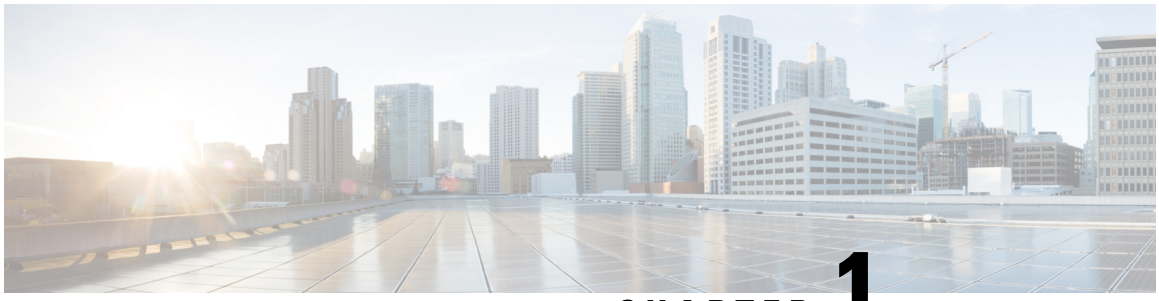
Cisco Crosswork Workflow Manager 1.0 Administrator Guide

First Published: 2023-06-01

Last Modified: 2023-07-20

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883



CHAPTER 1

Install CWM using OVA

This section contains the following topics:

Install CWM using OVA

The Crosswork Workflow Manager 1.0 is installed as a guest virtual machine by deploying an OVA image using the vSphere vCenter 7.0 virtualization platform.

Prerequisites

- vSphere vCenter 7.0 account with an ESXi 7.0 host.

Download CWM package

Before you begin

To get the Crosswork Workflow Manager 1.0 software package:

Step 1 Go to the Cisco Software Download service and in the search bar, type in '**Crosswork Workflow Manager**', then select it from the search list.

Step 2 From Select a software type, select **Crosswork Workflow Manager Software**.

Step 3 Download the Crosswork Workflow Manager software package for Linux.

Step 4 In a terminal, run the self-extracting signed binary. This extracts the `cwm-1.0.tar.gz` file and validates using the signature file.

Step 5 To extract the `cwm-1.0.tar.gz` file, double click on it (Mac users) or use gzip utility (Linux and Windows users). This will extract the CWM OVA file.

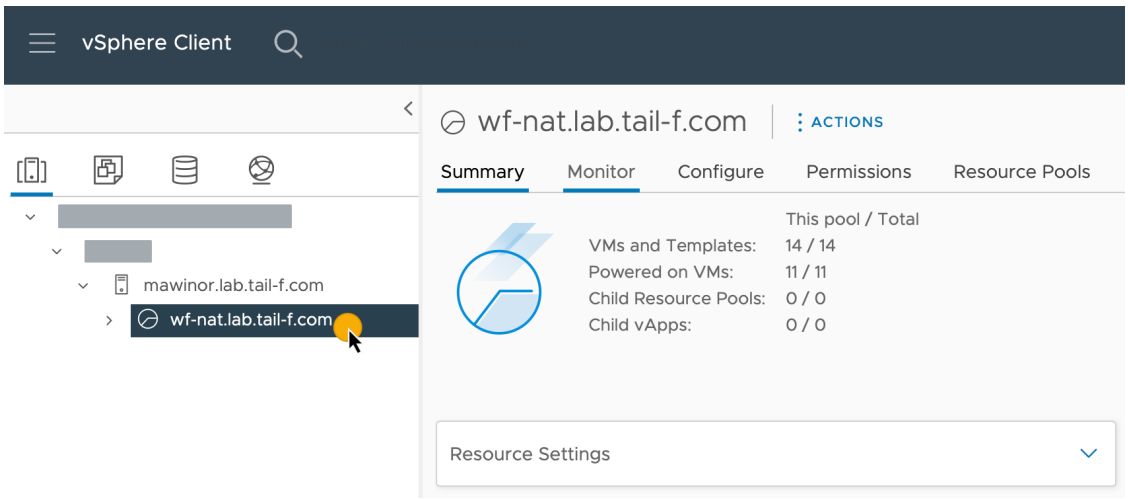
Deploy OVA and start VM

To create a virtual machine using the downloaded OVA image:

Step 1 Log in to your vSphere account.

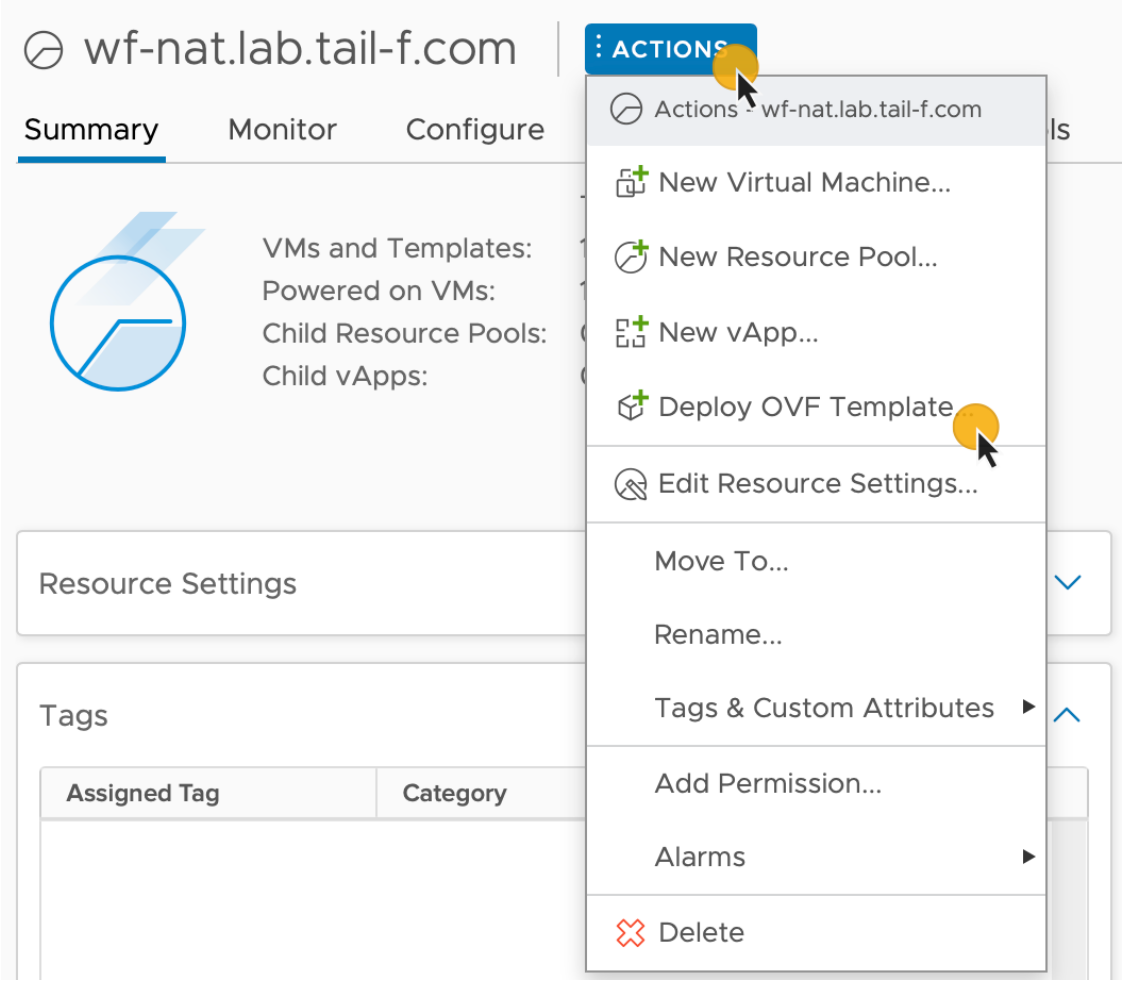
Step 2 In the **Hosts and Clusters** tab, expand your host and select your resource pool.

Figure 1: Resource pool



Step 3 Click the **Actions** menu and select **Deploy OVF Template**.

Figure 2: Deploy OVF template



Step 4 In the **Select an OVF template** step, click **Local file**, **Select files**, and select the CWM OVA image. Click **Next**.

Step 5 In the **Select a name and folder** step, provide a name for your VM and select it's location. Click **Next**.

Step 6 In the **Select a compute resource** step, select your resource pool. Click **Next**.

Step 7 In the **Review details** step, click **Next**.

Step 8 In the **Select storage** step, set **Select virtual disk format** to **Thin provision** and select your storage, then click **Next**.

Step 9 In the **Select network** step, you need to select destination networks for the **Control Plane** and **Northbound**:

a) **Note** Control plane settings are essential only in case of an HA cluster setup. For single-node setups, control plane settings need to be provided, but are not essential and should not conflict with any other devices connected to the control network.

