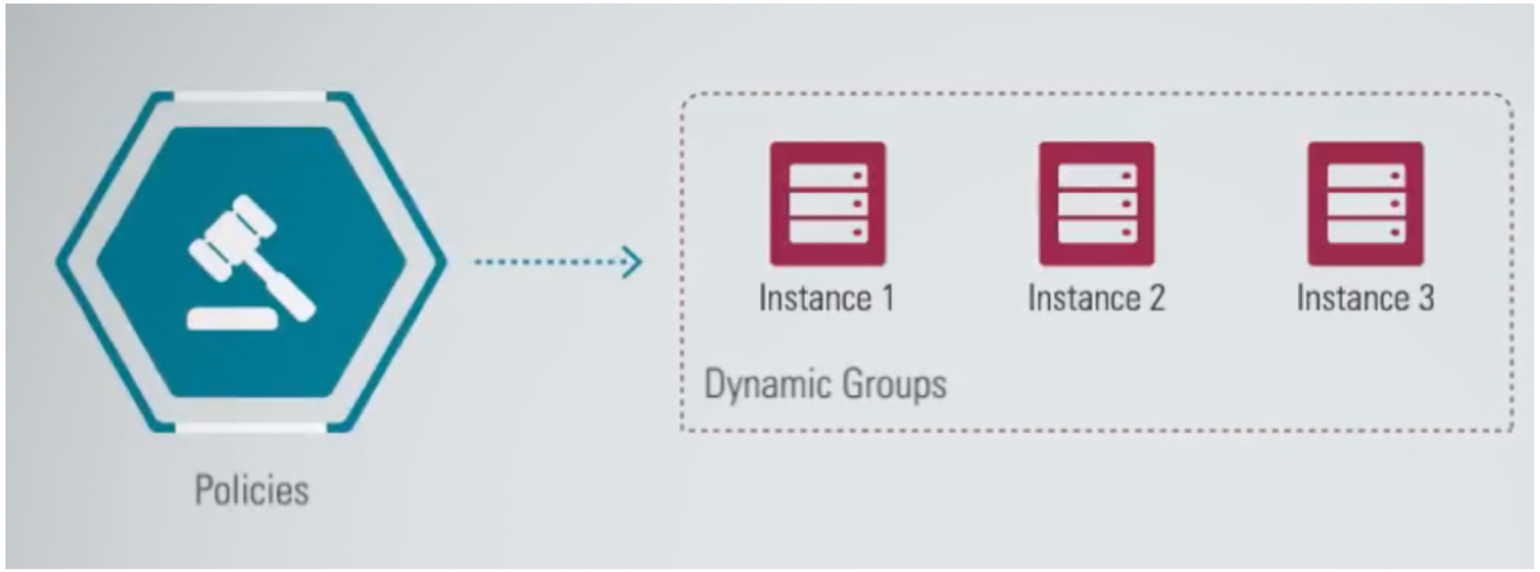


Possible to exclude specific instances from a Dynamic Group

```
All {instance.compartment.id = '<compartment_ocid>',
instance.id != '<instance1_to_exclude_ocid>',
instance.id != '<instance2_to_exclude_ocid>'}
```

Steps to create an Instance Principal

02 - Create a Policy dictating what permissions those instances should receive

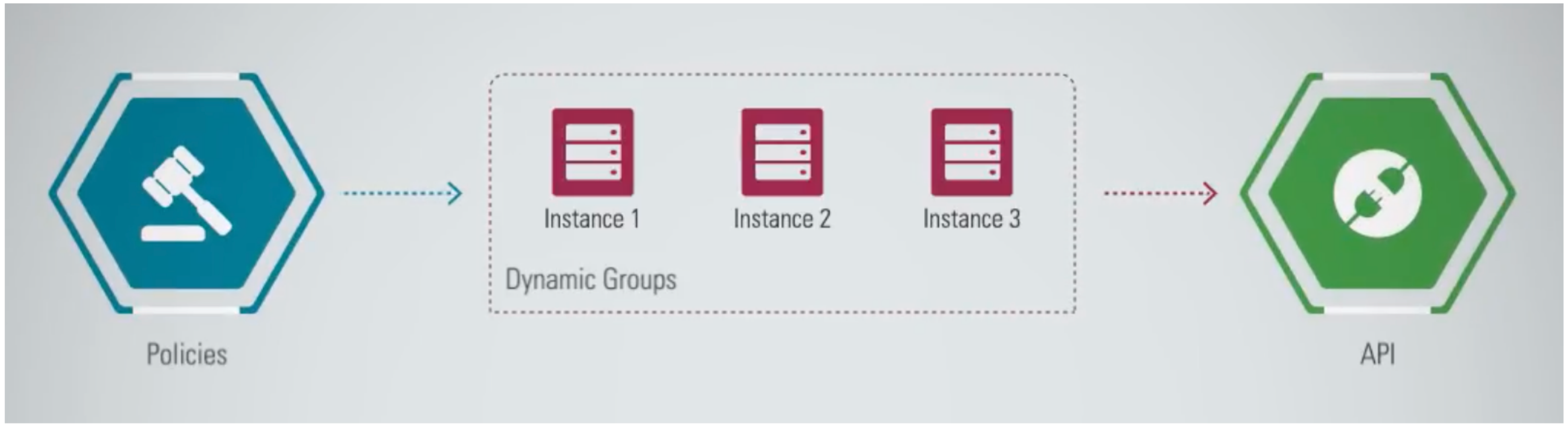


```
Allow dynamic-group <group-name> to manage buckets in tenancy

Allow dynamic-group <group-name> to manage objects in tenancy
```

