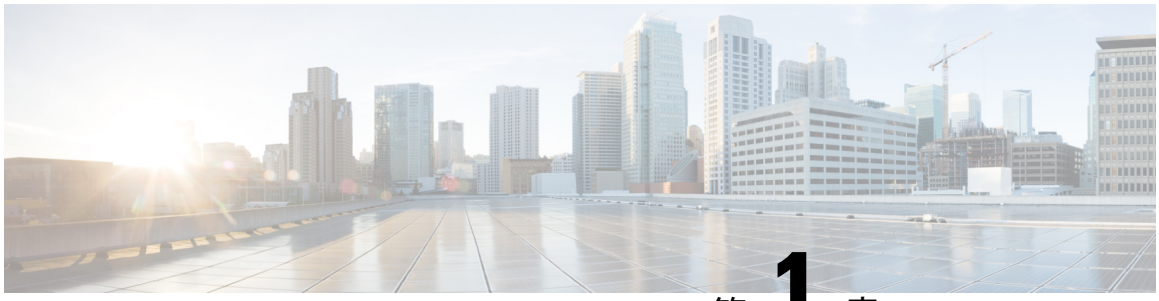
Cisco Crosswork Workflow Manager 1.0 管理者ガイド

初版：2023 年 6 月 1 日

シスコシステムズ合同会社
〒107-6227 東京都港区赤坂9-7-1 ミッドタウン・タワー
http://www.cisco.com/jp
お問い合わせ先：シスコ コンタクトセンター
0120-092-255 （フリーコール、携帯・PHS含む）
電話受付時間：平日 10:00～12:00、13:00～17:00
http://www.cisco.com/jp/go/contactcenter/



第 1 章

OVA を使用した CWM のインストール

ここでは、次の内容について説明します。

OVA を使用した CWM のインストール

Crosswork Workflow Manager 1.0 は、vSphere vCenter 7.0 仮想化プラットフォームを使用して OVA イメージを展開することにより、ゲスト仮想マシンとしてインストールされます。

前提条件

• vSphere vCenter 7.0 アカウントと ESXi 7.0 ホスト。

CWM パッケージのダウンロード

始める前に

Crosswork Workflow Manager 1.0 ソフトウェアパッケージを取得するには、次の手順を実行します。

ステップ 1 シスコ ソフトウェア ダウンロード サービスに移動し、検索バーに「**Crosswork Workflow Manager**」と入力し、検索リストからこれを選択します。

ステップ 2 [ソフトウェアタイプの選択（Select a software type）] から、[Crosswork Workflow Manager ソフトウェア（Crosswork Workflow Manager Software）] を選択します。

ステップ 3 Linux 用の Crosswork Workflow Manager ソフトウェアパッケージをダウンロードします。

ステップ 4 ターミナルで、自己解凍型の署名付きバイナリを実行します。実行すると `cwm-1.0.tar.gz` ファイルが抽出され、署名ファイルを使用して検証されます。

ステップ 5 `cwm-1.0.tar.gz` ファイルを抽出するには、ファイルをダブルクリックするか（Mac ユーザー）、gzip ユーティリティを使用します（Linux および Windows ユーザー）。これにより、CWM OVA ファイルが抽出されます。

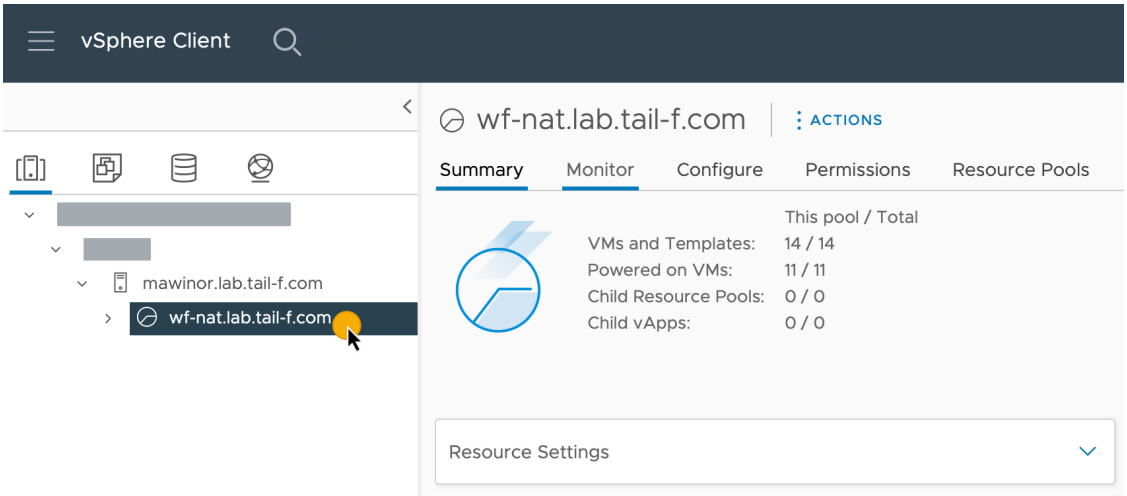
OVA の展開と VM の起動

ダウンロードした OVA イメージを使用して仮想マシンを作成するには、次の手順を実行します。

ステップ 1 vSphere アカウントにログインします。

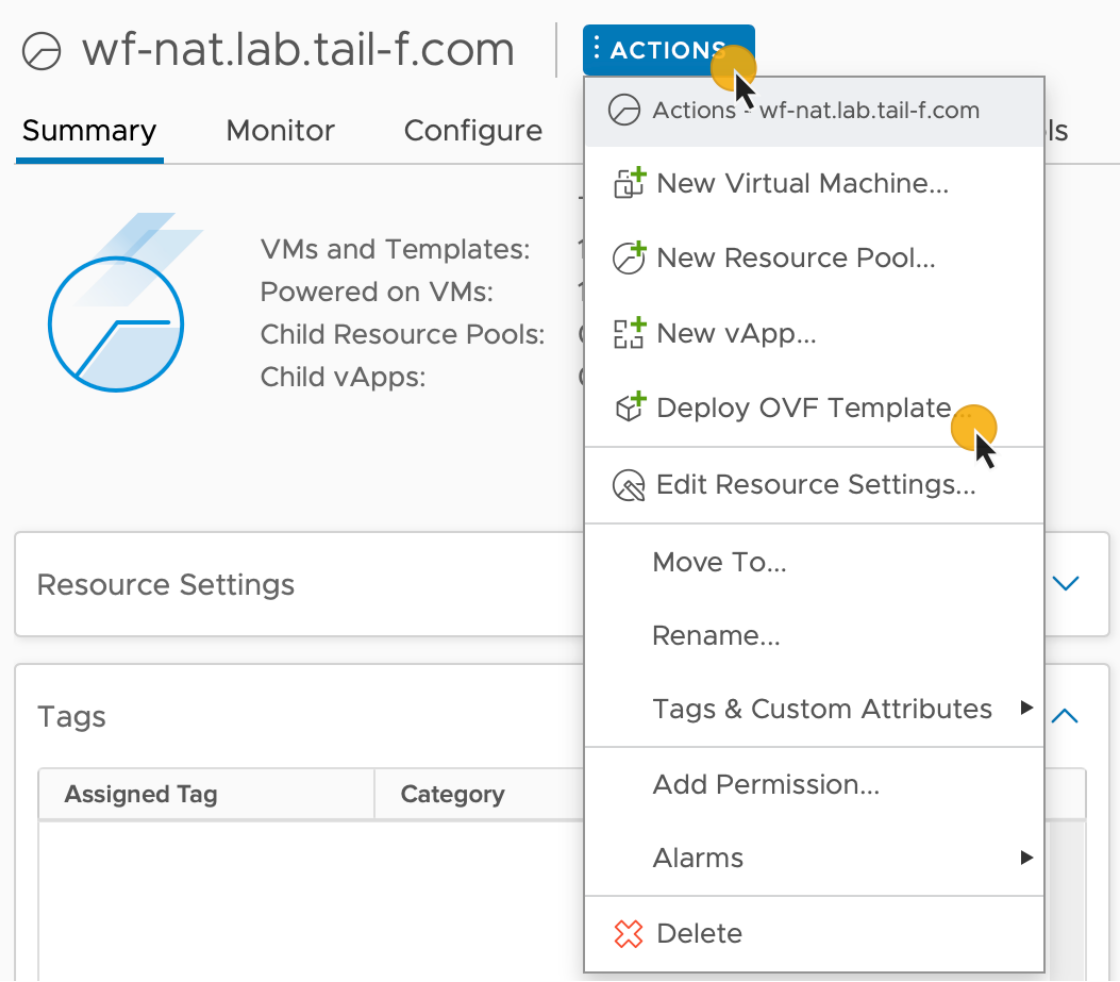
ステップ 2 [ホストおよびクラスタ（Hosts and Clusters）] タブで、ホストを展開してリソースプールを選択します。

図 *1*: リソースプール



ステップ 3 [アクション（Actions）] メニューをクリックし、[OVFテンプレートの展開（Deploy OVF Template...）] を選択します。

図 *2 : OVF* テンプレートの展開



ステップ 4 [OVF テンプレートの選択（Select an OVF template）] ステップで、[ローカルファイル（Local file）]、[ファイルの選択（Select files）] の順にクリックし、CWM OVA イメージを選択します。[次へ（Next）] をクリックします。

ステップ 5 [名前とフォルダの選択（Select a name and folder）] ステップで、VM の名前を入力して場所を選択します。[次へ（Next）] をクリックします。