Control Plane: select **PrivateNetwork**. If not available, select **VM Network**.

b) **Northbound**: select **VM Network**.

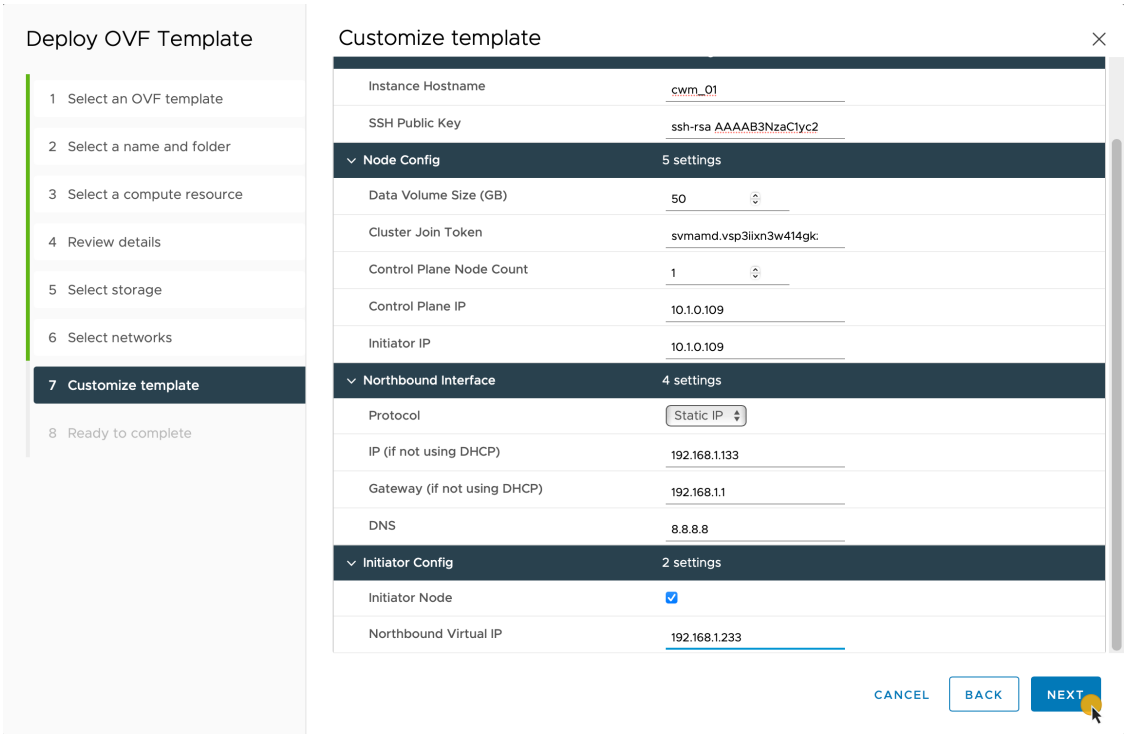
c) Click **Next**.

Step 10 In the **Customize template** step, provide the following selected properties:

a) **Instance Hostname**: type a name for your instance.

b) **SSH Public Key**: provide an SSH public key used for command-line access to the VM.
c) **Control Plane Node Count**: change to more than 1 only in case of HA cluster setup. Not supported for CWM version 1.0.
d) **Control Plane IP**: provide a network address for the control plane. This address cannot conflict with any other devices in the control network, but is otherwise inessential in a single-node setup.
e) **Initiator IP**: set the initiator IP for the starter node. In a single-node setup, it is the same address as *Control Plane IP**.
f) **IP (if not using DHCP)**: provide the network address for the node.
g) **Gateway (if not using DHCP)**: provide the gateway address. By default, it is `192.168.1.1`.
h) **DNS**: provide the address for the DNS. By default, it is `8.8.8.8`, or you can use your local DNS.
i) **Northbound Virtual IP**: provide the network address for the active cluster node. In a single-node setup this address is also required, as this is where the HTTP service is working.
j) Click **Next**.

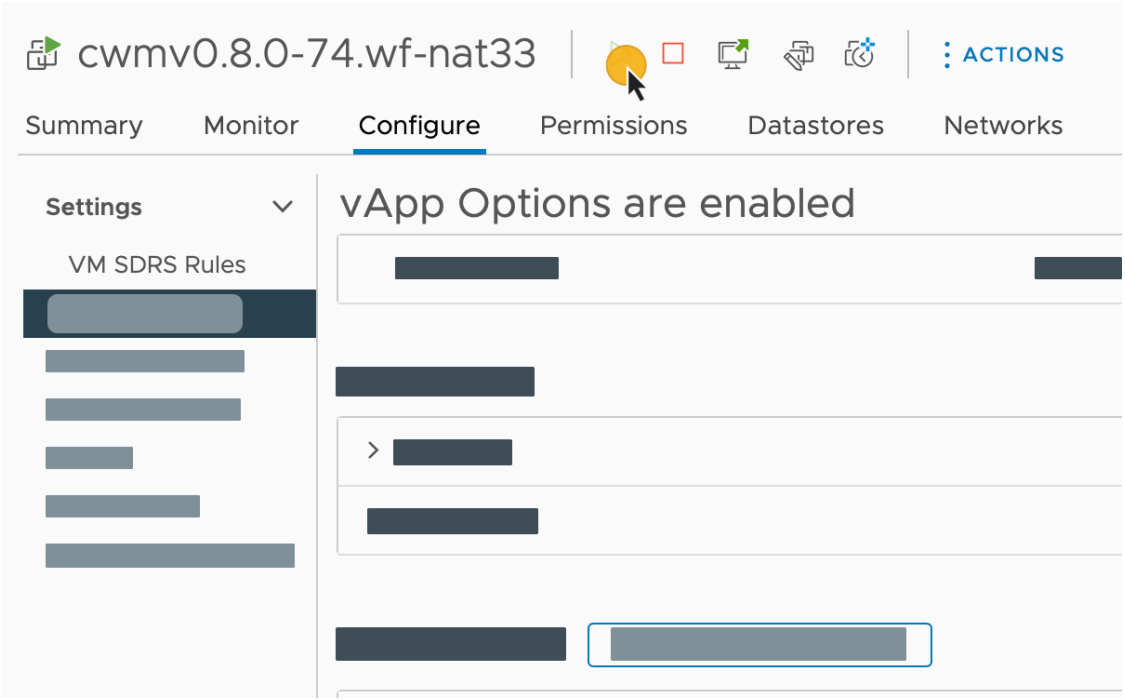
Figure 3: Customize template



Step 11 In the **Ready to complete**, click **Finish**. The deployment may take a few minutes.

Step 12 From the **Resource pool** list, select you newly created virtual machine and click the **Power on** icon.

Figure 4: Power on VM



Note If the VM doesn't power on successfully, this might be due to an intermittent infrastructure error caused by NxF. As a workaround, remove the existing VM and redeploy the OVA on a new one.

Create user

You can create CWM platform user accounts using the command-line access to the VM. Here's how to do it:

Step 1 Using a command-line terminal, log in to the NxF in your guest OS with SSH:

```
ssh -o UserKnownHostsFile=/dev/null  -p 22  nxf@<your_resource_pool_address>
```

Note The default port for SSH is `22`, change it to your custom port if applicable.

a) Optional: If you are logging in for the first time, provide the path name for your private key:

```
ssh -i <your_ssh_private_key_name_and_location> nxf@<your_resource_pool_address>
```