Steps to create an Instance Principal

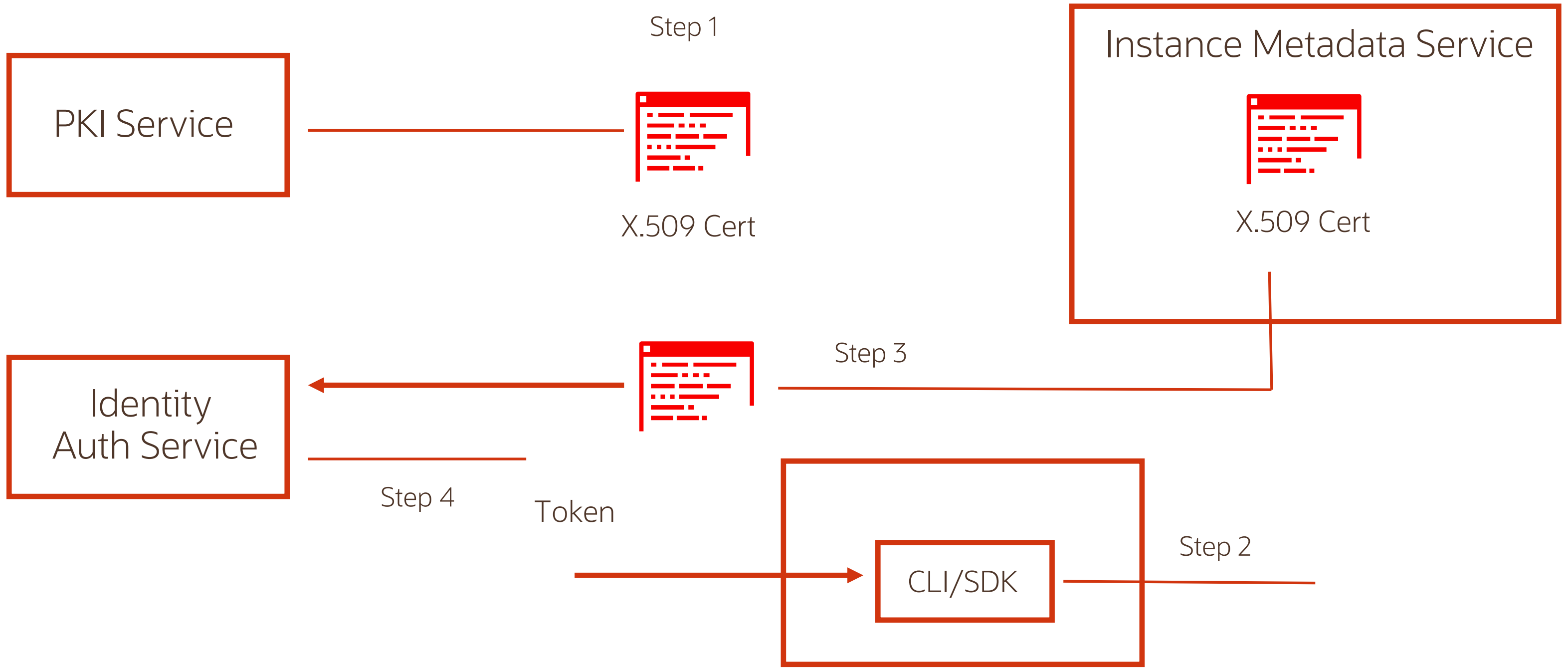
03: Customer deploys code to an instance. For example, OCI SDK/CLI is able to make calls to OCI APIs without customer configured credentials



```
[opc@webserver1 .oci]$ oci os ns get --auth instance_principal
{
  "data": ”mydata"
```

Java and Python SDKs and Terraform also support Instance Principal authorization

How it works?


PKI Service
Step 1
X.509 Cert
Instance Metadata Service
X.509 Cert
Identity Auth Service
Step 3
Step 4
Token
CLI/SDK
Step 2

How it works?

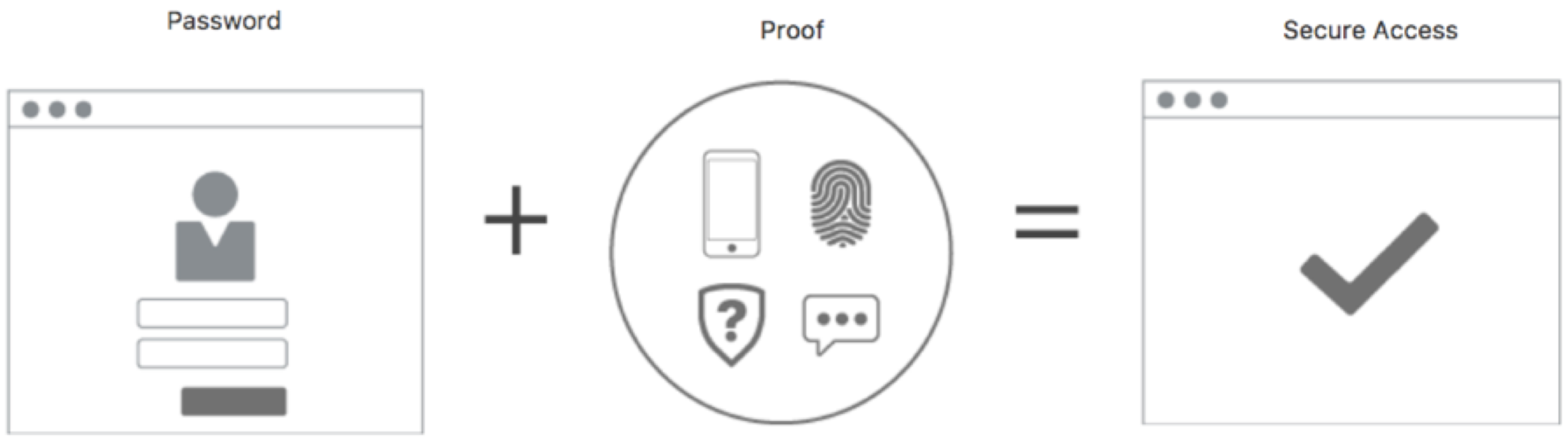
- The certificate is rotated multiple times a day and customers cannot change the frequency
- You can use this Curl command to query the X.509 certificates, curl http://169.254.169.254/opc/v1/identity/cert.pem

```
[opc@webserver1 .oci]$ curl
http://169.254.169.254/opc/v1/identity/cert.pem
-----BEGIN CERTIFICATE-----
MIIIPjCCBiagAwIBAgIQesV+WyeYgLqUxb4vSgrL/jANBgkqhkiG9w0BAQsFADCB
qTFzMHEGA1UECxNqb3BjLWRldmljZTo1NDo1Yjo4NTpiOTowMjo5Yjo4YTo4MDpl
YTo1MjoxNzo1MjozYjo1ZjowZjpmMzo1MTpkNjo1YzoxZjpmYTozYTo1MTo4OTow
ZDpjMTowNTo0MjphOTowYzplMTo4YjEyMDAGA1UEAxMpUEtJU1ZDIElkZW50aXR5
IEludGVybWVkaWF0ZSB1cy1hc2hidXJuLTEwHhcNMTgwNjE1MTc0MjU1WhcNMTgw
NjE1MTg0MjU1WjCCAbQxggFSMBwGA1UECxMVb3BjLWNlcnR0eXBlOmluc3RhbmNl
MGcGA1UECxNgb3BjLWluc3RhbmNlOm9jaWQxLmluc3RhbmNlLm9jMS5pYWQuYWJ1
d2NsanRrYWMyMjZzbDY1N3hsbHIzNWszaGozYWJra3I3dm9sd3BndWd6c3Nkdjd2
```

ORACLE

Part II: Multifactor Authentication (MFA)

