# System

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。

- **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。

# 正常性とログの確認

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

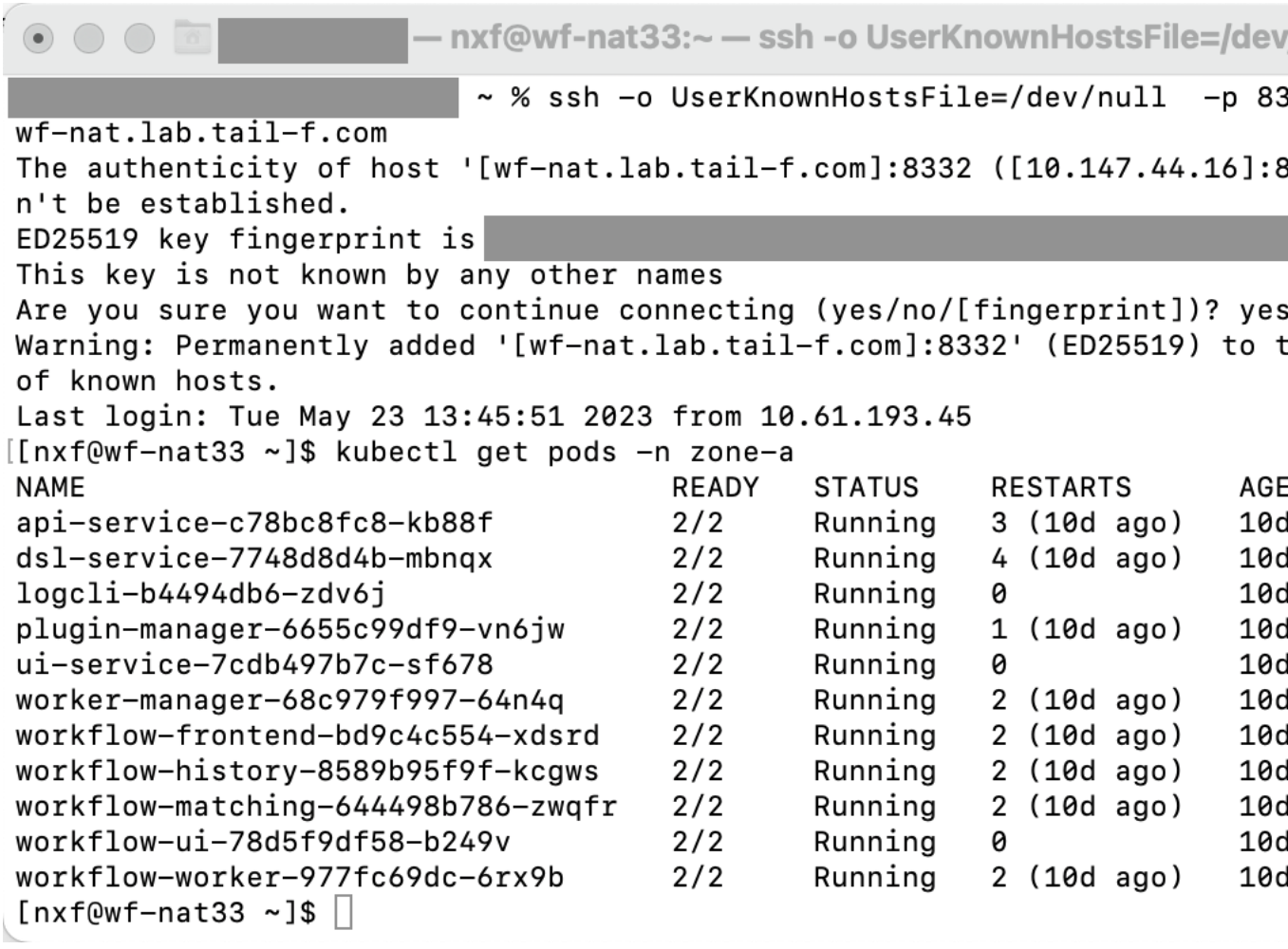
**ステップ 1** コマンドラインターミナルを使用して、SSH で仮想マシンの OS にログインします。

```
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。



**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。

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。