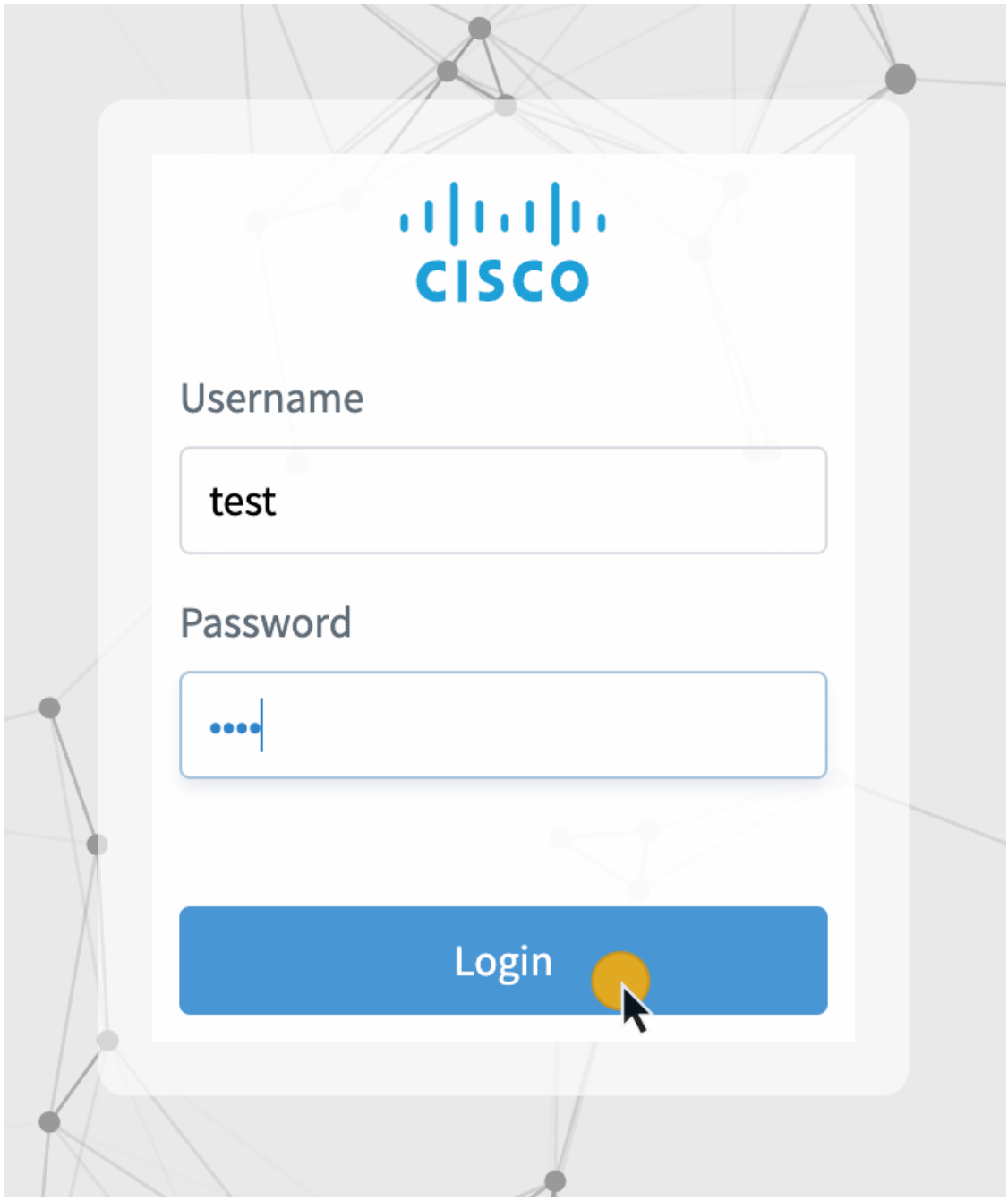
Step 2 To create a user, run the following command:

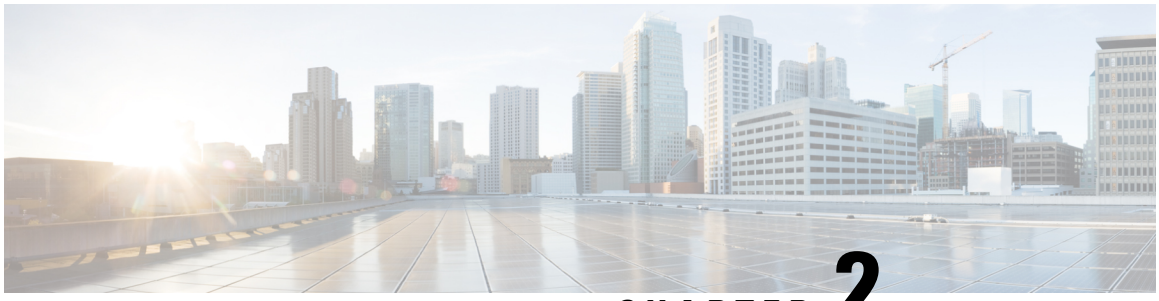
```
echo -en "test" | sedo security user add --password-stdin --access permission/admin --display-name
Tester test
```

Step 3 Go to the address that you selected for your node and default port `8443`. For example, https://wf.lab.cisco.com:8443/.

Step 4 Log in using the `test` username and password.

Figure 5: Log in





CHAPTER 2

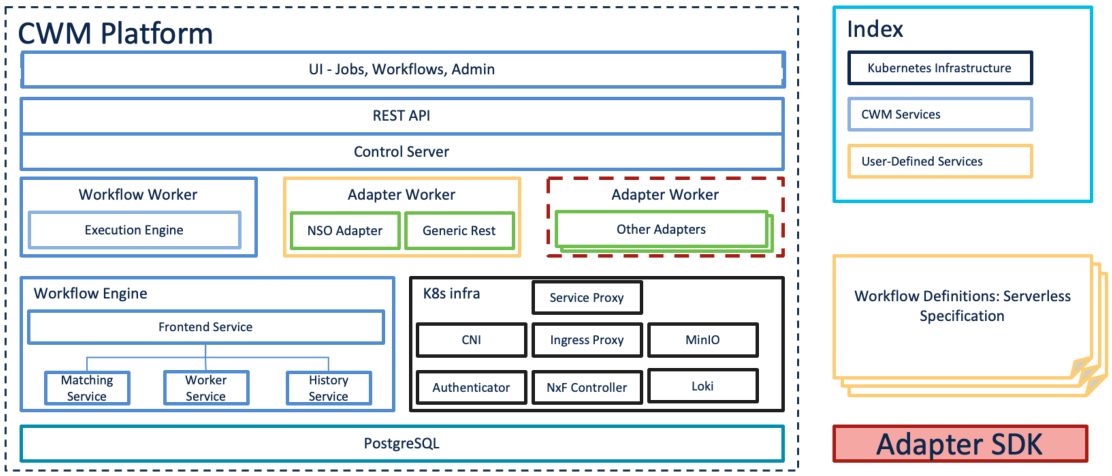
Architecture overview

This section contains the following topics:

Architecture overview

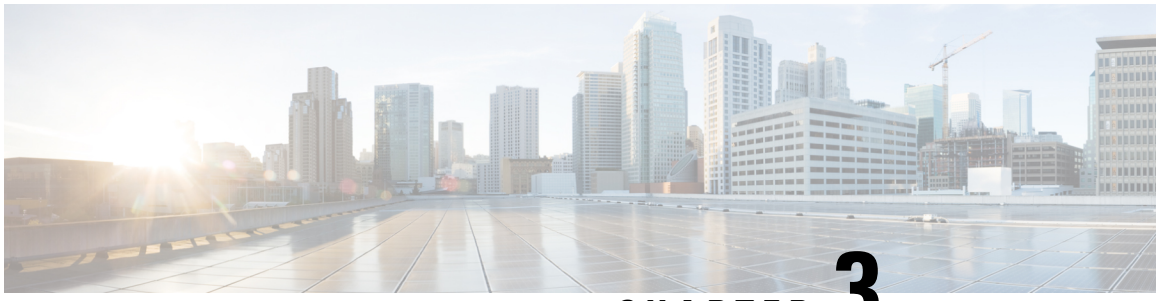
The Crosswork Workflow Manager architecture is a microservice-based solution that operates on top of the Kubernetes container orchestration system. This section shows a diagram presenting its core architectural components along with short descriptions of each.

Figure 6: Architecture overview



- **User Interface (UI)**: allows operators to add and instantiate workflows, enter workflow data, list running workflows, monitor job progress. The **Admin** section of the UI enables adding workers, managing worker processes and assigning activities from adapters to workers.
- **REST API**: includes all interaction with the CWM application: deploying adapters, publishing and instantiating workflows, managing workers, resources and secrets.
- **Control Server**: dispatches API requests to relevant microservices.
- **Workflow Engine**: it is the core component that conducts how workflows are handled; it interprets and manages the execution of workflow definitions.

- **Execution Engine (Workflow Worker)**: it is responsible for executing the workflow tasks. It receives the workflow tasks from the **Workflow Engine**, executes them in the correct order, and sends the results back to the **Workflow Engine**.
- **Adapter Workers**: they are processes responsible for executing the tasks defined in workflow definitions and adapter code. They receive the tasks from the **Workflow Worker**, execute them, and send the results back to the **Workflow Worker**. The Execution Workers are capable to load additional adapters as plugins, which allows them to work with different systems and technologies.
- **Adapters**: they interface and integrate with external systems, applications and technologies. Inside them, activities that can be consumed in a workflow are defined.
- **Adapter SDK**: a Software Development Kit that helps developers create new adapters to integrate with external systems.
- **Workflow Definitions**: workflow code written in the JSON format based on the Serverless Workflow specification.
- **K8s Infrastructure**: runtime platform for the CWM application. It is a collection of services that provide the necessary infrastructure to support the deployment and management of the application within a Kubernetes cluster.
- **PostgreSQL**: it is the database used by the system to store and manage its data.