ステップ 6 [コンピューティングリソースの選択（Select a compute resource）] ステップで、リソースプールを選択します。[次へ（Next）] をクリックします。

ステップ 7 [詳細の確認（Review details）] ステップで、[次へ（Next）] をクリックします。

ステップ 8 [ストレージの選択（Select storage）] ステップで、[仮想ディスクフォーマットの選択（Select virtual disk format）] を [シンプロビジョニング（Thin provision）] に設定し、ストレージを選択して、[次へ（Next）] をクリックします。

ステップ 9 [ネットワークの選択（Select network）] ステップで、**コントロールプレーン**と**ノースバウンド**の宛先ネットワークを選択する必要があります。

a) （注） コントロールプレーンの設定は、HA クラスタのセットアップの場合にのみ必要です。単一ノードのセットアップでは、コントロールプレーンの設定を指定する必要がありますが、必須ではなく、制御ネットワークに接続されている他のデバイスと競合しないようにする必要があります。

[コントロールプレーン（Control Plane）] : [プライベートネットワーク（PrivateNetwork）] を選択します。選択できない場合は、[VMネットワーク（VM Network）] を選択します。

b) [ノースバウンド（Northbound）] : [VMネットワーク（VM Network）] を選択します。

c) [次へ（Next）] をクリックします。

ステップ 10 [テンプレートのカスタマイズ（Customize template）] ステップで、次の選択されたプロパティを指定します。

a) [インスタンスのホスト名（Instance Hostname）] : インスタンスの名前を入力します。

b) [SSH公開キー（SSH Public Key）] : VM へのコマンドラインアクセスに使用される SSH 公開キーを指定します。

c) [コントロールプレーンノード数（Control Plane Node Count）] : HA クラスタセットアップの場合にのみ、1 以上に変更します。CWM バージョン 1.0 ではサポートされていません。

d) [コントロールプレーンIP（Control Plane IP）] : コントロールプレーンのネットワークアドレスを指定します。このアドレスは、制御ネットワーク内の他のデバイスと競合するこはできません。ただし、単一ノードのセットアップでは必須ではありません。

e) [イニシエータIP（Initiator IP）] : スターターノードのイニシエータIPを設定します。単一ノードセットアップでは、[コントロールプレーンIP（Control Plane IP）] * と同じアドレスです。

f) [IP]（DHCP を使用していない場合）：ノードのネットワークアドレスを指定します。

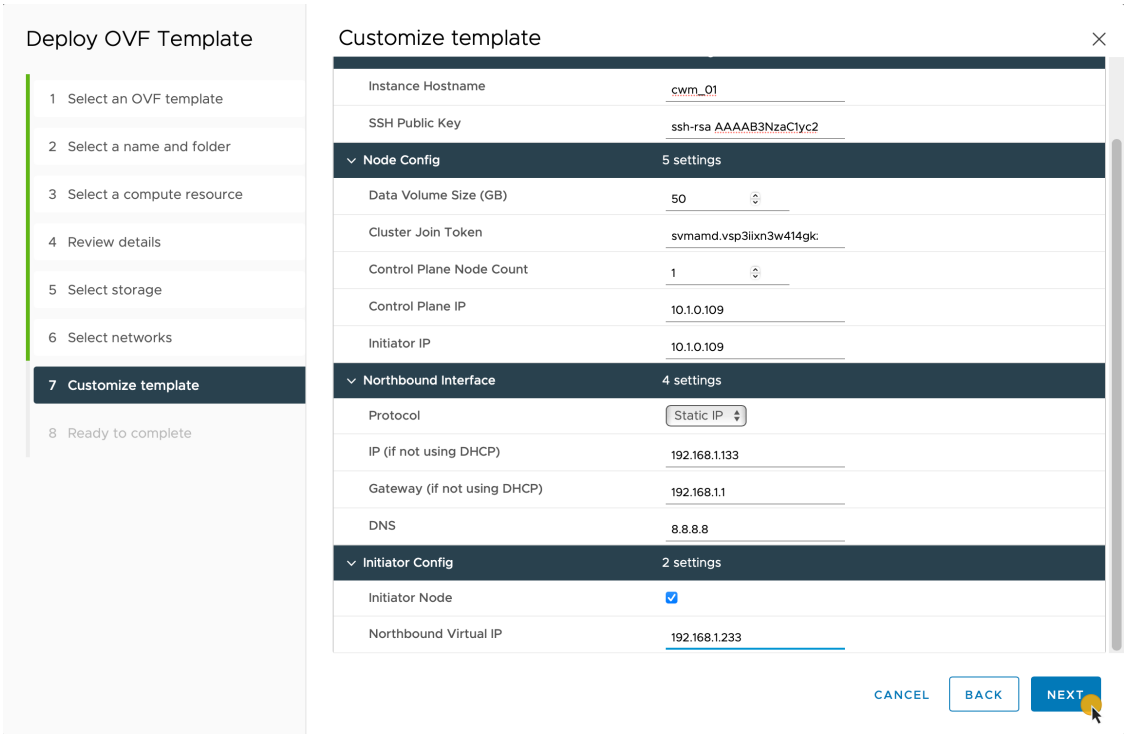
g) [ゲートウェイ（Gateway）]（DHCP を使用していない場合）：ゲートウェイアドレスを指定します。デフォルトでは、`192.168.1.1` です。

h) [DNS] : DNS のアドレスを指定します。デフォルトでは `8.8.8.8` ですが、ローカル DNS を使用することもできます。

i) [ノースバウンド仮想IP（Northbound Virtual IP）] : アクティブなクラスタノードのネットワークアドレスを指定します。単一ノードのセットアップでも、このアドレスは必須です。このアドレスで HTTP サービスが機能するためです。

j) [次へ（Next）] をクリックします。

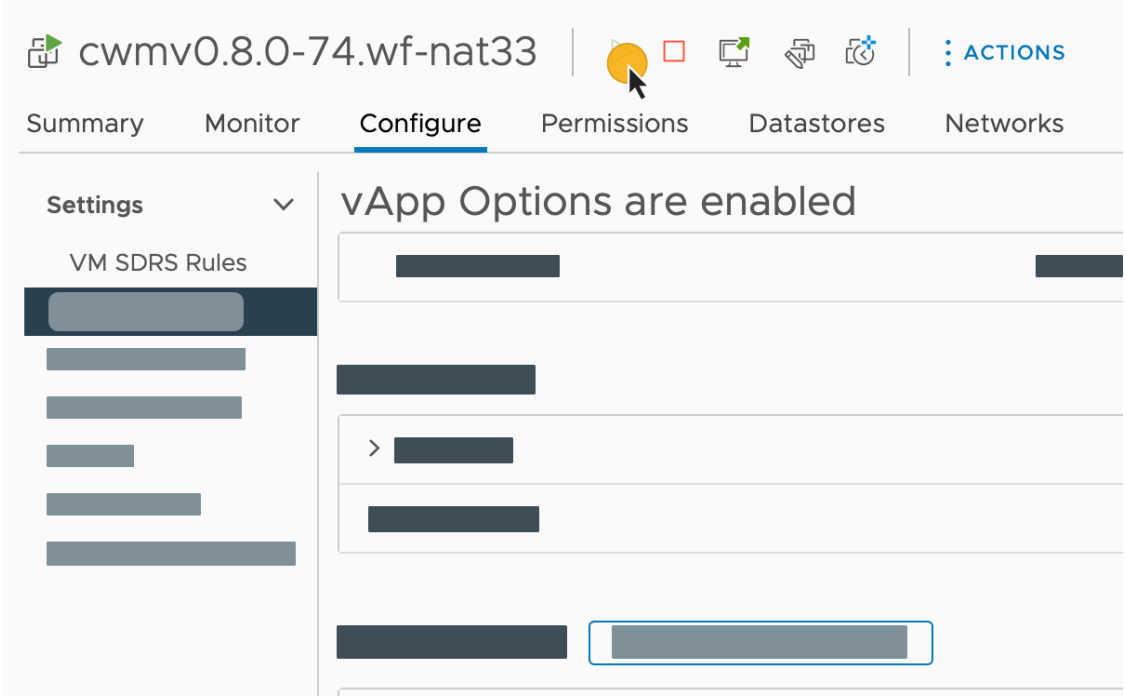
図 *3* : テンプレートのカスタマイズ



ステップ 11 [準備完了（Ready to Complete）] ページで [終了（Finish）] をクリックします。展開には数分かかる場合があります。

ステップ 12 [リソースプール（Resource pool）] リストから、新しく作成した仮想マシンを選択し、[電源オン（Power on）] アイコンをクリックします。

図 *4 : VM* の電源オン



（注） VM の電源が正常にオンにならない場合は、NxF が原因で断続的なインフラストラクチャ エラーが発生している可能性があります。回避策として、既存の VM を削除し、新しい VM に OVA を再展開します。