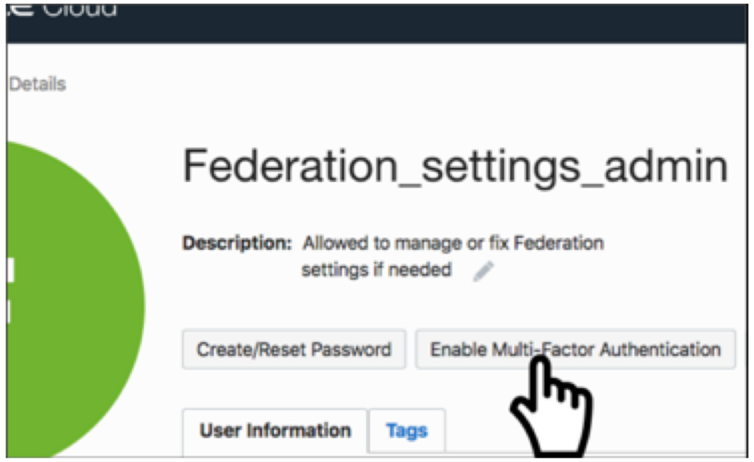
Multi-factor Authentication (MFA) - General Concepts



Multi-factor authentication (MFA) is a method of authentication that requires the use of more than one factor to verify a user's identity. Examples of authentication factors are a password (something you know) and a device (something you have).

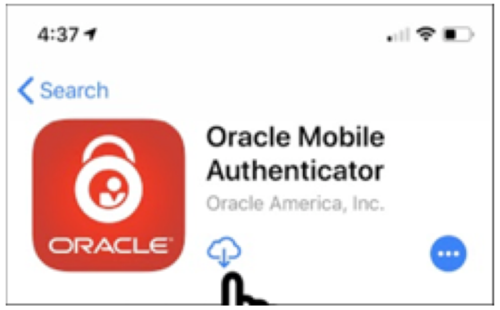
Multi-factor Authentication (MFA) - How to enable?

Step 1

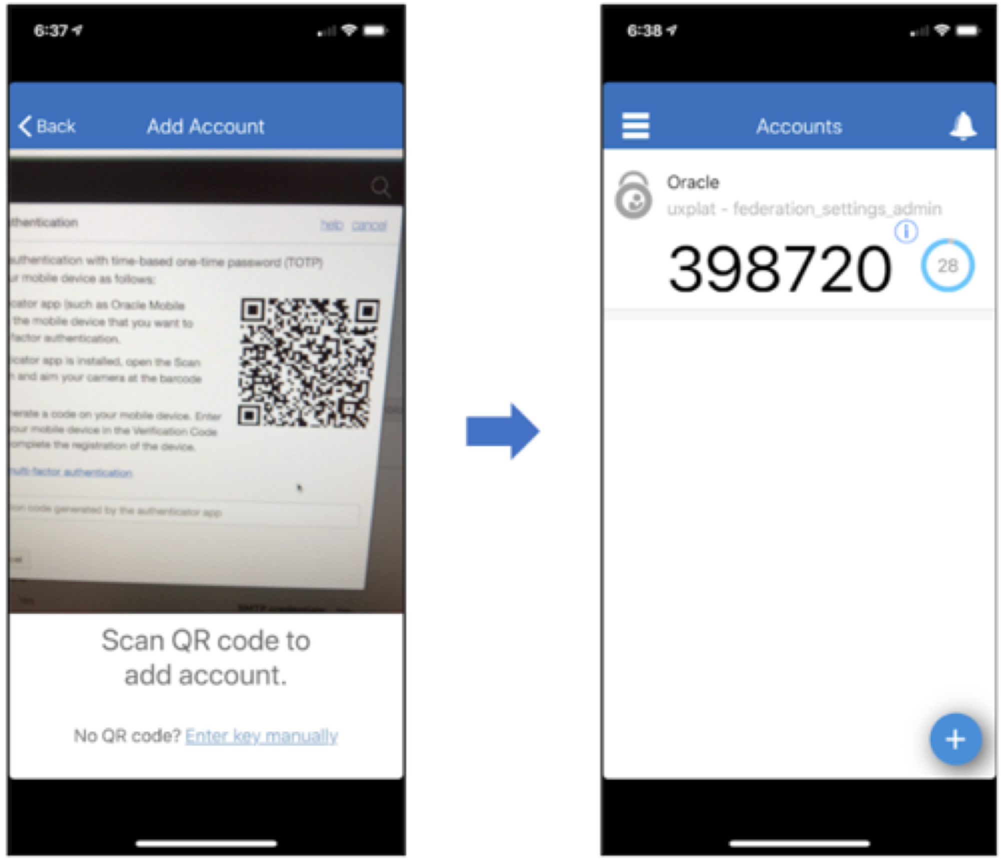


Description: Allowed to manage or fix Federation settings if needed

Step 2



Step 3


6:37
Back
Add Account
Scan QR code to add account.
No QR code? Enter key manually
6:38
Accounts
Oracle
uxplat - federation_settings_admin
398720
28

Part III: Advanced Policies

Policy Syntax

Allow **<subject>** to **<verb>** **<resource-type>** in **<location>** where <conditions>

Verb	Type of access
inspect	Ability to list resources
read	Includes inspect + ability to get user-specified metadata/actual resource
use	Includes read + ability to work with existing resources (the actions vary by resource type)*
manage	Includes all permissions for the resource

