Red Hat Ansible Automation Platform 2.4

Operator ベースのインストール向け Red Hat Ansible Automation Platform Automation Mesh

クラウドネイティブな方法で大規模に自動化する

Last Updated: 2024-08-01

Red Hat Ansible Automation Platform 2.4 Operator ベースのインストール向け Red Hat Ansible Automation Platform Automation Mesh

クラウドネイティブな方法で大規模に自動化する

法律上の通知

概要

このガイドでは、Operator ベースの Ansible Automation Platform 環境の一部として、自動化メッシュをデプロイする方法を説明します。

目次

はじめに

Red Hat Ansible Automation Platform に興味をお持ちいただきありがとうございます。Ansible Automation Platform は、Ansible を装備した環境に、制御、ナレッジ、委譲の機能を追加して、チームが複雑かつ複数層のデプロイメントを管理できるように支援する商用サービスです。

このガイドでは、Red Hat Ansible Automation Platform の Operator ベースのインストールにおける自動化メッシュのセットアップに関する要件とプロセスについて説明します。このガイドの更新により、Ansible Automation Platform の最新リリースの情報が追加されました。

RED HAT ドキュメントへのフィードバック (英語のみ)

このドキュメントを改善するための提案がある場合、またはエラーを見つけた場合は、テクニカルサポート (https://access.redhat.com) に連絡し、**docs-product** コンポーネントを使用して Ansible Automation Platform Jira プロジェクトで Issue を作成してください。

第1章 OPERATOR ベースの RED HAT ANSIBLE AUTOMATION PLATFORM 環境における自動化メッシュの計画

次のトピックには、Operator ベースの Red Hat Ansible Automation Platform 環境における自動化メッシュのデプロイメントを計画する際に役立つ情報が含まれています。このドキュメントでは、OpenShift Container Platform や Microsoft Azure マネージドアプリケーション上の Ansible Automation Platform など、Operator ベースのデプロイメントに自動化メッシュを設定する方法を説明します。

1.1. 自動化メッシュについて

自動化メッシュは、既存ネットワークを使用して互いにピアツーピア接続を確立しているノードを介して、大規模な分散ワーカーのコレクション全体で作業分散を容易にするオーバーレイネットワークです。

Red Hat Ansible Automation Platform 2 は、Ansible Tower と分離されたノードを Automation Controller および Automation Hub に置き換えます。Automation Controller は、UI、RESTful API、RBAC、ワークフローおよび CI/CD 統合を介して自動化のコントロールプレーンを提供します。一方、自動化メッシュは、制御および実行レイヤーを構成するノードの設定、検出、変更、または修正に使用できます。自動化メッシュは次の場合に役立ちます。

- 困難なネットワークトポロジーを通過する
- 実行機能 (**ansible-playbook** を実行しているマシン) をターゲットホストに近づける

ノード (コントロール、ホップ、および実行インスタンス) は、レセプターメッシュを介して相互接続され、仮想メッシュを構成します。

自動化メッシュは通信に TLS 暗号化を使用するため、外部ネットワーク (インターネットまたはその他) を通過するトラフィックは転送中に暗号化されます。

自動化メッシュは以下のものを提供します。

- 個別にスケーリングする動的クラスター容量。これにより、ダウンタイムを最小限に抑えてノードを作成、登録、グループ化、グループ化解除、および登録解除できます。
- コントロールプレーンと実行プレーンの分離。コントロールプレーンの容量とは関係なく Playbook の実行容量をスケーリングできます。
- 遅延に対する回復力があり、停止することなく再設定可能であり、停止した場合は動的に再ルーティングして別のパスを選択するデプロイメントの選択肢。
- **Federal Information Processing Standards** (FIPS) に準拠する双方向、マルチホップのメッシュ通信の可能性を含む接続性。

1.2. コントロールプレーンおよび実行プレーン

インスタンスは、相互に通信するデバイスのネットワークを構成します。インスタンスは自動化メッシュのビルディングブロックです。これらのビルディングブロックは、メッシュトポロジー内のノードとして機能します。自動化メッシュでは、ユニークなノードタイプを使用して **コントロール** プレーンと **実行** プレーンの両方を作成します。自動化メッシュトポロジーを設計する前に、コントロールプレーンおよび実行プレーン、ならびにそのノードタイプの詳細を確認してください。

1.2.1. コントロールプレーン

コントロールプレーンのインスタンスは、プロジェクトの更新や管理ジョブに加えて、Web サーバーやタスクディスパッチャーなどの永続的な Automation Controller サービスを実行します。ただし、Operator ベースのモデルには、ハイブリッドノードやコントロールノードはありません。Kubernetes クラスター上で実行されるコンテナーを構成するコンテナーグループはあります。それがコントロールプレーンを構成します。このコントロールプレーンは、Red Hat Ansible Automation Platform がデプロイされている Kubernetes クラスターに対してローカルです。