ユーザの作成

VM へのコマンドラインアクセスを使用して、CWM プラットフォームのユーザーアカウントを作成できます。その方法を次に説明します。

ステップ 1 コマンドラインターミナルを使用して、SSH でゲスト OS の NxF にログインします。

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

（注） SSH のデフォルトポートは 22 です。必要に応じてカスタムポートに変更してください。

a) オプション：初めてログインする場合は、秘密キーのパス名を入力します。

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

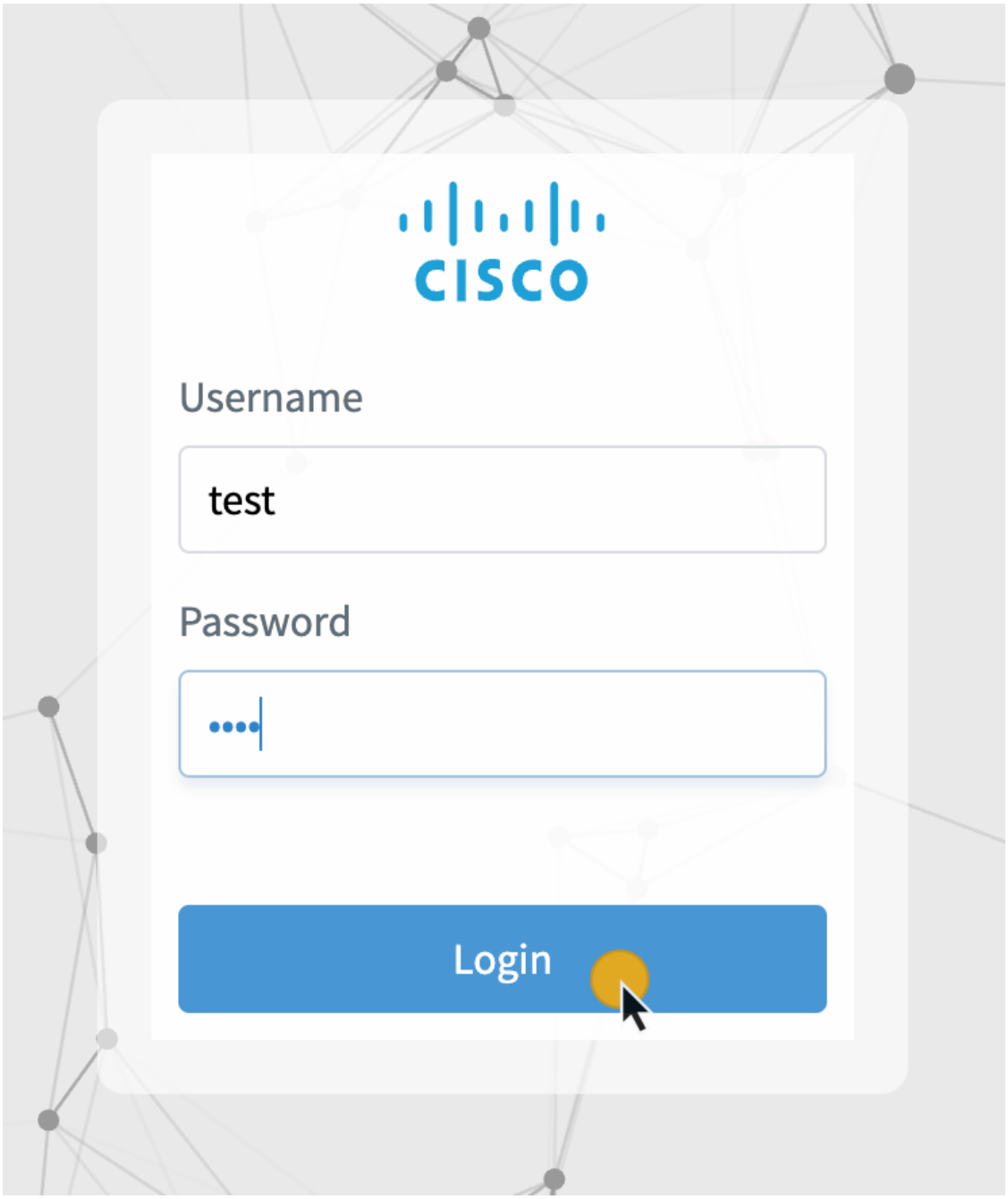
ステップ 2 ユーザーを作成するには、次のコマンドを実行します。

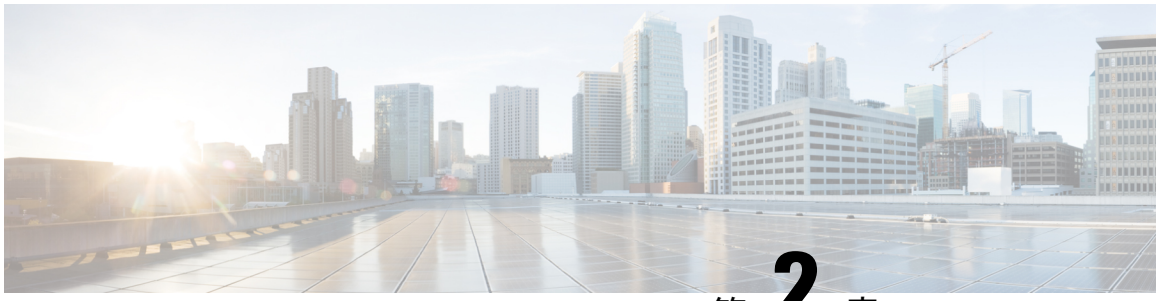
```
echo -en "test" | sedo security user add --password-stdin --access permission/admin --display-name
 Tester test
```

ステップ 3 ノードに選択したアドレスとデフォルトのポート `8443` に移動します。例：https://wf.lab.cisco.com:8443/。

ステップ 4 `test` ユーザー名とパスワードを使用してログインします。

図 *5:* ログイン





第 2 章

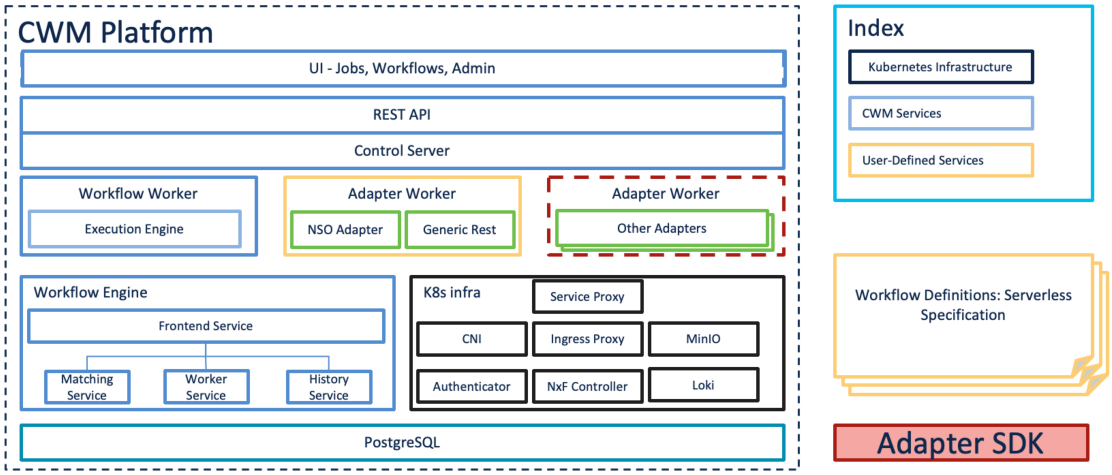
アーキテクチャの概要

ここでは、次の内容について説明します。

アーキテクチャの概要

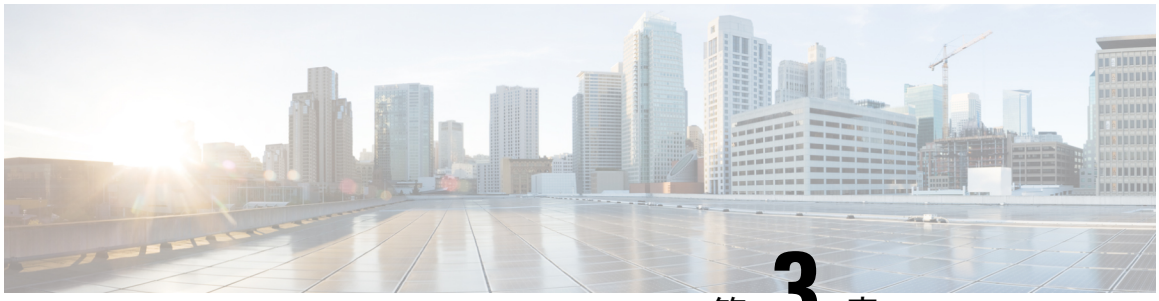
Crosswork Workflow Manager アーキテクチャは、Kubernetes コンテナ オーケストレーション システム上で動作するマイクロサービスベースのソリューションです。このセクションでは、コア アーキテクチャ コンポーネントを示す図と、それぞれの簡単な説明を示します。

図 *6 :* アーキテクチャの概要



- **ユーザーインターフェイス（UI）**：オペレータは、ワークフローの追加とインスタンス化、ワークフローデータの入力、実行中のワークフローの一覧表示、ジョブの進行状況の監視を行うことができます。UI の [管理（Admin）] セクションでは、ワーカーの追加、ワーカープロセスの管理、およびアダプタからワーカーへのアクティビティの割り当てを行うことができます。