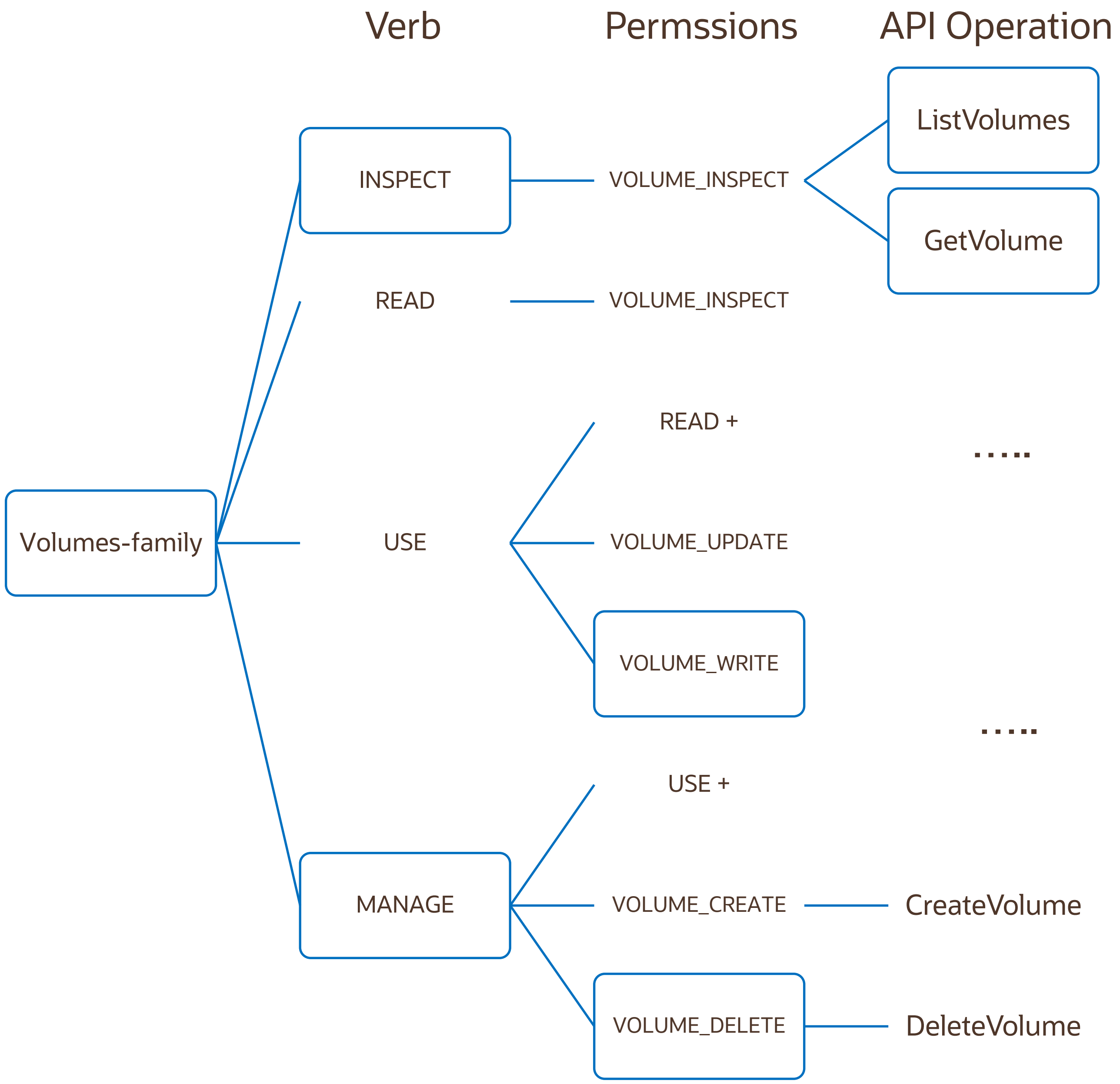
* In general, this verb does not include the ability to create or delete that type of resource

Aggregate resource-type	Individual resource type
all-resources	
database-family	db-systems, db-nodes, db-homes, databases
instance-family	instances, instance-images, volume-attachments, console-histories
object-family	buckets, objects
virtual-network-family	vcn, subnet, route-tables, security-lists, dhcp-options, and many more resources (link)
volume-family	Volumes, volume-attachments, volume-backups

The IAM Service has no family resource-type, only individual ones; Audit and Load Balancer have individual resources (load-balancer, audit-events)

Verbs & Permissions

- When you write a policy giving a group access to a particular verb and resource-type, you're actually giving that group access to one or more predefined permissions
- Permissions are the atomic units of authorization that control a user's ability to perform operations on resources
- As you go from inspect > read > use > manage, the level of access generally increases, and the permissions granted are cumulative
- Each API operation requires the caller to have access to one or more permissions. E.g., to use ListVolumes or GetVolume, you must have access to a single permission: VOLUME_INSPECT



Policy Syntax

Allow <subject> to <verb> <resource-type> in <location> where **<conditions>**

Conditions:
Syntax for a single condition: variable =|!= value

- 2 variable types: request (relevant to the request itself), and target (relevant to the resource(s) being acted upon in the request)
- E.g. variable request.operation represents the API operation being requested (e.g. ListUsers); target.group.name represents the name of the group
- variable name is prefixed accordingly with either request or target followed by a period

request.operation	The API operation name being requested
request.permission	The underlying permission(s) requested
request.user.id	OCID of the requesting user
request.groups.id	The OCIDs of groups requesting user is in
target.compartment.id	The OCID of the compartment
target.compartment.name	The name of the compartment specified in target.compartment.id
request.region	The key of the region the request is made in
request.ad	The name of the AD the request is made in

Example: `Allow group Phoenix-Admins to manage all-resources in tenancy where request.region='phx'`

Policy Syntax

Allow <subject> to <verb> <resource-type> in **<location>** where **<conditions>**

- Conditions: Syntax for a single condition: variable =|!= value

Type	Types of value
String	(single quotation marks are required around the value)
Pattern	/HR*/ (matches strings that start with "HR") /*HR/ (matches strings that end with "HR") /*HR*/ (matches strings with "HR")

- Syntax for multiple conditions: any|all {<condition>,<condition>,...}

Allow group XYZ to manage groups in tenancy

where any {request.operation='ListGroups',
request.operation='GetGroup',
request.operation='CreateGroup',
request.operation='UpdateGroup'}

Allow group XYZ to manage groups in tenancy

where all {request.permission='GROUP_INSPECT',
request.operation='ListGroups'}

Advanced Policy

- Policy for GroupAdmins group to manage any groups with names that start with "A-Users-"
  - `Allow group GroupAdmins to manage groups in tenancy where target.group.name = /A-Users-*/`

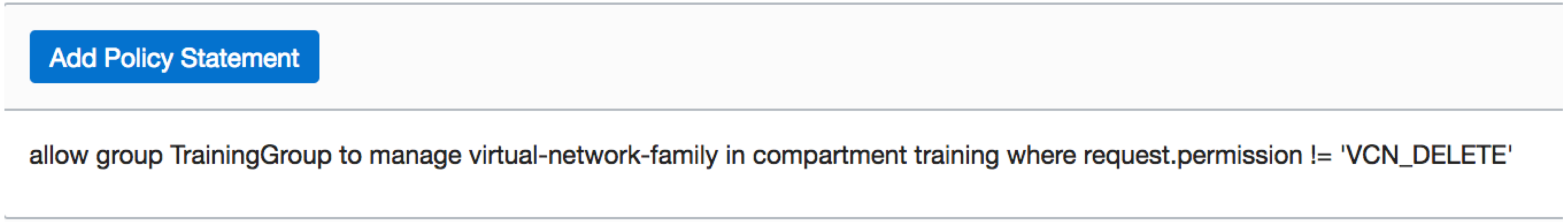
- Policy for GroupAdmins group to manage the membership of any group besides the Administrators group:
  - `Allow group GroupAdmins to use users in tenancy where target.group.name != 'Administrators'`

- Policy lets A-Admins create, update, or delete any groups whose names start with "A-", except for the A-Admins group itself
  - `Allow group GroupAdmins to manage groups in tenancy where all {target.group.name=/A-*/,target.group.name!='A-Admins'}`