1.2.2. 実行プレーン

実行プレーン は、コントロールプレーンの代わりに自動化を実行し制御機能を持たない実行ノードで構成されます。ホップノードは、通信に対応します。**実行プレーン** のノードは、地理的にコントロールプレーンから離れた、レイテンシーの高いユーザー空間のジョブのみを実行します。

- **実行ノード** - 実行ノードは、**podman** 分離により **ansible-runner** でジョブを実行します。このノードタイプは分離されたノードに似ています。これは、実行プレーンノードのデフォルトのノードタイプです。
- **ホップノード** - ジャンプホストと同様に、ホップノードはトラフィックを他の実行ノードにルーティングします。ホップノードは自動化を実行できません。

第2章 OPERATOR ベースの RED HAT ANSIBLE AUTOMATION PLATFORM 向け自動化メッシュ

自動化メッシュのスケーリングは、Red Hat Ansible Automation Platform の OpenShift デプロイメントで利用でき、インストールスクリプトを実行せずに、Automation Controller UI の **インスタンス** リソースを使用して、クラスターにノードを動的に追加または削除することで可能です。

インスタンスは、メッシュトポロジー内のノードとして機能します。自動化メッシュを使用すると、自動化のフットプリントを拡張できます。ジョブを起動する場所は、ansible-playbook が実行される場所とは異なる場合があります。

Automation Controller UI からインスタンスを管理するには、システム管理者またはシステム監査者の権限が必要です。

一般に、ノードのプロセッサーコア (CPU) とメモリー (RAM) が多いほど、そのノードで同時に実行するようにスケジュールできるジョブの数も多くなります。

詳細は、Automation Controller の容量決定とジョブへの影響 を参照してください。

2.1. 前提条件

自動化メッシュは、Red Hat Enterprise Linux (RHEL) 上で実行されるホップおよび実行ノードに依存します。Red Hat Ansible Automation Platform サブスクリプションにより、Ansible Automation Platform のコンポーネントの実行に使用できる 10 個の Red Hat Enterprise Linux ライセンスが付与されます。

Red Hat Enterprise Linux サブスクリプションの詳細は、Red Hat Enterprise Linux ドキュメントの システムの登録とサブスクリプションの管理 を参照してください。

次の手順では、自動化メッシュのデプロイメント用の RHEL インスタンスを準備します。

1. Red Hat Enterprise Linux オペレーティングシステムが必要です。メッシュ内の各ノードには、Automation Controller がアクセスできる静的 IP アドレス、または解決可能な DNS ホスト名が必要です。
2. 続行する前に、RHEL 仮想マシンの最小要件を満たしていることを確認している。詳細は、Red Hat Ansible Automation Platform のシステム要件 を参照してください。
3. 通信が必要なリモートネットワーク内に RHEL インスタンスをデプロイします。仮想マシンの作成については、**Red Hat Enterprise Linux** ドキュメントの 仮想マシンの作成 を参照してください。提案されたタスクを仮想マシン上で実行できるように、必ず仮想マシンの容量を十分にスケーリングしてください。
   - RHEL ISO は、access.redhat.com から入手できます。
   - RHEL クラウドイメージは、console.redhat.com の Image Builder を使用してビルドできます。

2.2. 自動化メッシュで使用するための仮想マシンのセットアップ

手順

1. 各 RHEL インスタンスに SSH で接続し、次の手順を実行します。ネットワークアクセスと制御によっては、SSH プロキシーまたは他のアクセスモデルが必要になる場合があります。
   以下のコマンドを使用します。
   -

```
ssh [username]@[host_ip_address]
```

たとえば、Amazon Web Services で実行されている Ansible Automation Platform インスタンスの場合は、以下のようになります。

```
ssh ec2-user@10.0.0.6
```

2. 後のステップで、ホップノードから実行ノードに接続するために使用できる SSH キーを作成またはコピーします。これは、自動化メッシュ設定のためだけに使用される一時キー、または長期間有効なキーの場合があります。SSH ユーザーとキーは、後の手順で使用されます。

3. **baseos** および **appstream** リポジトリーを使用して、RHEL サブスクリプションを有効にします。Ansible Automation Platform RPM リポジトリーは、**Red Hat Update Infrastructure** (RHUI) はなく、subscription-manager を通じてのみ利用できます。RHEL と RHUI を含む他の Linux フットプリントを使用しようとすると、エラーが発生します。

```
sudo subscription-manager register --auto-attach
```

アカウントで Simple Content Access が有効になっている場合は、次を使用します。

```
sudo subscription-manager register
```

Simple Content Access の詳細は、Simple Content Access の使用 を参照してください。

4. Ansible Automation Platform サブスクリプションと適切な Red Hat Ansible Automation Platform チャネルを有効にします。

```
# subscription-manager repos --enable ansible-automation-platform-2.4-for-rhel-8-x86_64-rpms for RHEL 8

# subscription-manager repos --enable ansible-automation-platform-2.4-for-rhel-9-x86_64-rpms for RHEL 9
```