- **REST API**：CWM アプリケーションとのすべての連携（アダプタの展開、ワークフローの公開とインスタンス化、ワーカー、リソース、およびシークレットの管理）が含まれます。
- **制御サーバー**：関連するマイクロサービスに API 要求をディスパッチします。
- **ワークフローエンジン**：ワークフローの処理方法を制御するコアコンポーネントです。ワークフロー定義の実行を解釈および管理します。
- **実行エンジン（ワークフローワーカー）**：ワークフロータスクの実行を担当します。**ワークフローエンジン**からワークフロータスクを受信し、正しい順序で実行し、結果を**ワークフローエンジン**に返します。
- **アダプタワーカー**：ワークフロー定義とアダプタコードで定義されたタスクの実行を担うプロセスです。**ワークフローワーカー**からタスクを受信して実行し、結果を**ワークフローワーカー**に送り返します。実行ワーカーは、追加のアダプタをプラグインとしてロードできるため、さまざまなシステムやテクノロジーと連携できます。
- **アダプタ**：外部システム、アプリケーション、およびテクノロジーとのインターフェイスとなり、これらと統合します。アダプタ内部では、ワークフローで使われるアクティビティが定義されます。
- **アダプタ SDK**：外部システムと統合するための新しいアダプタを作成する開発者を支援するソフトウェア開発キット。
- **ワークフロー定義**：サーバレスワークフロー仕様に基づいて JSON 形式で記述されたワークフローコード。
- **K8s インフラストラクチャ**：CWM アプリケーション用のランタイムプラットフォーム。これは、Kubernetes クラスタ内のアプリケーションの展開と管理をサポートするために必要なインフラストラクチャを提供するサービスの集合です。
- **PostgreSQL**：データを保存および管理するためにシステムで使用されるデータベースです。



第 3 章

管理

ここでは、次の内容について説明します。

アダプタの管理

外部ターゲットシステムと連携するには、CWM にアダプタが必要です。アダプタは、CWM API を使用して管理できます。アダプタの処理には、次の API エンドポイントを使用できます。

- `GET/adapter`：CWM アプリケーションに存在するアダプタのリストを取得します。
- `POST/adapter`：アダプタの **.tar** ファイルを CWM ストレージにアップロードします。
- `GET/adapter/{adapterId}`：CWM アプリケーションに存在する特定のアダプタの詳細を取得します。これには、アダプタで使用可能なすべてのアクティビティの一覧表示が含まれます。
- `PUT/adapter/{adapterId}`：既存のアダプタファイルを新しいアダプタバージョンで更新します。
- `DELETE/adapter/{adapterId}`：CWM アプリケーションからアダプタを削除します。
- `POST/adapter/{adapterId}/deploy`：アップロードされたアダプタファイルに基づいてシステムにアダプタを展開します。

アダプタのインストール

CWM アダプタは、**.tar** インストールファイルに含まれています。ワークフローで使用するには、事前にストレージにアップロードしてシステムに展開する必要があります。その方法について説明します。

アダプタファイルのアップロード

アダプタを展開する前に、アダプタの **.tar** ファイルを CWM ストレージにアップロードする必要があります。

ステップ 1 最新のアダプタ インストール ファイルを取得するか、独自のアダプタを作成します。

ステップ 2 CWM にログインし、左側のナビゲーションメニューから [Swagger] アイコンをクリックします。

ステップ 3 [adapters] セクションで、`POST/adapter` エンドポイントをクリックして展開します。エンドポイント内で、[試す（Try it out）] をクリックします。

ステップ 4 表示されるサブセクションで、[ファイルの選択（Choose File）] をクリックし、アダプタの **.tar** インストールファイルを選択して [アップロード（Upload）] をクリックし、[実行（Execute）] をクリックします。

サーバーの応答コードが `201` の場合、アダプタファイルは CWM データベースに正常にアップロードされています。

アダプタの展開

ステップ 1 CWM API の [アダプタ（adapters）] セクションで、`GET/adapter` エンドポイントをクリックして展開します。エンドポイント内で、[試す（Try it out）] と[実行（Execute）] をクリックします。

ステップ 2 サーバーの応答本文から、アップロードしたアダプタの `id` フィールドの値をコピーします。

ステップ 3 CWM API の [アダプタ（adapters）] セクションで、`POST/adapter/{adapterId}/deploy` エンドポイントをクリックして展開します。

ステップ 4 エンドポイント内で、[試す（Try it out）] をクリックします。[アダプタID（Adapter ID）] フィールドにアダプタ ID を貼り付けます。

ステップ 5 [createWorker] フィールドで、`createWorker` パラメータを `true` に設定できます。これにより、アダプタ ID と同じ名前のワーカーが作成されます。

ステップ 6 [実行（Execute）] をクリックします。

サーバーの応答コードが `201` の場合、アダプタプラグインは正常にインストールされています。

アダプタの削除

アダプタをストレージから完全に削除して「アンインストール」するには、次の手順を実行します。

ステップ 1 CWM API の [アダプタ（adapters）] セクションで、`GET/adapter` エンドポイントをクリックして展開します。エンドポイント内で、[試す（Try it out）] と[実行（Execute）] をクリックします。

ステップ 2 サーバーの応答本文から、アップロードしたアダプタの `id` フィールドの値をコピーします。

ステップ 3 CWM API の [アダプタ（adapters）] セクションで、`DELETE/adapter/{adapterId}` エンドポイントをクリックして展開します。

ステップ 4 エンドポイント内で、[試す（Try it out）] をクリックします。[アダプタID（Adapter ID）] フィールドにアダプタ ID を貼り付けます。

ステップ 5 [実行（Execute）] をクリックします。

ワーカーの管理

ワーカーは、ワークフロー定義とアダプタコードで定義されたアクションを実行するプロセスです。**オペレータ**ガイドで説明されているように CWM UI を使用してワーカーを管理できます。または以下で説明しているように CWM API を使用して管理できます。

ワーカーを管理するための次のアクションを使用できます。

- `GET/worker`：CWM アプリケーションに存在するワーカーのリストを取得します。
- `POST/worker`：CWM アプリケーション内に新しいワーカーを作成します。
- `GET/worker/{workerName}`：CWM アプリケーションに存在する特定のワーカーの詳細を取得します。
- `PUT/worker/{workerName}`：既存のワーカーを新しいパラメータ値で更新します。
- `DELETE/worker/{workerName}`：CWM アプリケーションからワーカーを削除します。
- `POST/worker/{workerName}/start`：アプリケーションで作成されたワーカーをアクティブにします。
- `POST/worker/{workerName}/stop`：アプリケーションで作成されたワーカーを非アクティブ化します。

ワーカーの作成

ステップ 1 CWM にログインし、左側のナビゲーションメニューから [Swagger] アイコンをクリックします。

ステップ 2 CWM API の [ワーカー（workers）] セクションで、POST/worker エンドポイントをクリックして展開します。エンドポイント内で、[試す（Try it out）] をクリックします。

ステップ 3 [ワーカーデータ（Worker data）] フィールドに、必要な値を入力します。

a) [activities]：展開されたアダプタの ID または特定のアダプタアクティビティを貼り付けます。

b) [startWorker]：`true` に設定します。

c) [workerName] : ワーカーの名前を指定します。

ステップ 4 [実行（Execute）] をクリックします。

ワーカーの開始

ステップ 1 CWM API の [ワーカー（workers）] セクションで、`POST/{workerName}/start` エンドポイントをクリックして展開します。エンドポイント内で、[試す（Try it out）] をクリックします。

ステップ 2 次の**パラメータ**のフィールドに、必要な値を入力します。

a) [開始するワーカーの名前（Name of a worker to start）] : 開始するワーカーの名前を貼り付けます。

b) [forceReload]: ワーカーを強制的に起動する場合は `true` に設定します。

ステップ 3 [実行（Execute）] をクリックします。

ワーカーの停止

ステップ 1 CWM API の [ワーカー（workers）] セクションで、`POST/{workerName}/stop` エンドポイントをクリックして展開します。エンドポイント内で、[試す（Try it out）] をクリックします。

ステップ 2 次の**パラメータ**のフィールドに、必要な値を入力します。

a) [停止するワーカーの名前（Name of a worker to stop）] : 停止するワーカーの名前を貼り付けます。

b) [forceStop]: ワーカーを強制的に停止する場合は `true` に設定します。

ステップ 3 [実行（Execute）] をクリックします。

ワークフローの管理

ワークフローインスタンスは、**オペレータ**ガイドで説明されているように CWM UI で、または CWM API を使用して管理できます。

- `POST/workflow`：CWM アプリケーション内に新しいワークフローインスタンスを作成します。
- `GET/workflow/list`：CWM アプリケーションに存在するアダプタのリストを取得します。

- `GET/workflow/{id}`：CWMアプリケーションに存在する特定のワークフローの詳細を取得します。
- `PUT/workflow/{id}`：既存のワークフローを新しいワークフロー定義で更新します。
- `DELETE/workflow/{id}`：選択したワークフローをCWMアプリケーションから削除します。

（注） ワークフローを管理するには、CWM UIを使用することが推奨されます。詳細については、オペレータガイドを参照してください。

リソースとシークレットの管理

CWMでは、アダプタは、他のシステムやアプリケーションなどの外部エンティティでアクションを実行できるようにするアクティビティを定義します。これらのエンティティは、ほとんどの場合、通常は接続と認証データを必要とするAPIを介して統合されます。CWMは、アクティビティがワークフローで使用されるときに、接続エンドポイントの詳細と認証データを実行時に渡すことができるフレームワークを提供します。したがって、ワークフローを実行するオペレータは、IPアドレス、ポート、ユーザー名、パスワードなど、これらのシステム（リソース）の詳細を知らない場合があります。

CWMは、データベース内のリソースとシークレットを安全に処理し、それぞれのIDでそれらを識別するためのフレームワークを提供します。ワークフローインスタンスを実行する場合は、リソースIDのみを渡す必要があり、残りのデータはリソースマネージャによってアダプタに送信されます。オペレータの介入やアダプタ開発者の追加開発は必要ありません。

リソースおよびシークレットのタイプ

リソースおよびシークレットのタイプは、ユーザーが作成したリソースとシークレットをタイプ別に整理するために使用される入れ物と考えることができます。タイプは特定のアダプタ内で定義され、アダプタのインストール時に自動的にシステムに追加されます。
`GET/secret/type/{type}` APIエンドポイントを使用して、特定のタイプに属するシークレットを一覧表示できます。