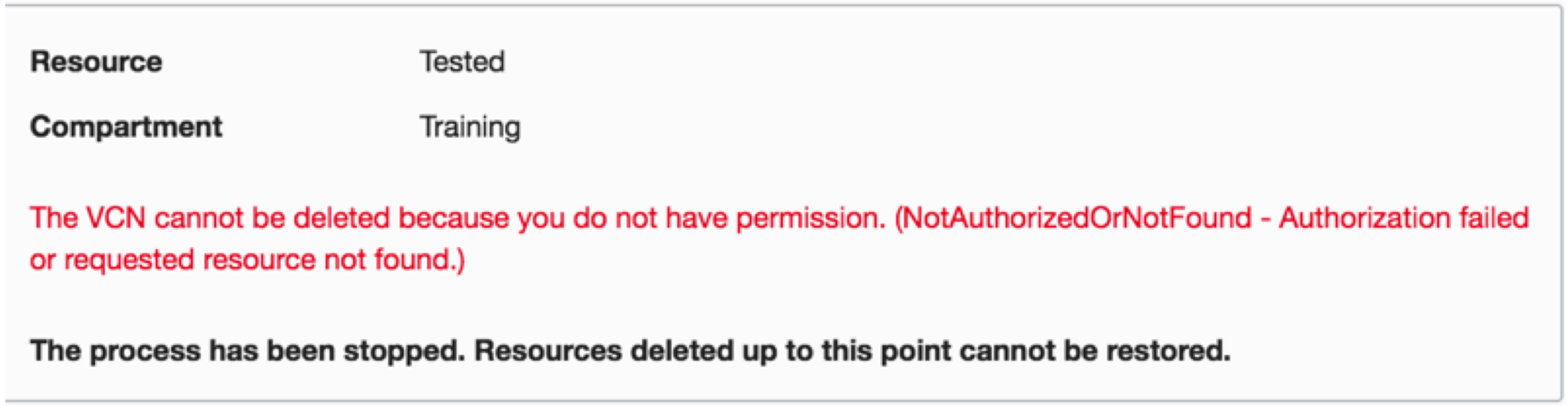
Scoping Access with Permissions or API Operations

- In a policy statement, you can use conditions combined with permissions or API operations to reduce the scope of access granted by a particular verb.
- Allow a user to manage VCN resources except have the ability to delete a VCN

  allow group TrainingGroup to manage virtual-network-family in compartment training
  where request.permission != 'VCN_DELETE'



allow group TrainingGroup to manage virtual-network-family in compartment training where request.permission != 'VCN_DELETE'



Resource	Tested
Compartment	Training

The VCN cannot be deleted because you do not have permission. (NotAuthorizedOrNotFound - Authorization failed or requested resource not found.)

The process has been stopped. Resources deleted up to this point cannot be restored.

Part IV: Identity Federation

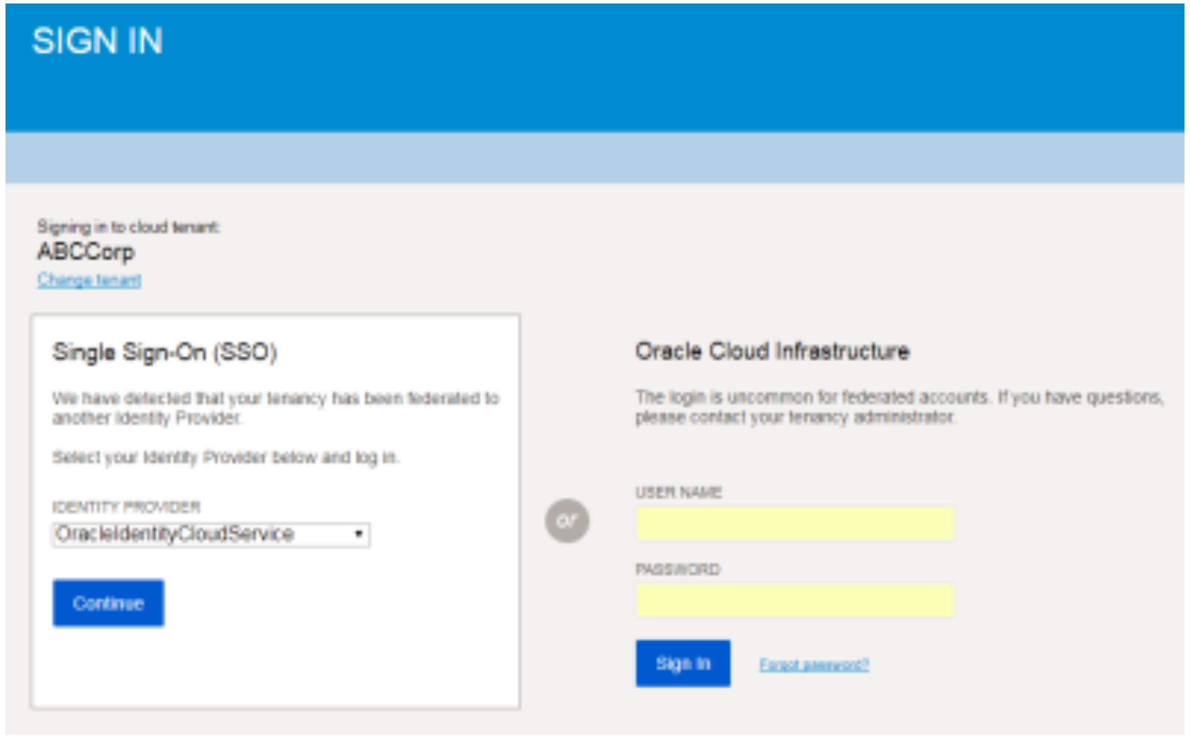
Best Practices for securing IAM – IAM Federation

Oracle recommends that you use federation to manage logins into the Console

- An administrator needs to set up a federation trust between the on-premises identity provider (IdP) and IAM, in addition to creating mapping between on-premises groups and IAM groups
- Federation is especially important for enterprises using custom policies for user authentication (for example, multifactor authentication).
- When using federation, Oracle recommends that you create a federation administrators group that maps to the federated IdP administrator group
- The federation administrators group will have administrative privileges to manage customer tenancy, while being governed by the same security policies as the federated IdP administrator group
- In this scenario, it is a good idea to have access to the local tenancy administrator user (that is, member of the default tenancy administrator IAM group), to handle any break-glass type scenarios (for example, inability to access resources through federation)
- Tenancies federated with Oracle Identity Cloud Service or Okta, can leverage SCIM (System for Cross-domain Identity Management). Such users can have the additional credentials such as API keys and auth tokens that are managed in the User Settings page.

Experience for Federated Users

- Federated users can use the Console to access Oracle Cloud Infrastructure (according to IAM policies for the groups the users are in).
- They'll be prompted to enter their Oracle Cloud Infrastructure tenant (for example, ABCCorp).
- They then see a page with two sets of sign-in instructions: one for federated users and one for non-federated (Oracle Cloud Infrastructure) users.


SIGN IN

Signing in to cloud tenant:
ABCCorp
Change tenant

Single Sign-On (SSO)

We have detected that your tenancy has been federated to another Identity Provider.