5. パッケージが最新であることを確認します。

```
sudo dnf upgrade -y
```

6. ansible-core パッケージをインストールします。

```
sudo dnf install -y ansible-core
```

2.3. 自動化メッシュノードタイプの定義

ジョブの容量を拡張するには、Automation Controller のデプロイメントと一緒に実行するように追加できるスタンドアロンの **実行ノード** を作成します。これらの実行ノードは、Automation Controller Kubernetes クラスターの一部ではありません。クラスター内で実行される制御ノードは、Receptor を介して実行ノードに接続し、作業を送信します。これらの実行ノードは、タイプ **execution** インスタンスとして、Automation Controller に登録されます。つまり、コントロールノードのように作業をディスパッチしたり、Web リクエストを処理したりするのではなく、ジョブを実行するためにのみ使用されます。

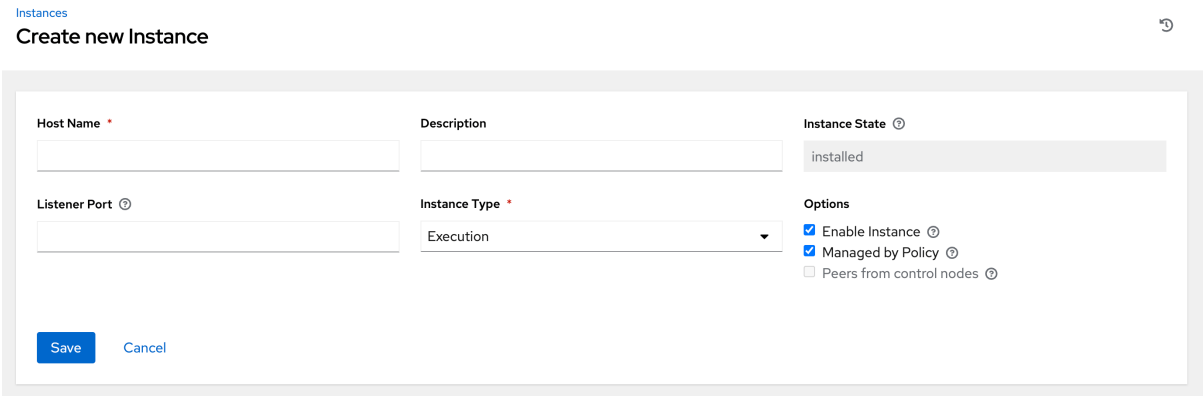
ホップノード は、Automation Controller のコントロールプレーンとスタンドアロン実行ノードの間に追加できます。これらのホップノードは Kubernetes クラスターの一部ではなく、タイプ **hop** のインス

タンスとして Automation Controller に登録されます。つまり、別のネットワークまたはより厳密なネットワーク内で到達不可能なノードへのインバウンドおよびアウトバウンドのトラフィックのみを処理します。

次の手順は、ホストのノードタイプを設定する方法を示しています。

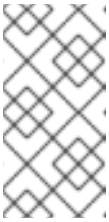
手順

1. ナビゲーションパネルから、**Administration → Instances** を選択します。
2. **Instances** リストページで、**Add** をクリックします。**Create new Instance** ウィンドウが開きます。



インスタンスには次の属性が必要です。

- **Host Name**: (必須) インスタンスの完全修飾ドメイン名 (パブリック DNS)) または IP アドレスを入力します。このフィールドは、インストーラーベースのデプロイメントの **hostname** に相当します。



注記

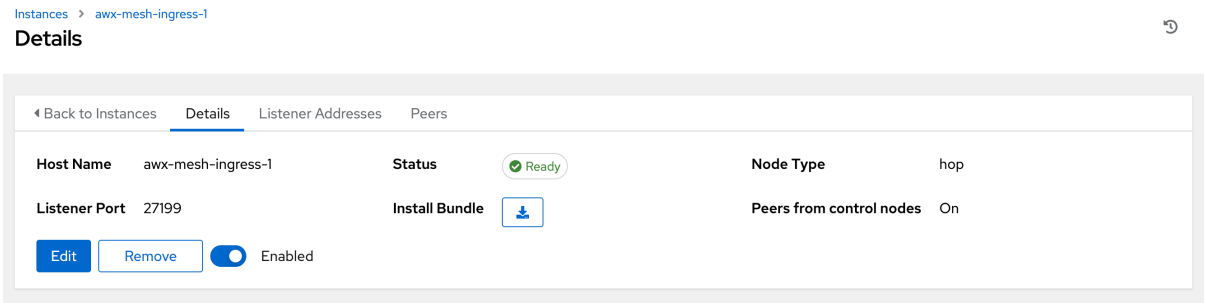
インスタンスがコントロールクラスターから解決できないプライベート DNS を使用している場合、DNS ルックアップルーティングは失敗し、生成された SSL 証明書は無効になります。代わりに IP アドレスを使用してください。