シークレットAPIエンドポイント

シークレットを管理するための次のアクションを使用できます。

- `GET/secret`：CWMアプリケーションに存在するシークレットのリストを取得します。
- `POST/secret`：CWMアプリケーション内に新しいシークレットを作成します。
- `GET/secret/type/{type}`：CWMアプリケーションに存在する、特定のタイプに属するシークレットを一覧表示します。

- `GET/secret/types`：CWM アプリケーションに存在するシークレットのタイプのリストを取得します。
- `GET/secret/{id}`：既存のシークレットの詳細を取得します。
- `DELETE/secret/{id}`：CWM アプリケーションからシークレットを削除します。
- `PATCH/secret/{id}`：CWMアプリケーションに存在するシークレットを新しいパラメータ値で更新します。

リソース API エンドポイント

リソースを管理するための次のアクションを使用できます。

- `GET/resource`：CWM アプリケーションに存在するリソースのリストを取得します。
- `POST/resource`：CWM アプリケーションに新しいリソースを作成します。
- `GET/resource/{resourceId}`：CWM アプリケーションに存在する特定のリソースの詳細を取得します。
- `PUT/resource/{resourceId}`：既存のリソースを新しいパラメータ値で更新します。
- `DELETE/resource/{resourceId}`：CWM アプリケーションからリソースを削除します。
- `GET/resourceType`：CWM アプリケーションに存在するリソースタイプのリストを取得します。
- `GET/resourceType/{resourceId}`：既存のリソースタイプの詳細を取得します。

シークレットの作成

ステップ 1 CWM にログインし、左側のナビゲーションメニューから [Swagger] アイコンをクリックします。

ステップ 2 CWM API の [シークレット（secrets）] セクションで、[POST /secret] エンドポイントをクリックして展開します。

ステップ 3 エンドポイント内で、[試す（Try it out）] をクリックし、[シークレット入力（Secret input）] フィールドにデータを入力します。入力の例を次に示します。

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

ステップ 4 [実行（Execute）] をクリックします。

サーバーの応答コードが 201 の場合、シークレットは正常に作成されており、シークレットを関連付けるリソースの作成を開始できます。

リソースの作成

ステップ 1 CWM API の [リソース（resources）] セクションで、POST /resource エンドポイントをクリックして展開します。

ステップ 2 エンドポイント内で、[試す（Try it out）] をクリックし、[リソース入力（Resource input）] フィールドにデータを入力します。入力の例を次に示します。

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

ステップ 3 [実行（Execute）] をクリックします。

サーバー応答コードが 201 の場合、リソースは正常に作成されています。

NxF を介したユーザーアクセス

CWM では、NextFusion（NxF）を介してユーザーアクセスと権限を管理できます。NxF はセキュリティの追加レイヤを追加し、単一認証エージェントとして機能するため、ローカル、LDAP、および SAML の各ユーザーを共有します。

ユーザー、ロールおよび権限

現在、1 つのロールと権限タイプ（管理者）のみがサポートされています。すべてのユーザーは、デフォルトで管理者権限に関連付けられています。

通常のユーザーで構成される大きなグループに CWM へのアクセスを許可するには、環境に応じて、LDAP または SAML SSO プロトコル（両方を同時に使用できます）を介したユーザー認証を設定します。

権限の範囲

管理者ロールには、CWMとそのすべての機能へのフルアクセス権があります。管理者は、ユーザーのアクセスと権限を制御できます。管理者権限を持つすべてのローカルユーザーは、必要に応じて新しいユーザーを作成できます。

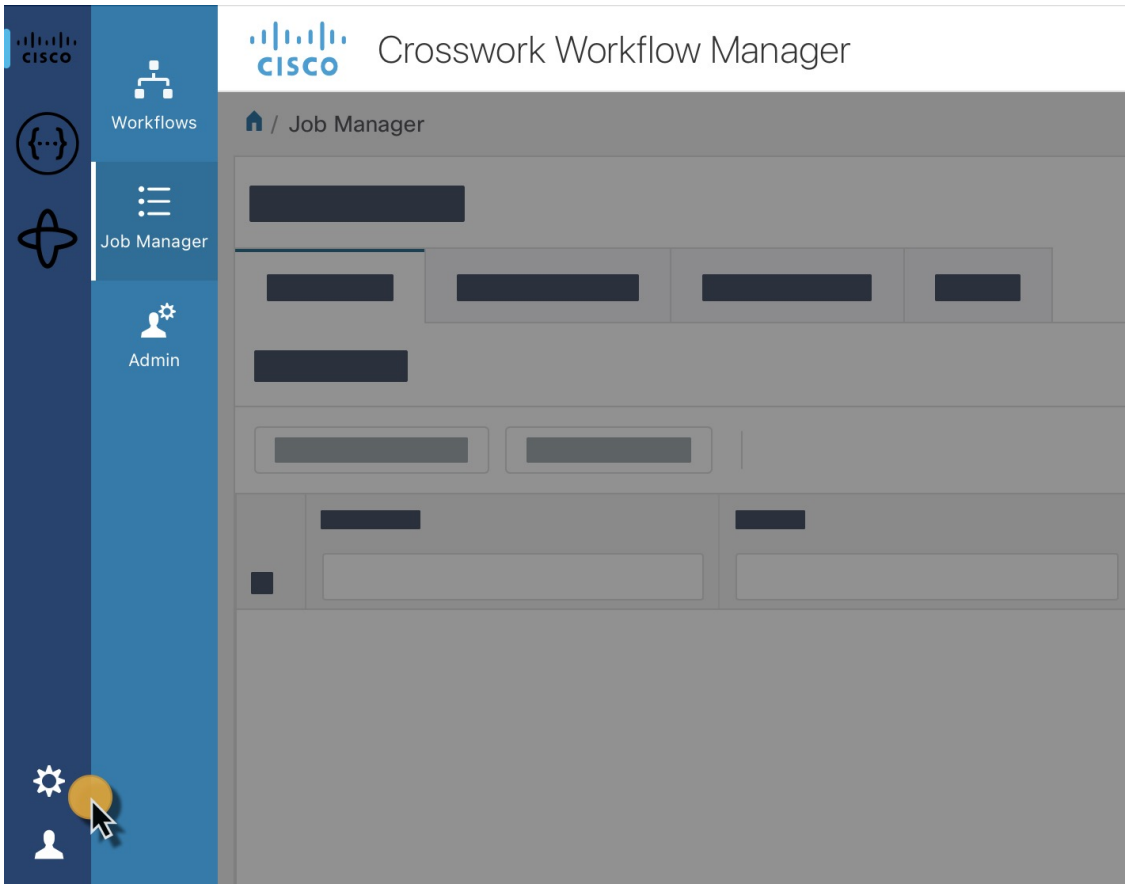
CWM 内の NxF 機能

NxF 機能は、管理者ユーザーが CWM UI の [設定（Settings）] タブから使用できます。CWM の NxF 機能にアクセスするには、次の手順を実行します。

ステップ 1 CWM で、左端のナビゲーションメニューに移動します。

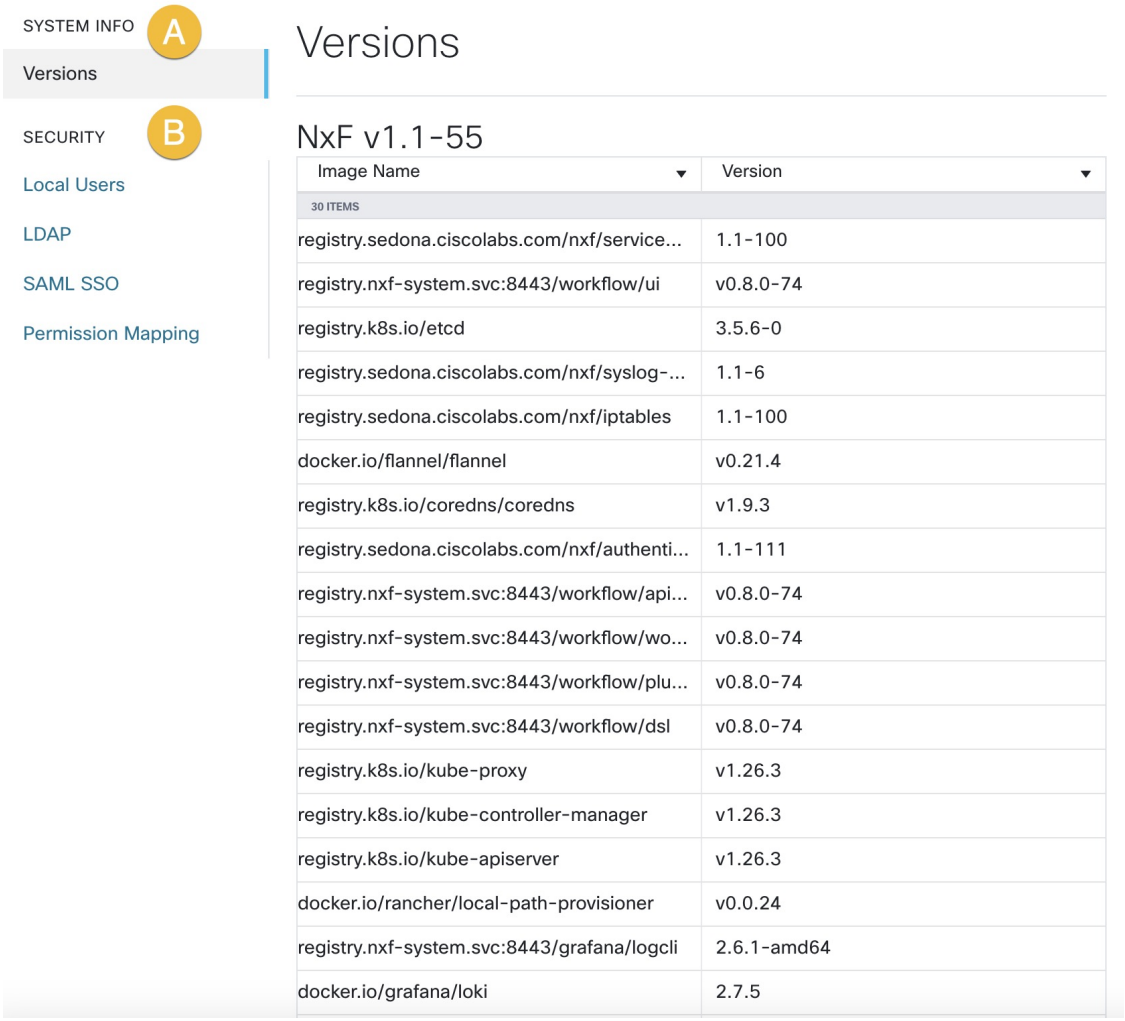
ステップ 2 [設定（Settings）] アイコン（歯車アイコン）をクリックします。