CHAPTER 3

Manage

This section contains the following topics:

Manage adapters

To interact with external target systems, CWM requires adapters. You can manage them using the CWM API. The following API endpoints are available for handling adapters:

- `GET/adapter`: gets a list of adapters existing in the CWM application.
- `POST/adapter`: uploads an adapter **.tar** file to CWM storage.
- `GET/adapter/{adapterId}`: gets the details of a specific adapter existing in the CWM application. Among others, it lists all the activities available in the adapter.
- `PUT/adapter/{adapterId}`: updates an existing adapter file with a new adapter version.
- `DELETE/adapter/{adapterId}`: deletes an adapter from the CWM application.
- `POST/adapter/{adapterId}/deploy`: deploys an adapter in the system based on the uploaded adapter file.

Install adapter

CWM adapters come in **.tar** installation files. Before they can be used in a workflow, they need to be uploaded to storage and deployed in the system. Here's how to do it.

Upload adapter file

Before you deploy an adapter, you need to upload the adapter **.tar** file to CWM storage:

Step 1 Get a latest adapter installation file or create your own adapter.

Step 2 Log in to CWM and from the navigation menu on the left, click the **swagger** icon.

Step 3 In the **adapters** section, click the `POST/adapter` endpoint to expand it. Inside the endpoint, click **Try it out**.

Step 4 In the subsection that appears, click **Choose File**, select the adapter **.tar** installation file and click **Upload**, then click **Execute**.

If the server response code is `201`, the adapter file is successfully uploaded into the CWM database.

Deploy adapter

Step 1 In the CWM API **adapters** section, click the `GET/adapter` endpoint to expand it. Inside the endpoint, click **Try it out** and **Execute**.

Step 2 From the server response body, copy the value of the `id` field for your uploaded adapter.

Step 3 In the CWM API **adapters** section, click the `POST/adapter/{adapterId}/deploy` endpoint to expand it.

Step 4 Inside the endpoint, click **Try it out**. Paste the adapter id into the **Adapter ID** field.

Step 5 In the **createWorker** field, you may set the `createWorker` parameter to `true`. This will create a worker with the same name as the adapter id.

Step 6 Click **Execute**.

If the server response code is `201`, the adapter plugin is successfully installed.

Delete adapter

To delete an adapter permanently from storage and "uninstall" it:

Step 1 In the CWM API **adapters** section, click the `GET/adapter` endpoint to expand it. Inside the endpoint, click **Try it out** and **Execute**.

Step 2 From the server response body, copy the value of the `id` field for your uploaded adapter.

Step 3 In the CWM API **adapters** section, click the `DELETE/adapter/{adapterId}` endpoint to expand it.

Step 4 Inside the endpoint, click **Try it out**. Paste the adapter id into the **Adapter ID** field.

Step 5 Click **Execute**.

Manage workers

Workers are processes that execute actions defined in workflow definitions and adapter code. You can manage them using the CWM UI as described in the **Operator** guide, or with the CWM API, as described below.

The following actions for managing workers are available:

- `GET/worker`: gets a list of workers existing in the CWM application.
- `POST/worker`: creates a new worker in the CWM application.

- `GET/worker/{workerName}`: gets the details of a specific worker existing in the CWM application.
- `PUT/worker/{workerName}`: updates an existing worker with new parameter values.
- `DELETE/worker/{workerName}`: deletes a worker from the CWM application.
- `POST/worker/{workerName}/start`: activates a worker created in the application.
- `POST/worker/{workerName}/stop`: deactivates a worker created in the application.

Create worker

Step 1 Log in to CWM and from the navigation menu on the left, click the **swagger** icon.

Step 2 In the CWM API **workers** section, click the `POST/worker` endpoint to expand it. Inside the endpoint, click **Try it out**.

Step 3 In the **Worker data** field, provide the required values:

a) "activities": paste the ID of your deployed adapter or specific adapter activity.
b) "startWorker": set to `true`.
c) "workerName": provide a name for your worker.

Step 4 Click **Execute**.

Start worker

Step 1 In the CWM API **workers** section, click the `POST/{workerName}/start` endpoint to expand it. Inside the endpoint, click **Try it out**.

Step 2 In the **parameters** fields, provide the required values:

a) "Name of a worker to start": paste the name the worker to be started.
b) "forceReload": set to `true` if you want to force the worker to start.

Step 3 Click **Execute**.

Stop worker

Step 1 In the CWM API **workers** section, click the `POST/{workerName}/stop` endpoint to expand it. Inside the endpoint, click **Try it out**.

Step 2 In the **parameters** fields, provide the required values:

a) "Name of a worker to stop": paste the name the worker to be stopped.
b) "forceStop": set to `true` if you want to force the worker to stop.

Step 3 Click **Execute**.

Manage workflows

Workflow instances can be managed both in the CWM UI as described in the **Operator** guide, or using the CWM API:

- `POST/workflow`: creates a new workflow instance in the CWM application.
- `GET/workflow/list`: gets a list of adapters existing in the CWM application.
- `GET/workflow/{id}`: gets the details of a specific workflow existing in the CWM application.
- `PUT/workflow/{id}`: updates an existing workflow with new workflow definition.
- `DELETE/workflow/{id}`: deletes a selected workflow from the CWM application.

Note The recommended method for managing workflows is through CWM UI. For details, refer to the **Operator** guide.

Manage resources and secrets

For CWM, adapters define activities that enable to execute actions in external entities, such as other systems or applications. These entities are, in most cases, integrated via APIs which usually require connection and authentication data. CWM provides a framework where when an activity is consumed in a workflow, the details of a connection endpoint and authentication data can be passed at runtime. Thus, the operator who runs a workflow may not know any details of these systems (resources) such as IP addresses, ports or usernames and password.

CWM provides a framework for secure handling of resources and secrets in database and identifying them by their respective IDs. When running a workflow instance, just the resource ID needs to be passed, with the rest of the data sent to the adapter by the Resource Manager without any intervention from the Operator or additional development from Adapter developer.

Resource and secret types

You can think of resource and secret types are buckets used to organize resources and secrets created by users by their type. Types are defined inside a given adapter and are added to the system automatically upon installing the adapter. You can list secrets belonging to a specific type using the `GET/secret/type/{type}` API endpoint.

Secrets API endpoints

The following actions for managing secrets are available:

- `GET/secret`: gets a list of secrets existing in the CWM application.
- `POST/secret`: creates a new secret in the CWM application.
- `GET/secret/type/{type}`: lists secrets existing in the CWM application that belong to a specific type.

- `GET/secret/types`: gets a list of secret types existing in the CWM application.
- `GET/secret/{id}`: gets details of an existing secret.
- `DELETE/secret/{id}`: deletes a secret from the CWM application.
- `PATCH/secret/{id}`: updates a secret existing in the CWM application with new parameter values.

Resources API endpoints

The following actions for managing resources are available:

- `GET/resource`: gets a list of resources existing in the CWM application.
- `POST/resource`: creates a new resource in the CWM application.
- `GET/resource/{resourceId}`: gets the details of a specific resource existing in the CWM application.
- `PUT/resource/{resourceId}`: updates an existing resource with new parameter values.
- `DELETE/resource/{resourceId}`: deletes a resource from the CWM application.
- `GET/resourceType`: gets a list of resource types existing in the CWM application.
- `GET/resourceType/{resourceId}`: gets the details of an existing resource type.

Create secret

Step 1 Log in to CWM and from the navigation menu on the left, click the **swagger** icon.

Step 2 In the CWM API **secrets** section, click the `POST /secret` endpoint to expand it.

Step 3 Inside the endpoint, click **Try it out**, and provide your data into the **Secret input** field. Example input can look like this:

```
{
"secret": {
"username": "admin",
"password": "admin"
},
"secretId": "NSOSecret",
"secretType": "basicAuth"
}
```

Step 4 Click **Execute**.

If the server response code is `201`, the secret is successfully created and you can start creating a resource to associate the secret with.

Create resource

Step 1 In the CWM API **resources** section, click the `POST /resource` endpoint to expand it.

Step 2 Inside the endpoint, click **Try it out**, and provide your data into the **Resource input** field. Example input can look like this:

```
{
    "resource": {
        "scheme": "http",
        "host": "127.0.0.1",
        "port": 8080
    },
    "resourceId": "NSOLocal",
    "resourceType": "cisco.nso.resource.v1.0.0",
    "secretId": "NSOSecret"
}
```

Step 3 Click **Execute**.

If the server response code is `201`, the resource is successfully created.

User access via NxF

The CWM allows you to manage user access and permissions via NextFusion (NxF). NxF adds an additional layer of security and works as a Single Authentication Agent, thus sharing local, LDAP and SAML users.

Users, roles, and permissions

Currently, only one role and permission type (admin) is supported. Every user is associated with admin permissions by default.

To allow access to CWM to a larger group of regular users, set the user authentication via LDAP or/and SAML SSO protocols (you can have both at the same time), depending on your environment.

Permissions scope

Admin role has full access to the CWM and all of its functionalities; admins can control user access and permissions. All local users with admin permissions can create new users as needed.