- オプション: **Description**: インスタンスの説明を入力します。
- **Instance State**: このフィールドは自動入力され、インストール中であることを示します。変更はできません。
- **リスナーポート**: このポートは、受信接続をリッスンするために receptor に使用されます。ポートを設定に適したポートに設定できます。このフィールドは、API の **listen_port** に相当します。デフォルト値は 27199 ですが、独自のポート値を設定できます。
- **インスタンスタイプ**: **execution** ノードおよび **hop** ノードのみを作成できます。Operator ベースのデプロイメントは、コントロールノードまたはハイブリッドノードをサポートしません。
  オプション:
  - **インスタンスの有効化**: 実行ノード上でジョブを実行できるようにするには、このボックスをオンにします。

- **Managed by Policy** ボックスをオンにして、ポリシーがインスタンスの割り当て方法を決定できるようにします。
- **Peers from control nodes** ボックスをオンにして、コントロールノードがこのインスタンスに自動的にピアリングできるようにします。Automation Controller に接続されているノードの場合は、**Peers from Control** nodes ボックスをチェックして、そのノードと Automation Controller の間に直接通信リンクを作成します。その他のすべてのノードの場合:
  - ホップノードを追加しない場合は、**Peers from Control** がオンになっていることを確認してください。
  - ホップノードを追加する場合は、**Peers from Control** がチェックされていないことを確認してください。
  - ホップノードと通信する実行ノードの場合は、このボックスをオンにしないでください。
- 実行ノードをホップノードとピアリングするには、 icon next to the **Peers** フィールドをクリックします。
  Select Peers ウィンドウが表示されます。

  実行ノードをホップノードにピアリングします。

3. **Save** をクリックします。

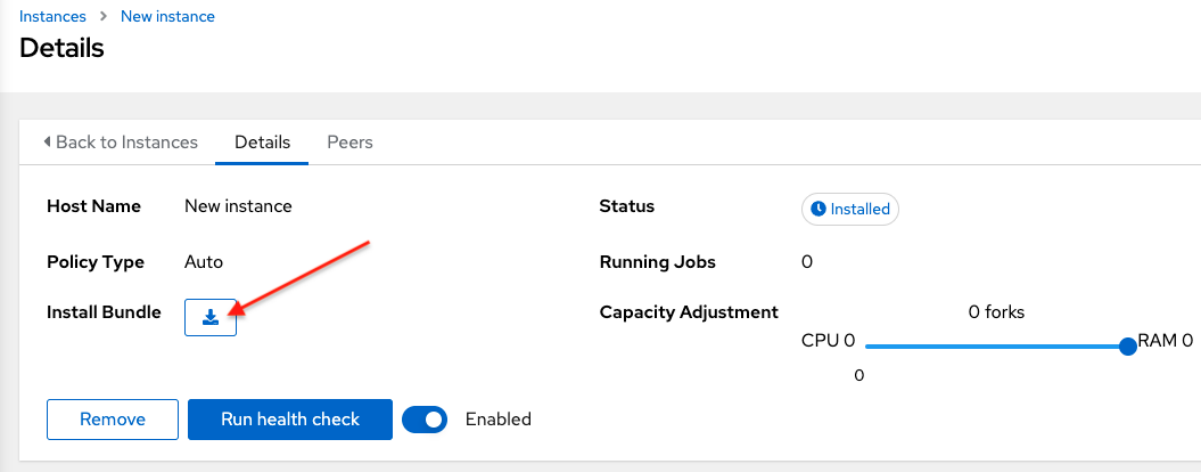


4. 更新したトポロジーをグラフィック表示するには、トポロジービューアー を参照してください。

注記

新しく作成したインスタンスに SSH アクセスできる任意のコンピューターから、次の手順を実行します。

5. **Install Bundle** の横にある アイコンをクリックして、この新しいインスタンスと、作成されたノードを自動化メッシュにインストールするために必要なファイルを含む tar ファイルをダウンロードします。



インストールバンドルには、TLS 証明書とキー、認証局、および適切な Receptor 設定ファイルが含まれています。

```
receptor-ca.crt
work-public-key.pem
receptor.key
install_receptor.yml
inventory.yml
group_vars/all.yml
requirements.yml
```

6. ダウンロードした **tar.gz** インストールバンドルを、ダウンロードした場所から展開します。これらのファイルがリモートマシン上の正しい場所にあることを確認するために、インストールバンドルには **install_receptor.yml** Playbook が含まれています。Playbook には Receptor コレクションが必要です。次のコマンドを実行して、コレクションをダウンロードします。

```
ansible-galaxy collection install -r requirements.yml
```

7. **ansible-playbook** コマンドを実行する前に、**inventory.yml** ファイル内の次のフィールドを編集します。

```
all:
  hosts:
    remote-execution:
      ansible_host: 10.0.0.6
      ansible_user: <username> # user provided
  ansible_ssh_private_key_file: ~/.ssh/<id_rsa>
```