図 *7: NxF* の設定



ステップ 3 展開されたドロワに、次の項目が表示されます。

図 *8* : 設定のドロワ



a) [システム情報（System Info）] セクションには、NxF および CWM マイクロサービスの最新バージョンに関する情報が表示されます。

b) [セキュリティ（Security）] セクションには、アクセス管理に関する次の項目が表示されます。

- [ローカルユーザー（Local Users）]：UI を介してローカルユーザーを表示、作成、および編集できます。
- [LDAP]：ユーザー認証の LDAP 設定を構成できます。
- [SAML SSO]：ユーザー認証の SAML シングルサインオン設定を構成できます。
- [権限マッピング（Permission Mapping）]：シスコ ポリシー管理ツールを使用して権限管理を操作できます。

ローカルユーザーの追加

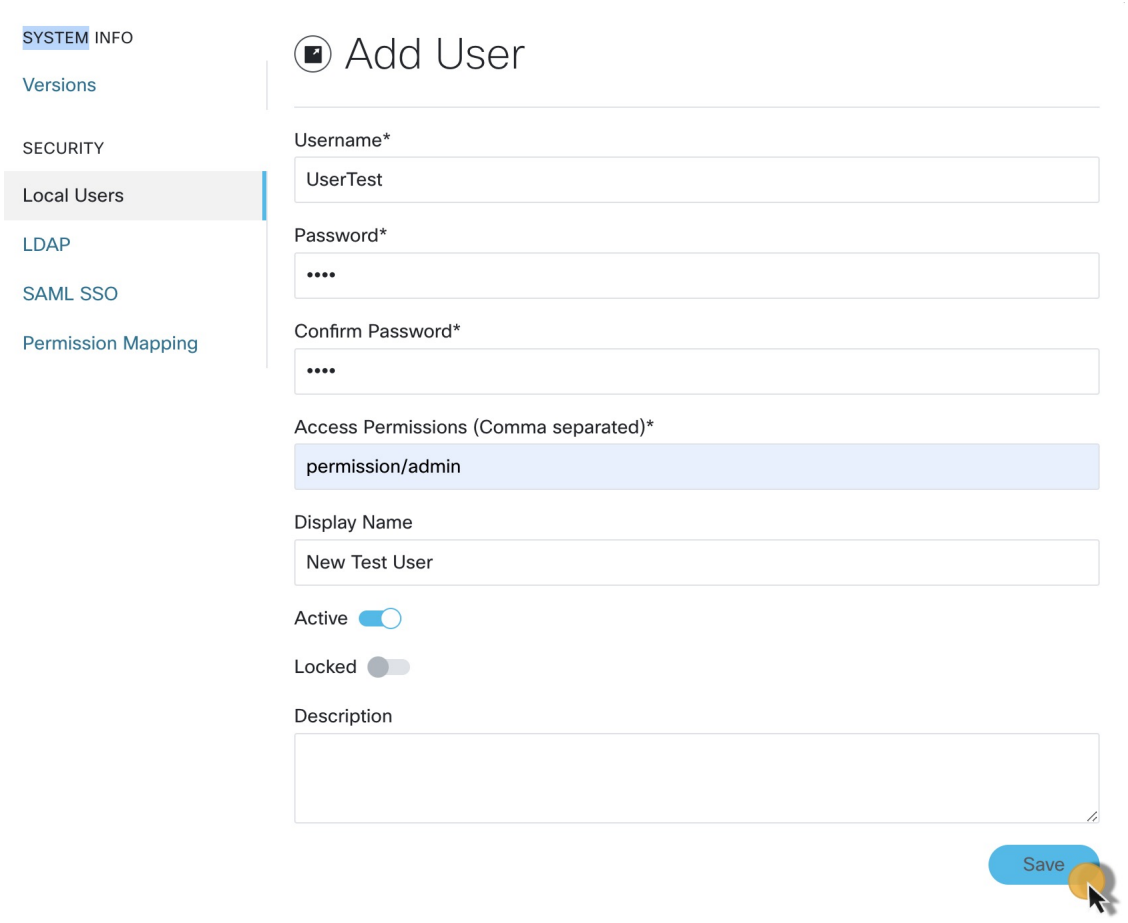
ステップ 1 CWM で、左端のナビゲーションメニューに移動します。

ステップ 2 CWM（Cisco アイコン）から [ローカルユーザー（Local Users）] タブに移動します。

ステップ 3 [追加...（Add...）] をクリックします。

ステップ 4 [ユーザーの追加（Add User）] パネルで、必須フィールド（アスタリスクでマークされているフィールド）の [ユーザー名（Username）]（CWM へのログインに使用）、[パスワード（Password）]、[パスワードの確認（Confirm Password）]、[アクセス権限（Access Permissions）]（`permission/admin` と入力）に入力します。[説明（Description）] と [表示名（Display Name）]（CWM でユーザー名の横に表示される）はオプションのフィールドです。

図 *9 : NxF* ユーザーの追加



ステップ 5 オプションボタンを使用して、ユーザーステータスを設定します。両方のオプションボタンを同時に無効または有効にできます。

- [アクティブ有効（Active enabled）]：ユーザーは CWM にログインできます。

- [アクティブ無効（Active disabled）] : ユーザーは CWM へのログインが禁止されます。
- [ロック有効（Locked enabled）] ：ユーザーの削除を防止します。
- [Lロック無効（Locked disabled）] : ユーザーの削除を許可します。

ステップ 6 [保存（Save）] をクリックします。

LDAP を介した認証の設定

CWM では、ローカルユーザーのサポートに加えて、LDAP（Lightweight Directory Access Protocol）サーバーとの統合によって LDAP ユーザーを追加できます。