Select your Identity Provider below and log in.

IDENTITY PROVIDER
OracleIdentityCloudService

Continue

or

Oracle Cloud Infrastructure

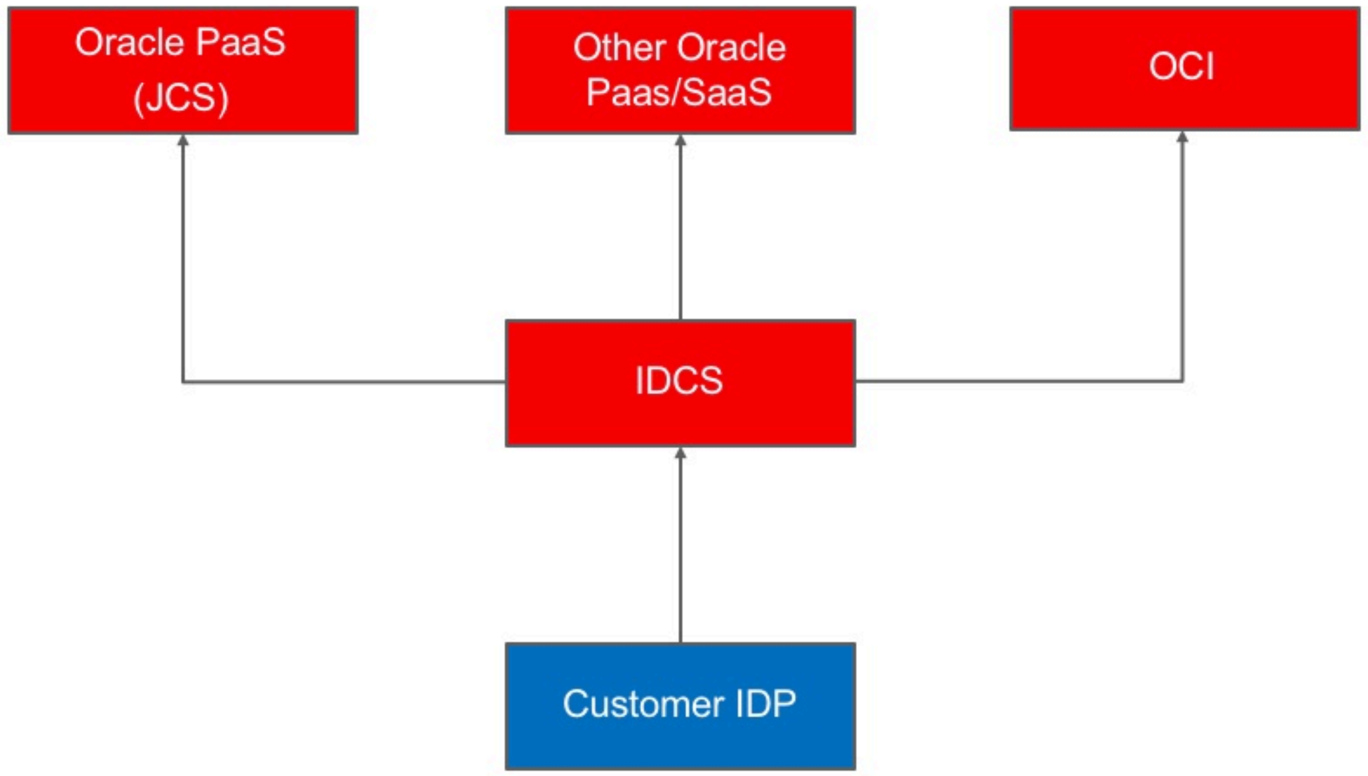
The login is uncommon for federated accounts. If you have questions, please contact your tenancy administrator.

USER NAME

PASSWORD

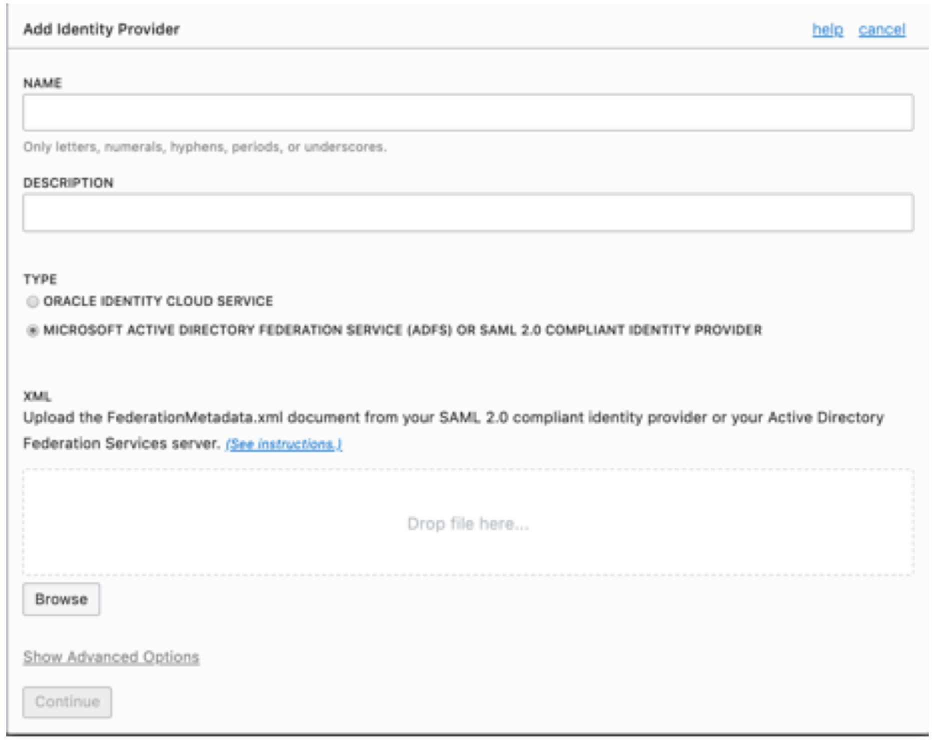
Sign In
Forgot password?

Federation with IDCS - Recommended



Federation with Microsoft Active Directory

- Get required information from Active Directory Federation Services.
- Federate Active Directory with Oracle Cloud Infrastructure:
  - Add the identity provider (AD FS) to your tenancy and provide the required information.
  - Map Active Directory groups to IAM groups.
- In Active Directory Federation Services, add Oracle Cloud Infrastructure as a trusted, relying party.
- In Active Directory Federation Services, add the claim rules required in the authentication response by Oracle Cloud Infrastructure.
- Test your configuration by logging in to Oracle Cloud Infrastructure with your Active Directory credentials.


Add Identity Provider

NAME

Only letters, numerals, hyphens, periods, or underscores.