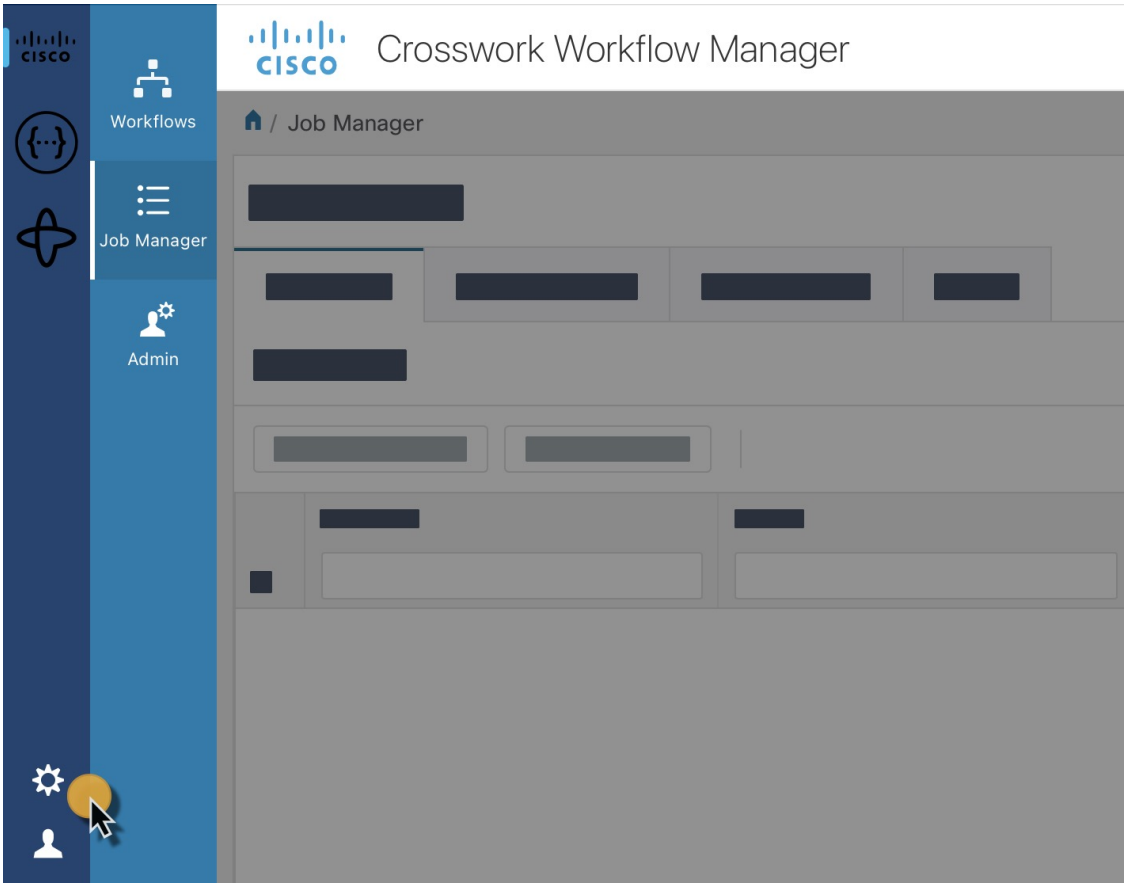
NxF functionality in CWM

NxF functionality is available for admin users from the **Settings** tab in CWM UI. To access NxF functionality in the CWM:

Step 1 In the CWM, go to the outermost navigation menu on the left.

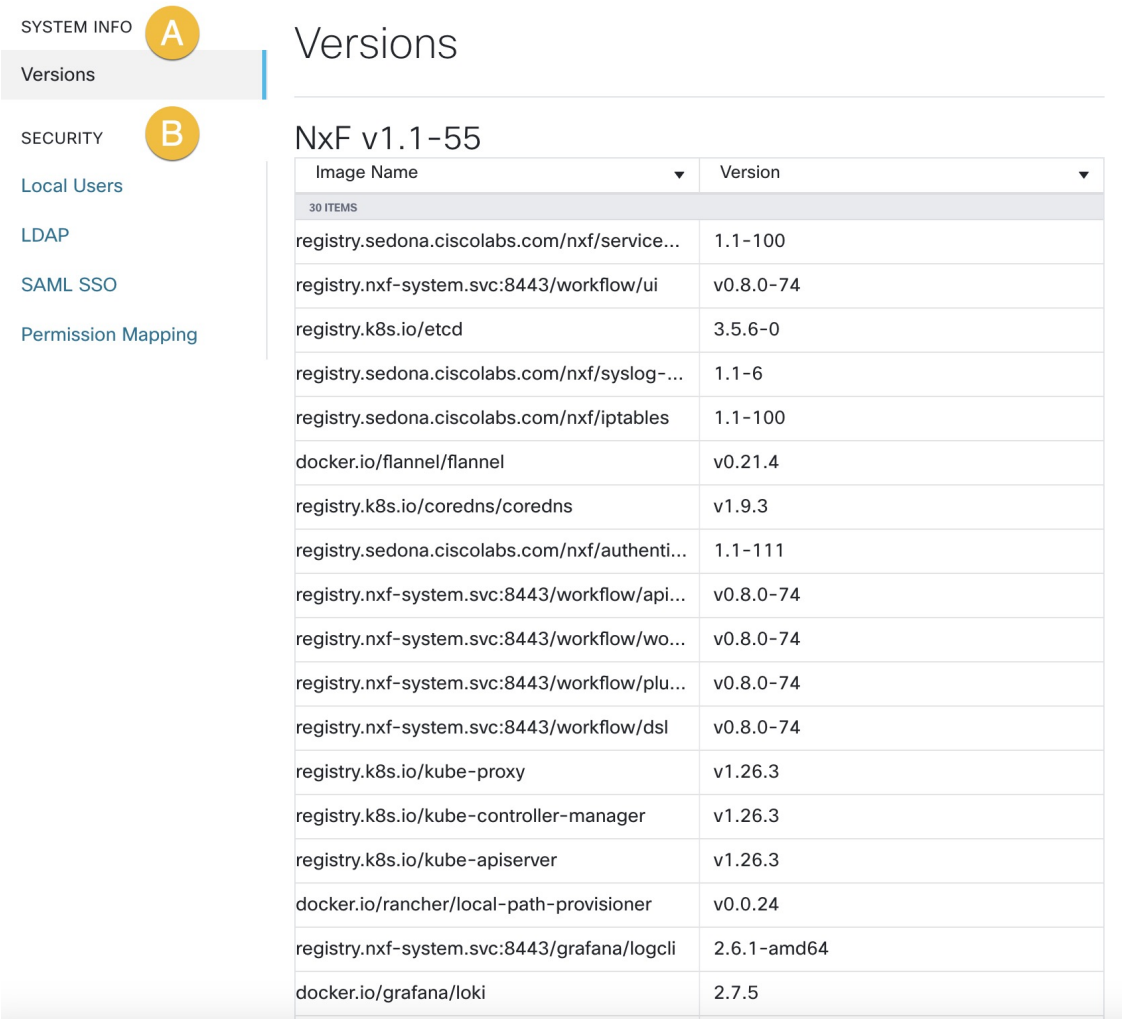
Step 2 Click the **Settings** icon (gear icon).

Figure 7: NxF settings



Step 3 In the expanded drawer, you can find the following:

Figure 8: Settings drawer



a) **System Info** section with information about the latest versions of NxF and CWM microservices.
b) **Security** section for access management:

- **Local Users**: where you can display, create and edit local users via UI.
- **LDAP**: where you can set LDAP settings for user authentication.
- **SAML SSO**: where you can set SAML Single-Sign-On settings for user authentication.
- **Permission Mapping**: where you can handle permission management via Cisco Policy Management Tool.