- **ansible_host** が、ノードの IP アドレスまたは DNS に設定されていることを確認します。
- **ansible_user** を、インストールを実行しているユーザー名に設定します。
- **ansible_ssh_private_key_file** を設定して、インスタンスへの接続に使用される秘密鍵のファイル名を含めます。
- **inventory.yml** ファイルの内容はテンプレートとして機能し、メッシュトポロジーでの receptor ノードのインストールおよび設定中に適用されるロールの変数が含まれています。他のフィールドの一部を変更したり、高度なシナリオではファイル全体を置き換えたりすることができます。詳細は、Role Variables を参照してください。

8. プライベート DNS を使用するノードの場合は、次の行を **inventory.yml** に追加します。

```
ansible_ssh_common_args: <your ssh ProxyCommand setting>
```

これは、**install-ceptor.yml** Playbook に、proxy コマンドを使用してローカル DNS ノード経由でプライベートノードに接続するように指示します。

9. 属性を設定したら、**Save** をクリックします。作成したインスタンスの **Details** ページが開きます。

10. ファイルを保存して続行します。

11. インストールバンドルを実行してリモートノードをセットアップし、**ansible-playbook** を実行するシステムには、**ansible.receptor** コレクションがインストールされている必要があります。

```
ansible-galaxy collection install ansible.receptor
```

または、以下を実行します。

```
ansible-galaxy install -r requirements.yml
```

- **requirements.yml** ファイルから receptor コレクションの依存関係をインストールすると、そこで指定された receptor のバージョンが一貫して取得されます。さらに、今後必要になる可能性のある他のコレクションの依存関係も取得されます。

- Playbook が実行されるすべてのノードに receptor コレクションをインストールします。インストールしない場合は、エラーが発生します。

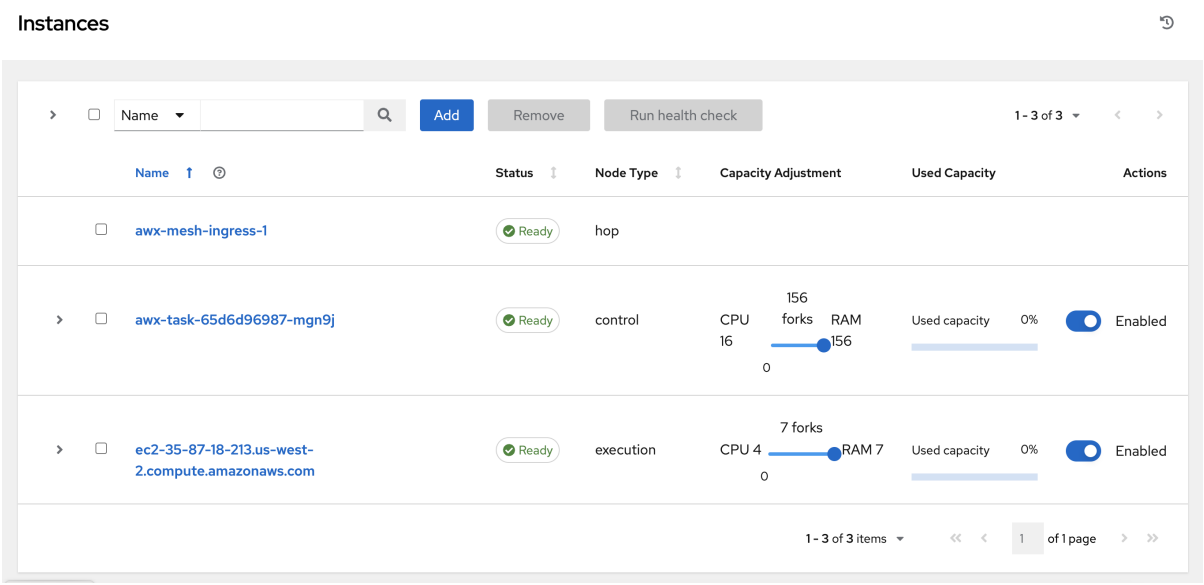
12. **receptor_listener_port** が定義されている場合、マシンには、受信 TCP 接続を確立するために使用可能なオープンポート (27199 など) も必要です。次のコマンドを実行して、receptor 通信用にポート 27199 を開きます。

```
sudo firewall-cmd --permanent --zone=public --add-port=27199/tcp
```

13. 自動化メッシュを更新するマシン上で、次の Playbook を実行します。

```
ansible-playbook -i inventory.yml install_receptor.yml
```