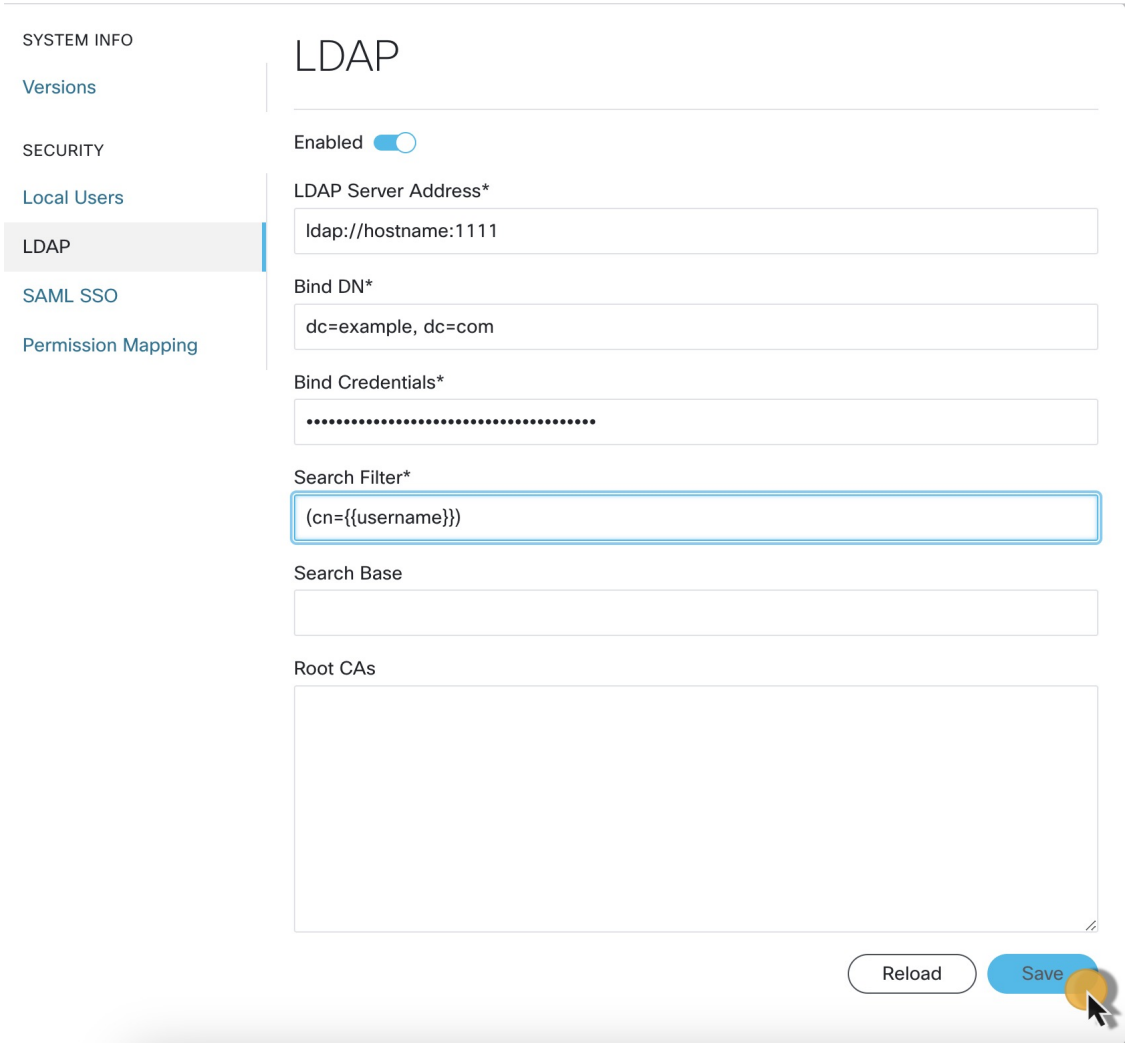
ステップ 1 CWM で、左端のナビゲーションメニューに移動します。

ステップ 2 CWM（Cisco アイコン）から、 [LDAP] タブに移動します。

ステップ 3 [有効（Enabled）] オプションボタンをクリックします。

ステップ 4 必須フィールド（アスタリスクでマークされているフィールド）の [LDAPサーバーアドレス（LDAP Server Address）]、[バインドDN（Bind DN）]、[バインドクレデンシャル（Bind Credentials）]、および [検索フィルタ（Search Filter）] に入力します。[検索ベース（Search Base）] と [ルートCA（Root CAs）] はオプションです。

図 *10 : LDAP*



ステップ 5 [保存（Save）] をクリックします。

SAML SSO を介した認証の設定

CWM は、SAML（セキュリティ アサーション マークアップ言語）プロトコルに基づいてシングルサインオンアクセスを取得するために、LDAP ユーザーと非 LDAP ユーザーの両方をサポートする SAML SSO 機能を提供します。CWM の SAML SSO は、LDAP と同時に、または LDAP なしで有効にできます。

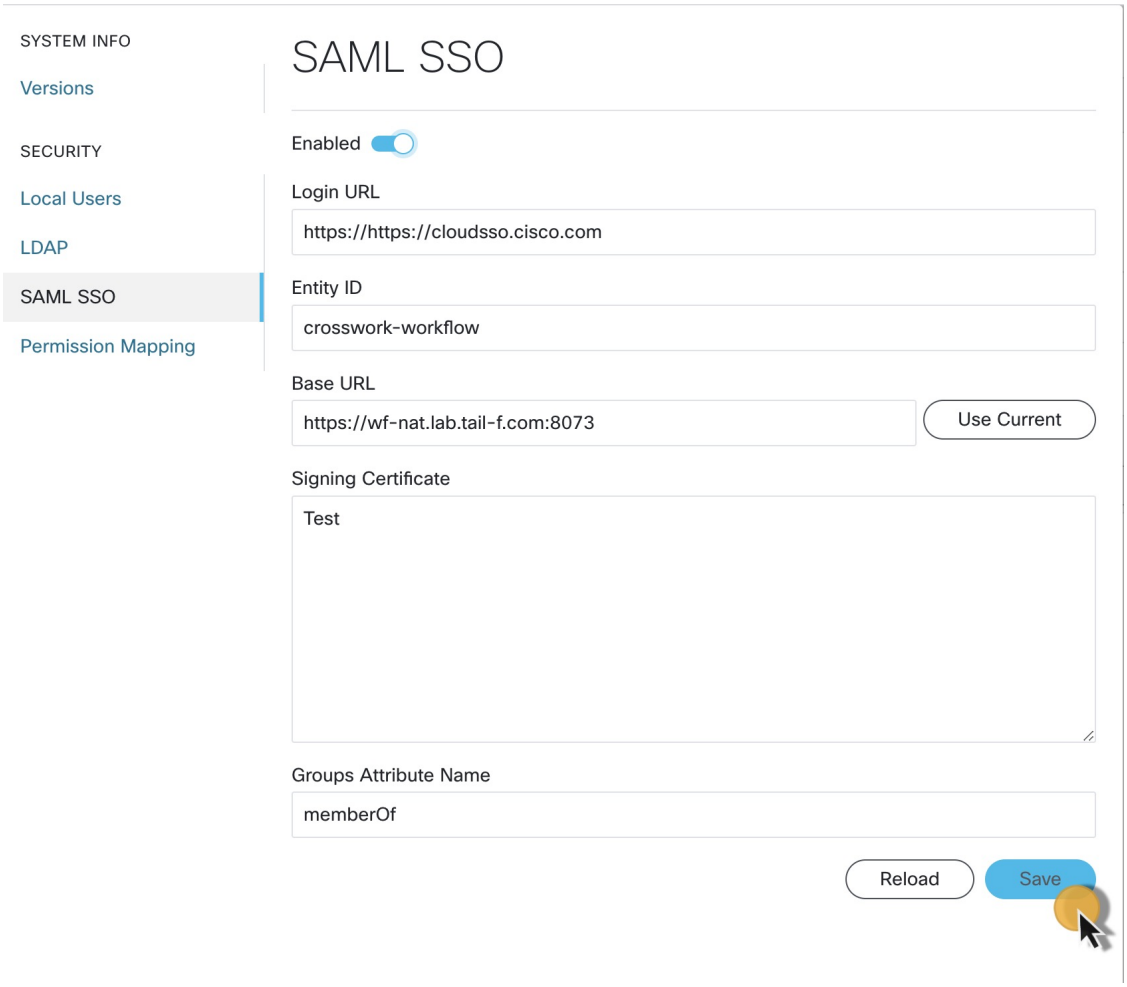
ステップ 1 CWM で、左端のナビゲーションメニューに移動します。

ステップ 2 CWM（Cisco アイコン）から [SAML SSO] タブに移動します。

ステップ 3 [有効（Enabled）] オプションボタンをクリックします。

ステップ 4 必須フィールド（[ログインURL（Login URL）]、[エンティティID（Entity ID）]、[ベースURL（Base URL）]、[署名証明書（Signing Certificate）]、および [グループ属性名（Groups Attribute Name）]）に入力します。

図 *11 : NxF SAMLSSO*



ステップ 5 [保存（Save）] をクリックします。

権限マッピングの設定

シスコ ポリシー管理ツール（PMT）を使用して、ユーザーのグループに特定の権限を付与できます。

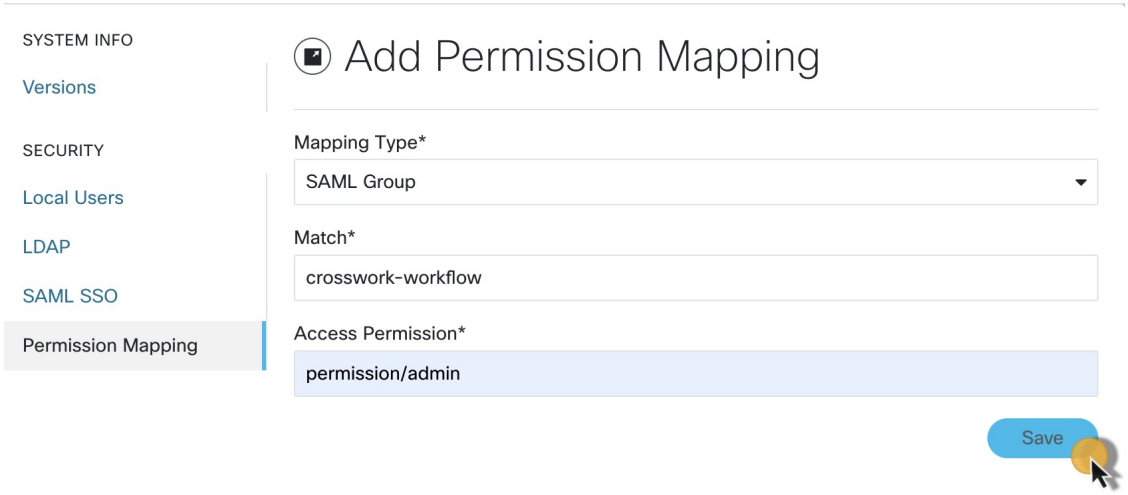
ステップ 1 CWM で、左端のナビゲーションメニューに移動します。

ステップ 2 CWM（Cisco アイコン）から [権限マッピング（Permission Mapping）] タブに移動します。

ステップ 3 [追加...（Add...）] をクリックします。.