DESCRIPTION

TYPE

ORACLE IDENTITY CLOUD SERVICE

[✓ MICROSOFT ACTIVE DIRECTORY FEDERATION SERVICE (ADFS) OR SAML 2.0 COMPLIANT IDENTITY PROVIDER]

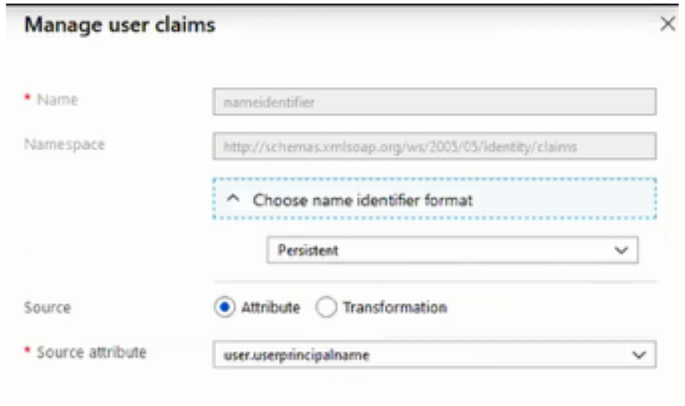
XML

Upload the FederationMetadata.xml document from your SAML 2.0 compliant identity provider or your Active Directory Federation Services server. (See instructions.)

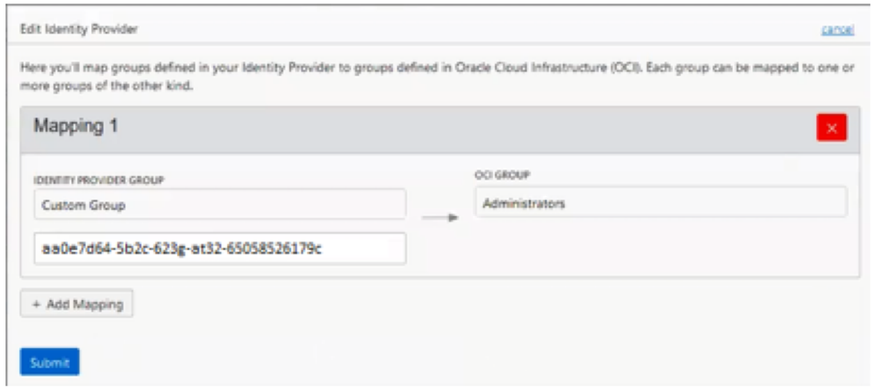
https://docs.cloud.oracle.com/iaas/Content/Identity/Tasks/federatingADFS.htm

Federation with Microsoft Azure Active Directory

- In Oracle Cloud Infrastructure, download the federation metadata document.
- In Azure AD, set up Oracle Cloud Infrastructure Console as an enterprise application.
- In Azure AD, configure the Oracle Cloud Infrastructure enterprise application for single sign-on.
- In Azure AD, set up the user attributes and claims.
- In Azure AD, assign user groups to the application.
- In Azure AD, download the Azure AD SAML metadata document.
- In Oracle Cloud Infrastructure, set up Azure AD as an identity provider.
- In Oracle Cloud Infrastructure, map your Azure AD groups to Oracle Cloud Infrastructure groups.
- In Oracle Cloud Infrastructure, set up the IAM policies to govern access for your Azure AD groups.
- Share the Oracle Cloud Infrastructure sign-in URL with your users.


Manage user claims

* Name	nameidentifier
Namespace	http://schemas.xmlsoap.org/ws/2005/05/identity/claims
Choose name identifier format	Persistent
Source	[✓ Attribute] Transformation
* Source attribute	user.userprincipalname


Edit Identity Provider

Here you'll map groups defined in your Identity Provider to groups defined in Oracle Cloud Infrastructure (OCI). Each group can be mapped to one or more groups of the other kind.

Mapping 1

IDENTITY PROVIDER GROUP	OCI GROUP
Custom Group	Administrators
aa0e7d64-5b2c-623g-at32-65058526179c	

https://docs.cloud.oracle.com/iaas/Content/Identity/Tasks/federatingADFSazure.htm

Reference IAM Model: Authentication and user management

All access by humans go through federation with a customer's corporate identity provider (IdP) to leverage their proven Auth mechanisms (MFA) and management capabilities (password complexity/rotation)

Use case	Feature
Human using console	Use SAML2.0 federation between corporate IdP and OCI IAM
Human using the CLI/SDK	Create a federated user with SCIM or an OCI IAM user with an API signing key
Human using a PaaS/SaaS app	Use SAML2.0 federation between corporate IdP and OCI IAM
Code running in OCI that calls OCI native APIs	Use Instance Principals
Code running outside OCI that calls OCI APIs	Create an OCI IAM "user" with an API signing key. The "user" in this case represents a software agent, not a human
"Break-glass" access by a human when federation fails	• Create an OCI IAM user in the default Admins group • Set a random Console password of sufficient length/complexity • Store this password in a software password manager or physical safe • Password is for infrequent use and should not be human memorizable • Use once – rotate password after every use • Monitor via Audit Service • Alarm on any use or attempted use of "break-glass" user • Outside the "break-glass" scenario, there is no reason to have an OCI IAM user with a Console password

Part V: Reference IAM model for Enterprises

Compartments

- A compartment is a collection of related resources (VCN, instances,..) that can be accessed only by groups that have been given permission (by an administrator in your organization)
- Compartments help you organize and control access to your resources
- Design considerations:
  - When creating a resource (for example, instance, block storage volume, VCN, subnet), you must decide in which compartment to put it.
  - Compartments are logical, not physical, so related resource components can be placed in different compartments.
  - You can create a hierarchy of compartments up to six compartments deep under the tenancy (root compartment).
  - When you write a policy rule to grant a group of users access to a resource, you always specify the compartment to apply the access rule to. So if you choose to distribute resources across compartments, remember that you will need to provide the appropriate permissions for each compartment for users that will need access to those resources.
  - If you want to delete a compartment, you must delete all resources in the compartment first.