Add local user

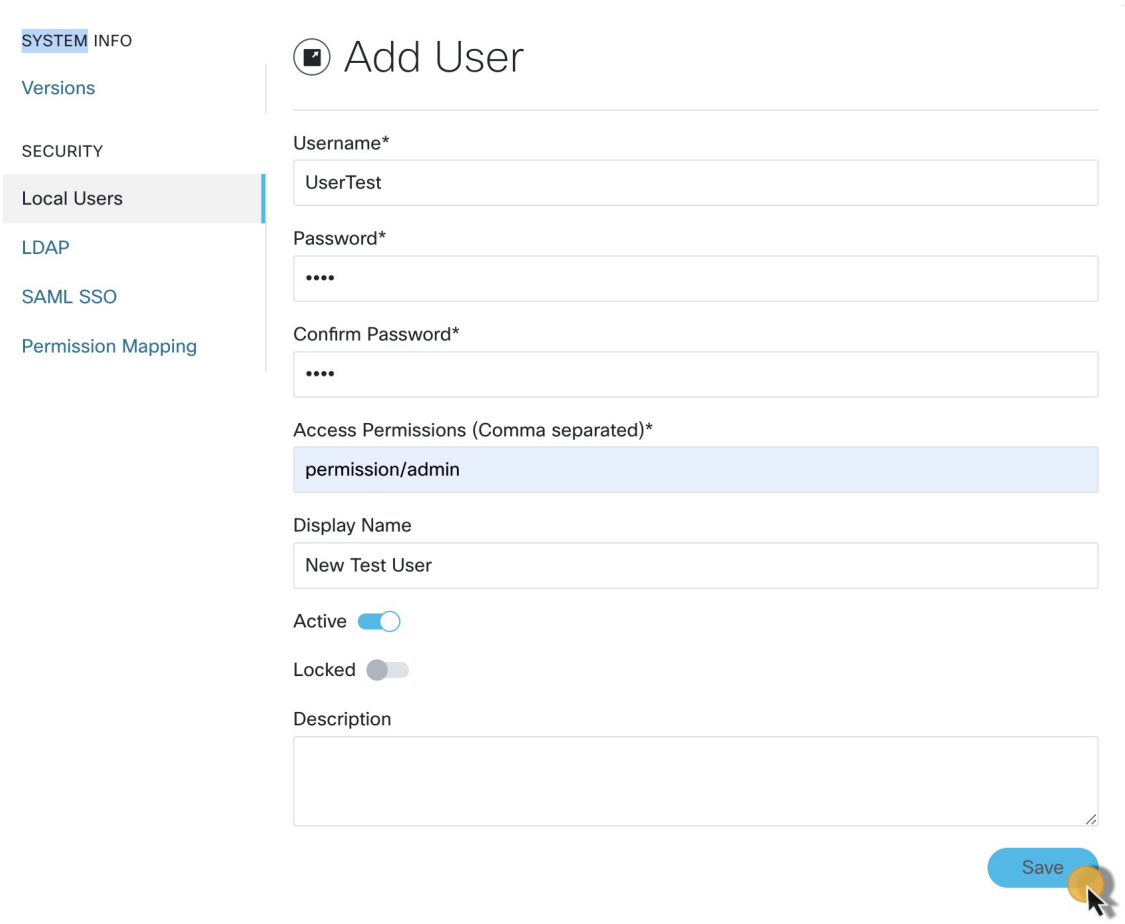
Step 1 In the CWM, go to the outermost navigation menu on the left.

Step 2 Navigate to **CWM** (Cisco icon) -> **Local Users** tab.

Step 3 Click **Add...**

Step 4 In the Add User panel, fill in the mandatory fields (marked with an asterisk): Username (used to log in to the CWM), Password, Confirm Password and Access Permissions (enter `permission/admin`). The Description and Display Name (visible next to the username in the CWM) are optional fields.

Figure 9: NxF Add User



Step 5 Use radio buttons to set the user status. You can make both radio buttons disabled or enabled at the same time.

- **Active enabled**: allows the user to log in to the CWM.
- **Active disabled**: forbids the user to log in to the CWM.
- **Locked enabled**: prevents deleting the user.
- **Locked disabled**: allows removal of the user.

Step 6 Click **Save**.

Set up authentication via LDAP

Besides supporting local users, CWM allows adding LDAP users through integration with LDAP (Lightweight Directory Access Protocol) servers.

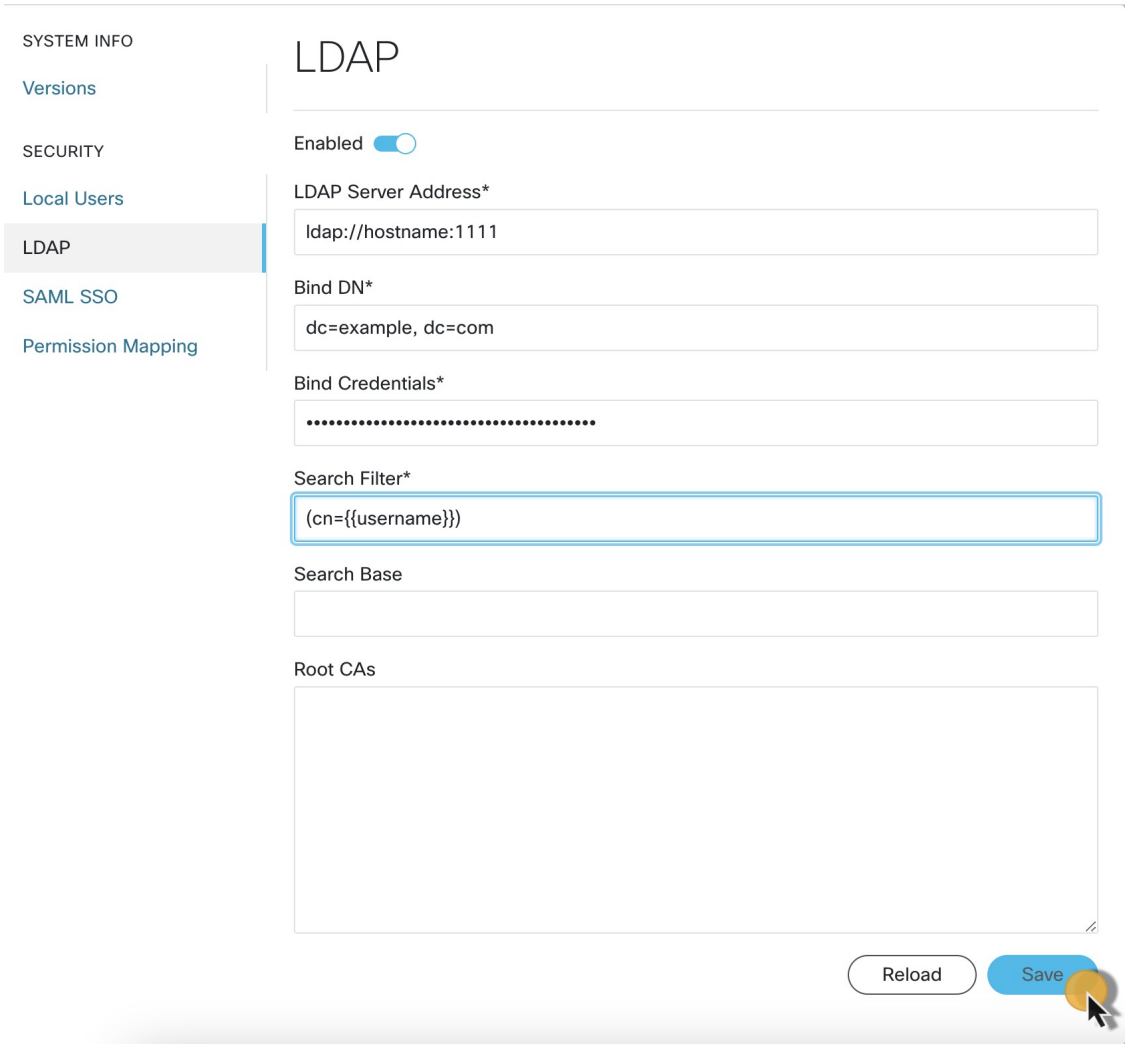
Step 1 In the CWM, go to the outermost navigation menu on the left.

Step 2 Navigate to **CWM** (Cisco icon) -> **LDAP** tab.

Step 3 Click the **Enabled** radio button.

Step 4 Fill in the mandatory fields (marked with an asterisk): LDAP Server Address, Bind DN, Bind Credentials and Search Filter. Search Base and Root CAs are optional.

Figure 10: LDAP



Step 5 Click **Save**.

Set up authentication via SAML SSO

CWM offers SAML SSO feature that supports both LDAP and non-LDAP users to gain single sign-on access based on the protocol SAML (Security Assertion Markup Language). You can enable SAML SSO for CWM along with LDAP or without it.