ステップ 4 [権限マッピングの追加（Add Permission Mapping）] パネルで、ドロップダウンメニューからマッピングタイプ（SAML ユーザー、SAML グループ、LDAP ユーザー、または LDAP グループ）を選択します。

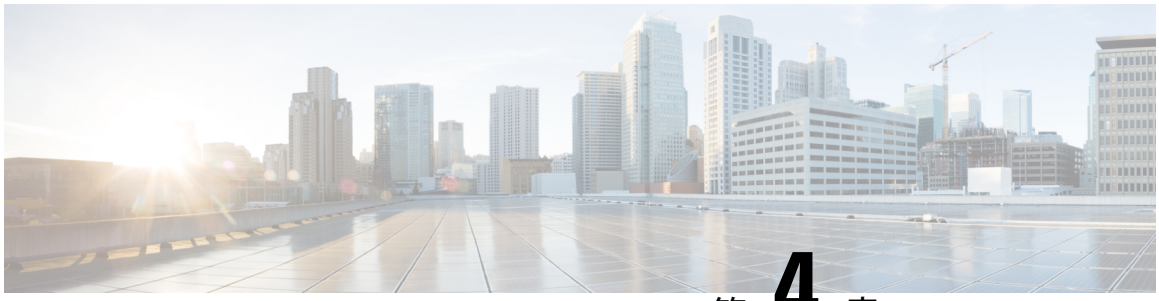
図 *12*：権限マッピング



ステップ 5 [一致（Match）] フィールドに、シスコ ポリシー管理ツールのエントリを入力します。一致は、ポリシー管理ツールの UI から [OAuthクライアント（OAuth Clients）] タブに移動して、[クライアントID（Client ID）] 列で見つけることができます。

ステップ 6 [アクセス権限（Access Permission）] フィールドに適切な権限（例：`permission/admin`）を入力します。

ステップ 7 [保存（Save）] をクリックします。



第 4 章

プラットフォームの正常性とログ

ここでは、次の内容について説明します。

プラットフォームの正常性とログ

CWM は、Kubernetes クラスタアーキテクチャをランタイム環境として活用するマイクロサービスベースのアプリケーションです。したがって、Kubernetes コマンドを使用して CWM アプリケーションの正常性を確認できます。

（注） サポートされているすべての `kubectl` コマンドを表示するには、VM の OS にログインし、`kubectl --help` を使用します。

ポッドステータスの確認

ステップ 1 コマンドラインターミナルを使用して、SSH で仮想マシンの OS にログインします。

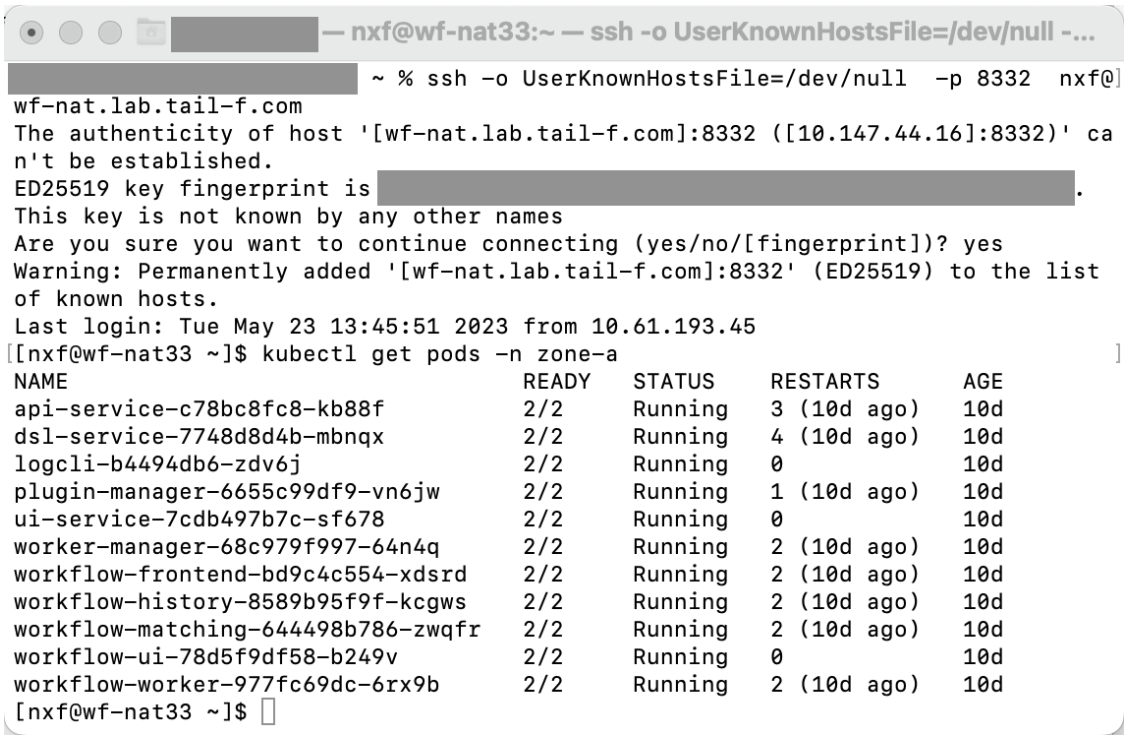
```
ssh -o UserKnownHostsFile=/dev/null  -p 22  nxf@<your_resource_pool_address>
```

ステップ 2 名前空間 `zone-a`（これは、CWM マイクロサービスを含むポッドのデフォルトの名前空間です）のポッドのステータスを確認するには、次のコマンドを実行します。

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

ステップ 3 ポッドのリストが表示されます。

図 *13 : k8s* ポッドの取得



ステップ 4 ポッドのステータスが `Running` 以外の場合は、次のコマンドを使用してポッドを「再起動」できます。

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

ポッドは削除されますが、Kubernetes 設定は宣言型であるため、削除されたポッドは効果的に再作成されて再実行します。

ログの確認と収集

アプリケーションログは、**Loki logCLI** コマンドライン インターフェイスで確認できます。CWM プラットフォームからログを収集するには、次の手順を実行します。

ステップ 1 コマンドラインターミナルを使用し、SSH クライアントを使用してシステムに接続します。

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

（注） `SSH_PORT` と `ip_address_of_deployment` を適宜調整します。

ステップ 2 ログインに成功したら、次のコマンドを使用して、実行中のすべてのポッドを一覧表示します。

```
kubectl get pods -A
```

結果の例：

```
[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
```

ステップ 3 zone-a 名前空間で使用可能な logcli ツールを特定します。この例では、logcli-5c7ddbc95d-mkpcc という名前のポッドです。

ステップ 4 正しいポッドに接続し、フィルタを適用可能なログラベルを一覧表示します。

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

ステップ 5 zone-a 名前空間で実行されているすべてのアプリケーションからログを収集し、単一のファイルに保存します。トラブルシューティングイベントが発生したときに関連する期間からログを収集するように、-since オプションを調整してください。

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

ステップ 6 同様に、便宜上別のファイルを使用して、他の名前空間からログを収集します。

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

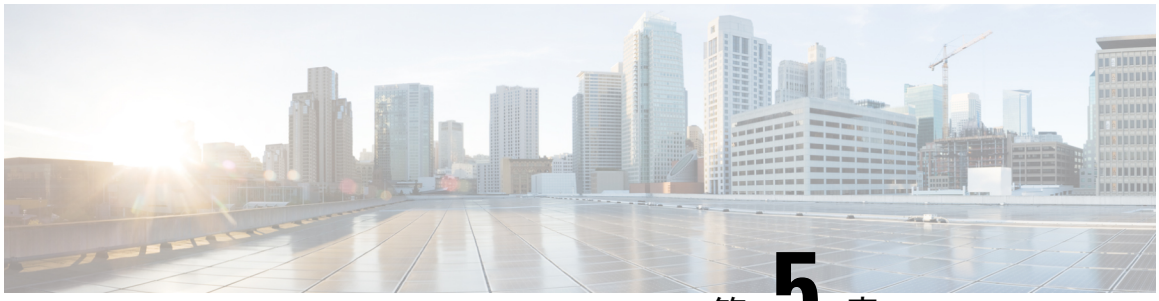
```
kubectl exec --namespace=zone-a -ti logcli-5c7ddbc95d-mkpcc -- logcli query '{namespace="kube-system"}'
 --since 60m > kube-system.log
```

ステップ 7 SCP ツールを使用して、システムからデスクトップにログファイルをコピーします。

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

ステップ 8 最後に、ログをサポートに送信し、発生している問題の詳細な説明を提供できます。

（注） logCLI コマンドと使用方法の詳細については、logCLI Grafana のドキュメントを参照してください。



第 5 章

CWM API Postman コレクション

ここでは、次の内容について説明します。

CWM API Postman コレクション

CWM API は、Representational State Transfer（REST）の設計原則に従って開発されました。API には、JSON データ形式を使用した HTTP を使用してアクセスします。関連する HTTP 応答コードで、要求の処理の成否が示されます。データ取得メソッドには GET 要求が必要です。一方、データを追加、変更、または削除するメソッドには POST、PUT、PATCH、または DELETE メソッドが必要です。要求が誤ったリクエストタイプで送信されると、エラーが返されます。CWM UI からアクセスする Swagger インターフェイスが製品に直接組み込まれていますが、使いやすさを考慮して、サンプル要求を含む Postman コレクションも提供されています。

API Postman コレクションの使用

コレクションのインポートと環境の設定：

始める前に

- Postman Web アプリアカウントを作成するか、Postman デスクトップをインストールします。
- このリンクをクリックして、JSON 形式の Postman コレクションをダウンロードします。zip アーカイブを解凍します。

ステップ 1 Postman を開き、[コレクション（Collections）] に移動します。

ステップ 2 [インポート（Import）] をクリックし、[インポートするには任意の場所にドロップ（DropAnywhere to Import）] 画面からフォルダを選択し、zip アーカイブから解凍したフォルダをポイントします。

ステップ 3 [環境（Environment）] に移動し、新しくインポートした**テスト**環境を選択します。

ステップ 4 CWM の IP アドレスとポートに合わせて **baseUrl** 変数と **port** 変数の現在の値を指定し、変更を保存します。

これで設定が完了し、コレクションを使用する準備が整いました。

翻訳について
このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。