この Playbook を実行すると、自動化メッシュが設定されます。



メッシュからインスタンスを削除するには、インスタンスの削除 を参照してください。

2.4. インスタンスグループの作成

新しいインスタンスグループを作成するには、次の手順を実行します。

手順

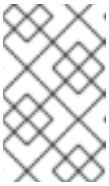
1. ナビゲーションパネルから **Administration → Instance Groups** を選択します。
2. **Add instance group** リストから **Add** を選択します。
3. 以下のフィールドに該当する詳細を入力します。
   - **Name**: 名前は一意でなければならず、"controller" という名前に指定しないようにしてください。
   - **Policy instance minimum**: 新規インスタンスがオンラインになると、このグループに自動的に最小限割り当てられるインスタンス数を入力します。
   - ***Policy instance percentage**: スライダーを使用して、新規インスタンスがオンラインになると、このグループに自動的に最小限割り当てられるインスタンスの割合を選択します。

     注記

     新しいインスタンスグループを作成する場合、ポリシーインスタンスフィールドは必要ありません。値を指定しない場合、**Policy instance minimum** と **Policy instance percentage** は、デフォルトで 0 に設定されます。

   - **Max concurrent jobs**: 特定のジョブに対して実行できるフォークの最大数を指定します。
   - **Max forks**: 特定のジョブに対して実行できる同時ジョブの最大数を指定します。

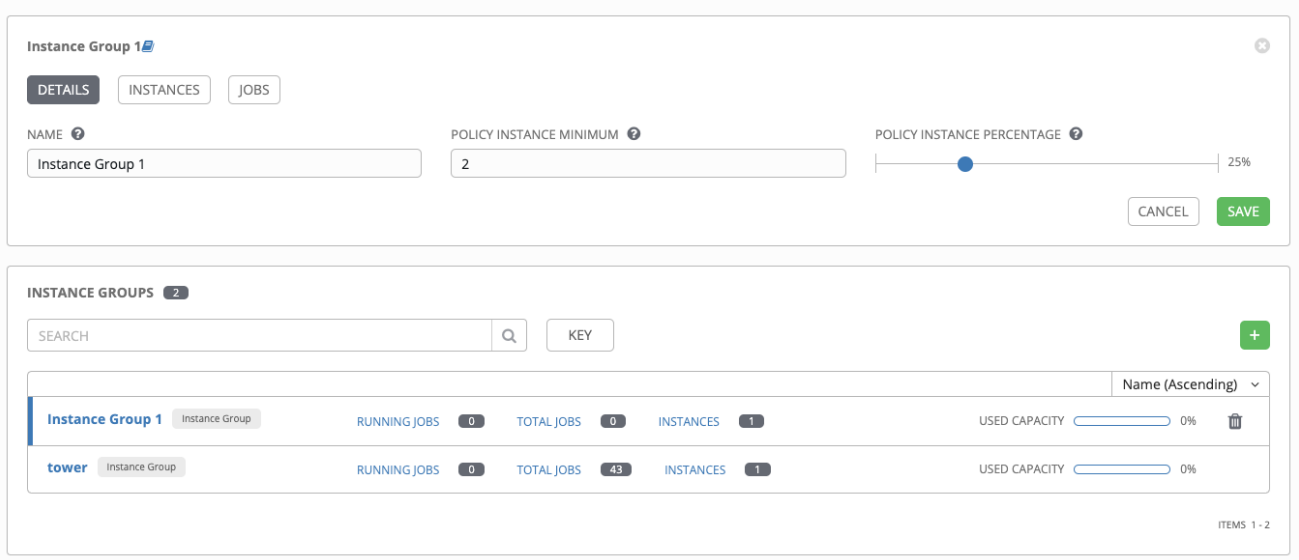


注記

Max concurrent jobs と **Max forks** のデフォルト値 0 は、制限がないことを示します。詳細は、**Automation Controller 管理ガイド** の インスタンスグループの容量制限 を参照してください。