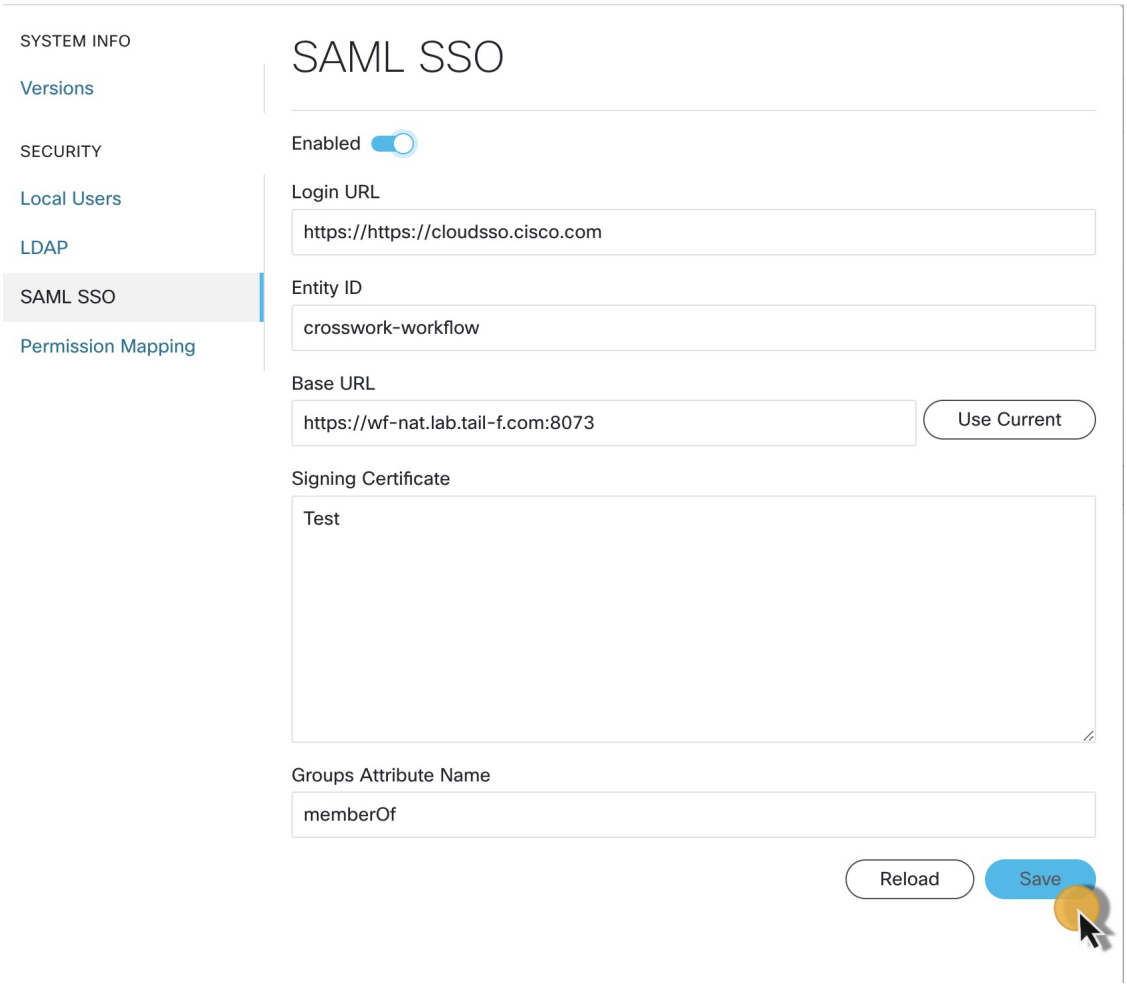
Step 1 In the CWM, go to the outermost navigation menu on the left.

Step 2 Navigate to **CWM** (Cisco icon) -> **SAML SSO** tab.

Step 3 Click the **Enabled** radio button.

Step 4 Fill in the mandatory fields: Login URL, Entity ID, Base URL, Signing Certificate and Groups Attribute Name.

Figure 11: NxF SAMLSSO



Step 5 Click **Save**.

Set up permission mapping

You can give specific permissions to a group of users via Cisco Policy Management Tool (PMT).

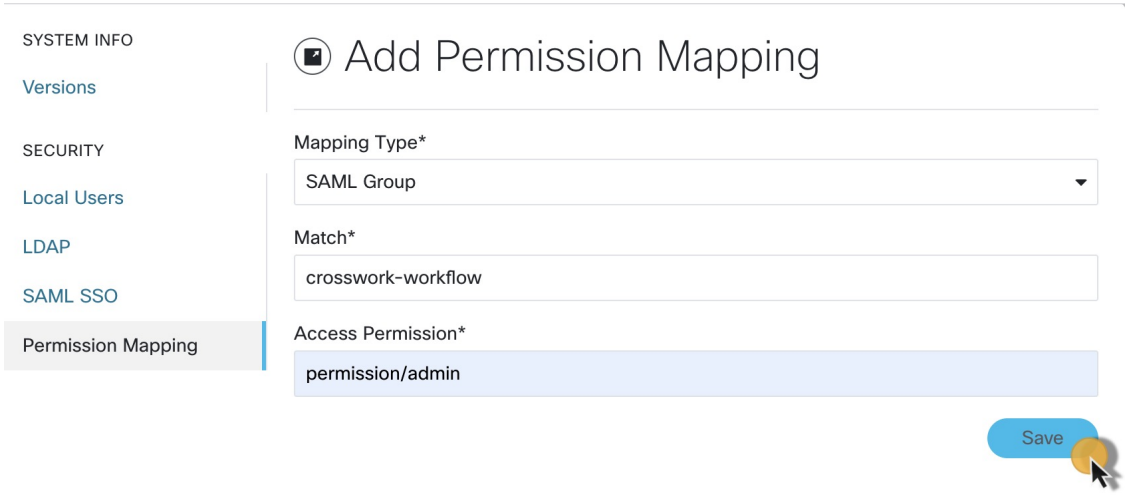
Step 1 In the CWM, go to the outermost navigation menu on the left.

Step 2 Navigate to **CWM** (Cisco icon) -> **Permission Mapping** tab.

Step 3 Click **Add...**.

Step 4 In the Add Permission Mapping panel, choose one **Mapping Type** from the dropdown menu: SAML User, SAML Group, LDAP User, or LDAP Group.

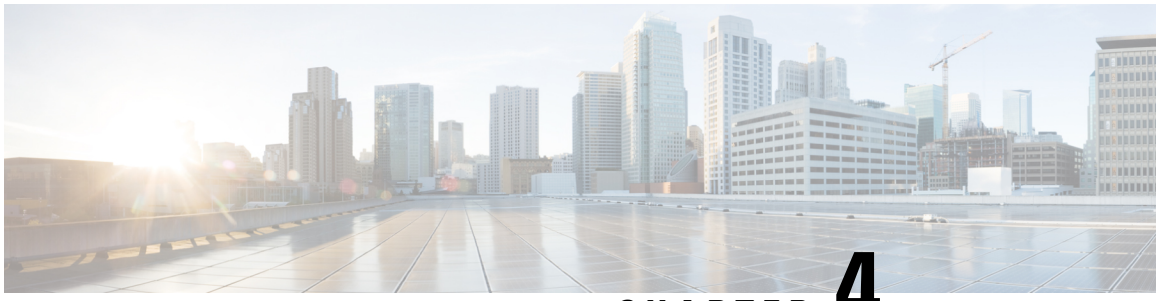
Figure 12: Permission mapping



Step 5 Fill in the Match field with the entry from the Cisco Policy Management Tool. You can find the match in PMT UI -> **OAuth Clients** tab -> Client ID Column.

Step 6 Enter appropriate permission (for example `permission/admin`) in the Access Permission field.

Step 7 Click **Save**.



CHAPTER 4

Platform health and logs

This section contains the following topics:

Platform health and logs

CWM is a microservice-based application that leverages Kubernetes cluster architecture as its runtime environment. The health of the CWM application can thus be checked using Kubernetes commands.

Note To see all the supported `kubectl` commands, log in to the OS on your VM and use `kubectl --help`.

Check pod status

Step 1 Using a command-line terminal, log in to the OS on your virtual machine with SSH:

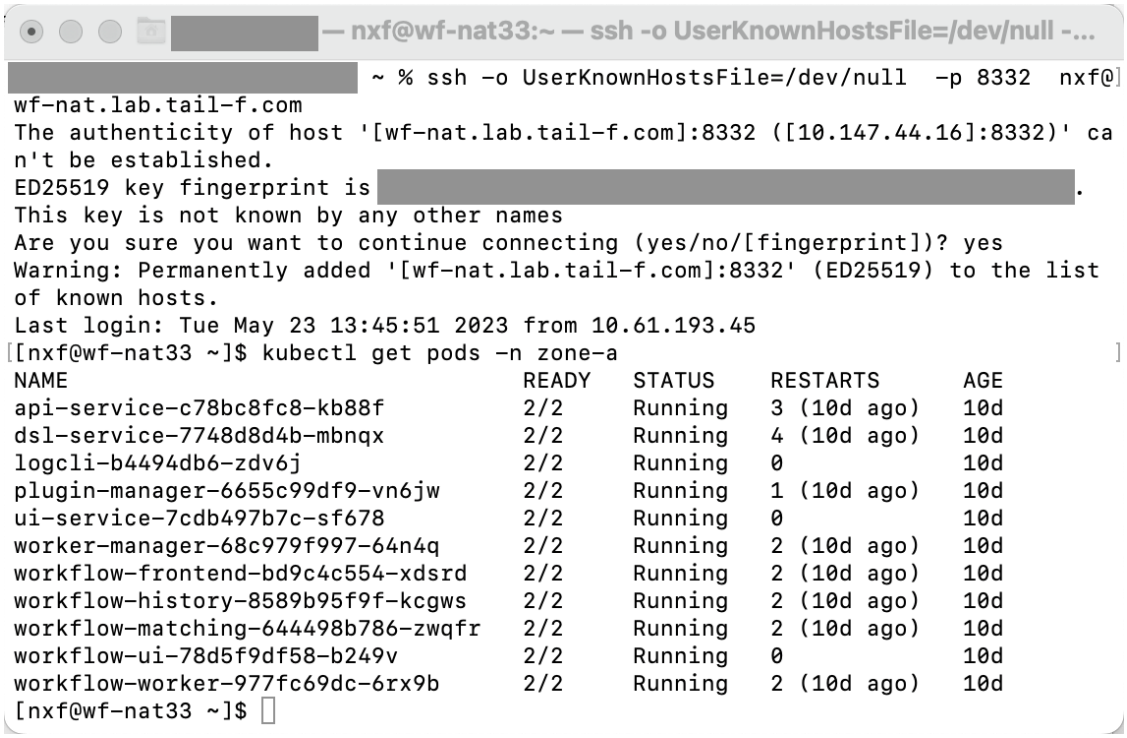
```
ssh -o UserKnownHostsFile=/dev/null  -p 22  nxf@<your_resource_pool_address>
```