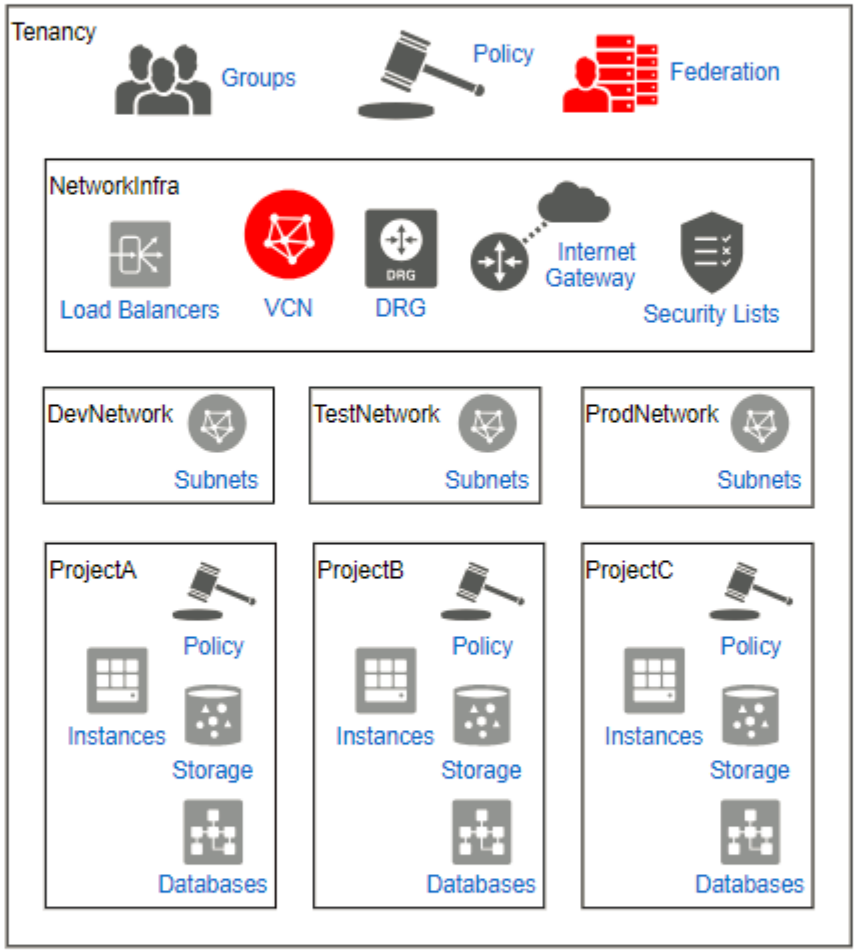
Moving Resources to a Different Compartment

- Most resources can be moved after they are created. There are a few resources that you can't move from one compartment to another.
- Some resources have attached resource dependencies and some don’t. Not all attached dependencies behave the same way when the parent resource moves.
- For some resources, the attached dependencies move with the parent resource to the new compartment. The parent resource moves immediately, but in some cases attached dependencies move asynchronously and are not visible in the new compartment until the move is complete.
- For other resources, the attached resource dependencies do not move to the new compartment. You can move these attached resources independently.
- After you move the resource to the new compartment, the policies that govern the new compartment apply immediately and affect access to the resource. Depending on your compartment organization, metering, billing, and alarms can also be affected.

Moving a Compartment to a Different Parent Compartment

- Be aware of following complications before you move a compartment:
  - Required IAM Policy
  - Restrictions on Moving Compartments
  - Understanding the Policy Implications When You Move a Compartment
  - Understanding Compartment Quota Implications When You Move a Compartment
  - Understanding Tagging Implications When You Move a Compartment

Reference IAM Model: Compartments



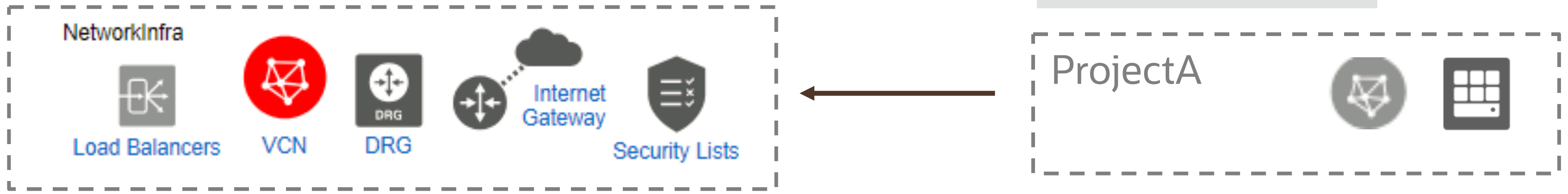
- Compartment: NetworkInfra
  - Critical network infrastructure that should be centrally managed by network admins
  - Resources: Security Lists, Internet Gateways, DRGs, the top-level VCN(s), etc.
- Compartment: ProdNetwork
  - Production environment that may or may not be centrally managed but is typically under change management
  - Modeled as a separate compartment to easily write policy about who can use (i.e. attach resources to) the network
  - Optionally Databases and Storage may be included here depending on whether they are shared resources or not
  - Resources: Subnets, (Databases), (File Storage)

Reference IAM Model: Compartments

Tenancy

Groups
NetworkAdmins (John)

Groups
A-Admins (Tom)



- Allow group NetworkAdmins to MANAGE virtual-network-family in compartment NetworkInfra
- Allow group NetworkAdmins to manage instance-family in compartment NetworkInfra

- John creates a Network in NetworkInfra compartment
- John can't terminate, reboot or launch new instances into ProjectA compartment

- Allow group A-Admins to USE virtual-network-family in compartment NetworkInfra
- Allow group A-Admins to manage all-resources in compartment ProjectA

- Tom launches instances in ProjectA using the VCN in NetworkInfra compartment
- Tom cannot launch instance inside the NetworkInfra compartment

The instances Tom launched reside in the VCN from a network topology standpoint but from an access standpoint, they're in the ProjectA compartment, not the NetworkInfra compartment where the VCN is

Example IAM compartment and policy design

Set up compartment admins that have authority to create policies for users for that compartment. These are not tenancy admins

Let the tenancy admin:

- Create the required users, and two groups: *mycompartmentadmins*, *mycompartmentusers*.
- Assign the compartment admins to the *mycompartmentadmins* group.
- Create a compartment: e.g., *mycompartment*.
- Create a policy, attached at the tenancy level:

```
Allow group mycompartmentadmins to use users in tenancy
Allow group mycompartmentadmins to manage groups in tenancy where target.group.name='mycompartmentusers’
Allow group mycompartmentadmins to manage policies where target.compartment.name = mycompartment
```

Your compartment admin should be able to:

- Assign the required users to *mycompartmentusers*.
- Define a policy (attached to *mycompartment*) for the specific permissions to be granted to the *mycompartmentusers* group.For example, if you want the users in that group to be able to manage all the resources in the compartment:

```
Allow group mycompartmentusers to manage all-resources in compartment mycompartment
```

Summary

You should now be familiar with the following

- Using Instance principals for your applications
- Advanced Policy Syntax
- Multi-factor Authentication
- Federating OCI with Oracle Identity Cloud Service (IDCS)
- Federating OCI with Microsoft Active Directory
- Federating OCI with Microsoft Azure Active Directory
- Reference IAM model
    - Compartments design
    - Example IAM compartment and policy design

Oracle Cloud always free tier:
oracle.com/cloud/free/

OCI training and certification:
oracle.com/cloud/iaas/training
oracle.com/cloud/iaas/training/certification
education.oracle.com/oracle-certification-path

OCI hands-on labs:
ocitraining.qloudable.com/provider/oracle

Oracle learning library videos on YouTube:
youtube.com/user/OracleLearning