4. **Save** をクリックします。

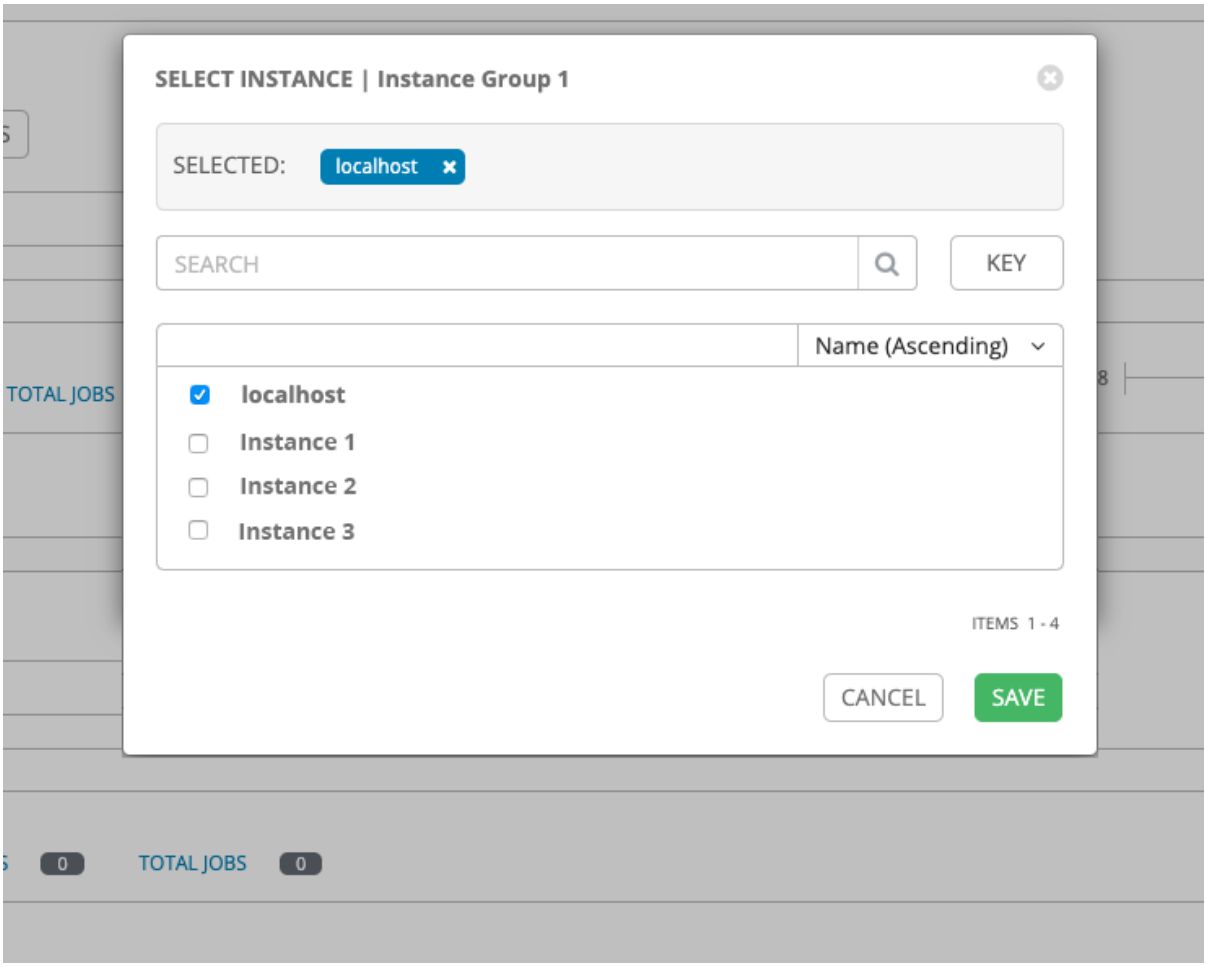
インスタンスグループが正常に作成されると、新しく作成されたインスタンスグループの **Details** タブが残り、インスタンスグループ情報を確認および編集できるようになります。これは、**Instance Groups** ビューから編集 ( ) アイコンをクリックしたときに開く画面と同じです。**Instances** を編集し、このインスタンスグループに関連付けられた **Jobs** を確認することもできます。



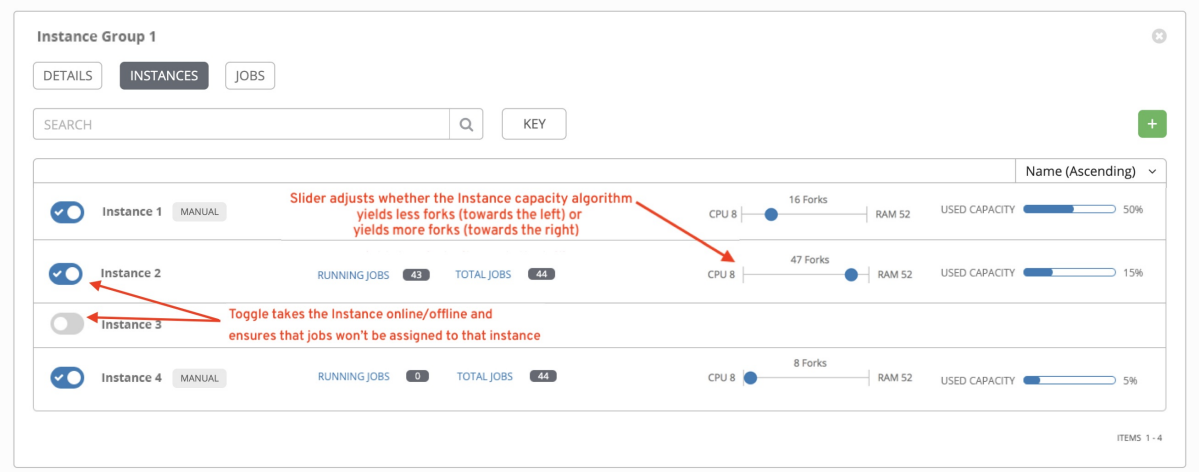
2.5. インスタンスグループへのインスタンスの関連付け

手順

1. **Instance Groups** ウィンドウの **Instance** タブを選択します。
2. **Associate** をクリックします。
3. リストから 1 つ以上の使用可能なインスタンスの横にあるチェックボックスをクリックして、インスタンスグループに関連付けるインスタンスを選択します。