Step 2 To check status of pods for namespace `zone-a` (this is the default namespace for pods contaning CWM microservices), run the following command:

```
kubectl get pods -n zone-a
```

Step 3 A list of pods will appear:

Figure 13: Get k8s pods



Step 4 If a pod has a status different from `Running`, you can 'restart' it using the following command:

```
kubectl delete pod <pod_name> -n zone-a
```

The pod will be deleted, but as Kubernetes configuration is declarative, it will effectively recreate the deleted pod and rerun it.

Check and collect logs

Application logs can be checked with **Loki logCLI** command-line interface. To gather logs from the CWM platform, follow these steps:

Step 1 Using a command-line terminal, connect to the system using SSH client:

```
ssh  -pSSH_PORT nxf@ip_address_of_deployment
```

Note Adjust `SSH_PORT` and `ip_address_of_deployment` accordingly.

Step 2 After successful login, use the command below to list all running pods:

```
kubectl get pods -A
```

Example result:

```
[nxf@wf-nat-08 ~]$ kubectl get pods -A
NAMESPACE            NAME                                          READY   STATUS      RESTARTS
AGE
```

```
kube-flannel         kube-flannel-ds-trr95                     1/1     Running     0
103m
kube-system          coredns-htg9j                             1/1     Running     0
103m
kube-system          etcd-wf-nat-08                            1/1     Running     0
103m
kube-system          kube-apiserver-wf-nat-08                  1/1     Running     0
103m
kube-system          kube-controller-manager-wf-nat-08         1/1     Running     0
103m
kube-system          kube-proxy-c25f5                          1/1     Running     0
103m
kube-system          kube-scheduler-wf-nat-08                  1/1     Running     0
103m
local-path-storage   local-path-provisioner-6fb6f599c7-ckcjc   1/1     Running     0
103m
nxf-system           authenticator-5db8885675-qlrmg            2/2     Running     0
102m
nxf-system           controller-cbd87f8c5-6tg6f                2/2     Running     1 (102m ago)
102m
nxf-system           ingress-proxy-56f7c9899d-6st6j            1/1     Running     0
102m
nxf-system           kafka-0                                   1/1     Running     0
102m
nxf-system           loki-7c994678f8-fnrs9                     3/3     Running     0
102m
nxf-system           minio-0                                   2/2     Running     0
103m
nxf-system           postgres-0                                2/2     Running     0
102m
nxf-system           promtail-v6tb4                            1/1     Running     0
102m
nxf-system           registry-7dd84db44f-n5q7h                 2/2     Running     0
102m
nxf-system           vip-wf-nat-08-28131000-772k5              0/1     Completed   0
3m42s
zone-a               api-service-745759bffc-v6r25              2/2     Running     2 (100m ago)
100m
zone-a               dsl-service-77d5fc96cc-5nv42              2/2     Running     3 (100m ago)
100m
zone-a               logcli-5c7ddbc95d-mkpcc                   2/2     Running     0
100m
zone-a               plugin-manager-665b7bbd4d-jvqdk           2/2     Running     1 (100m ago)
100m
zone-a               ui-service-57cf6d6bcc-smmvt               2/2     Running     0
100m
zone-a               worker-manager-6d6b445d46-r6nzk           2/2     Running     1 (99m ago)
100m
zone-a               workflow-frontend-77bc897549-kcz5k        2/2     Running     1 (99m ago)
100m
zone-a               workflow-history-58bdb85b8d-88t25         2/2     Running     1 (99m ago)
100m
zone-a               workflow-history-58bdb85b8d-h22bd         2/2     Running     1 (99m ago)
100m
zone-a               workflow-history-58bdb85b8d-ph5fh         2/2     Running     1 (99m ago)
100m
zone-a               workflow-matching-86cfc5577c-4mxhb        2/2     Running     1 (99m ago)
100m
zone-a               workflow-ui-68f857645-9mq9v               2/2     Running     0
100m
zone-a               workflow-worker-8496898f7b-wcrqs          2/2     Running     1 (99m ago)
100m
```

Step 3 Identify the logcli tool available in the `zone-a` namespace. In this example, it is the pod named `logcli-5c7ddbc95d-mkpcc`.

Step 4 Connect to the correct pod and list the available log labels for filtering:

```
kubectl exec --namespace=zone-a -ti logcli-5c7ddbc95d-mkpcc -- logcli labels
app
container
filename
level
namespace
node_name
pod
stream
```

Step 5 Gather logs from all applications running in the `zone-a` namespace and save them to a single file. Make sure to adjust the `--since` option to collect logs from the relevant time period when the troubleshooting event occurred:

```
kubectl exec --namespace=zone-a -ti logcli-5c7ddbc95d-mkpcc -- logcli query '{namespace="zone-a"}'
--since 60m > zone-a.log
```

Step 6 Similarly, collect logs from other namespaces, using different files for convenience:

```
kubectl exec --namespace=zone-a -ti logcli-5c7ddbc95d-mkpcc -- logcli query '{namespace="nxf-system"}'
 --since 60m > nxf-system.log
```

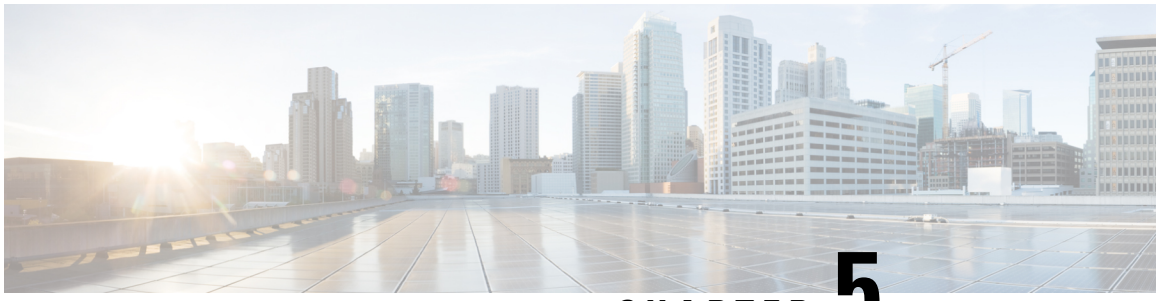
```
kubectl exec --namespace=zone-a -ti logcli-5c7ddbc95d-mkpcc -- logcli query '{namespace="kube-system"}'
 --since 60m > kube-system.log
```

Step 7 Use the SCP tool to copy the log files from the system to your desktop:

```
scp -P SSH_PORT nxf@ip_address_of_deployment:"*.log".
```

Step 8 Finally, you can send the logs to support and provide a detailed description of the issue you are experiencing.

Note For more details on the logCLI commands and usage, refer to logCLI Grafana documentation.



CHAPTER 5

CWM API Postman collection

This section contains the following topics:

CWM API Postman collection

The CWM API was developed according to the Representational State Transfer (REST) design principles. The API is accessed using HTTP with JSON data format. The success or failure of the request is indicated by the relevant HTTP response code. Data retrieval methods require a GET request, while methods for adding, changing, or deleting data require POST, PUT, PATCH, or DELETE methods. Errors will be returned if the request is sent with the wrong request type. Built directly into the product is a Swagger interface accessed from the CWM UI, but for ease of use, a Postman collection with example requests is also provided.

Use API Postman collection

Import collection and set environment:

Before you begin

- Create Postman Web app account or install Postman Desktop.
- Download the Postman collection in JSON format by clicking this link. Unpack the zip archive.

Step 1 Open Postman and go to **Collections**.

Step 2 Click **Import**, select **folders** from the **Drop anywhere to import** screen and point to the folder that you have unpacked from the zip archive.

Step 3 Go to **Environments** and select the newly imported **test** environment.

Step 4 Provide current values for the **baseUrl** and **port** variables to fit your CWM IP address and port and save the changes.

Now you're set up and ready to use the collection.