4. 以下の例では、インスタンスグループに追加するインスタンスが、その容量に関する情報と合わせて表示されます。



2.6. 実行ノードでのジョブの実行

ジョブを実行する場所を指定する必要があります。指定しなかった場合、デフォルトでコントロールクラスターで実行されます。

これを行うには、ジョブテンプレートを設定します。

ジョブテンプレートの詳細は、**Automation Controller ユーザーガイド** の ジョブテンプレート を参照してください。

手順

1. **Templates** リストビューには、現在利用可能なジョブテンプレートが表示されます。この画面から、ワークフローのジョブテンプレートの起動 ( )、編集 ( )、コピー ( ) を実行できます。

2. 必要なジョブを選択し、 アイコンをクリックします。

3. ジョブを実行する **Instance Group** を選択します。ジョブテンプレートでインスタンスグループを使用するには、システム管理者によってユーザーまたはチームに権限が付与されている必要があることに注意してください。複数のジョブテンプレートを選択した場合は、選択した順序によって実行の優先順位が設定されます。

4. **Next** をクリックします。

5. **Launch** をクリックします。

2.7. OPENSHIFT CONTAINER PLATFORM デプロイメントのシークレットの取得

注記

これは、Microsoft Azure 上の Ansible Automation Platform には適用されません。

Automation Controller に付属するデフォルトの実行環境を使用してリモート実行ノードで実行する場合は、実行環境イメージをプルするための認証情報を含むプルシークレットを Automation Controller に追加する必要があります。

これを行うには、Automation Controller の namespace でプルシークレットを作成し、次のように Operator で **ee_pull_credentials_secret** パラメーターを設定します。

手順

1. 次のコマンドを使用してシークレットを作成します。

```
oc create secret generic ee-pull-secret \
    --from-literal=username=<username> \
    --from-literal=password=<password> \
    --from-literal=url=registry.redhat.io

oc edit automationcontrollers <instance name>
```

2. 次を使用して、**ee_pull_credentials_secret** と **ee-pull-secret** を仕様に追加します。

```
spec.ee_pull_credentials_secret=ee-pull-secret
```

3. Automation Controller UI からインスタンスを管理するには、システム管理者またはシステム監査者の権限が必要です。

2.8. インスタンスの削除

Add Instances ページから、ノードに対してヘルスチェックを追加、削除、または実行できます。

注記

追加のノードを作成する場合、RHEL パッケージをインストールする手順に従う必要があります。この追加のノードを既存のホップノードにピアリングする場合は、各ノードにインストールバンドルもインストールする必要があります。

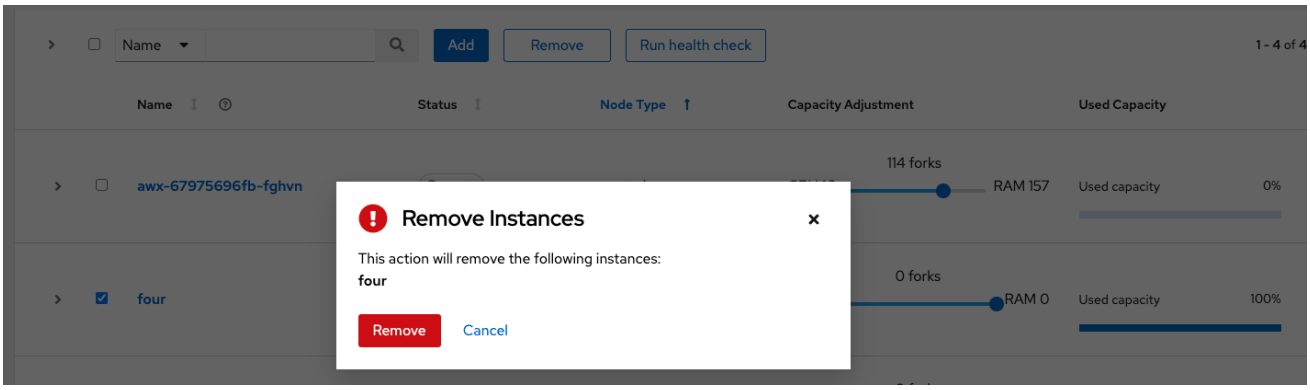
インスタンスの横にあるチェックボックスでインスタンスを選択し、インスタンスを削除するか、ヘルスチェックを実行します。

注記

- UI を使用してノードを削除すると、ノードが「削除」され、そのステータスが表示されなくなります。UI で削除する前にノードの仮想マシンを削除すると、エラーが表示されます。
- インストールバンドルを再インストールする必要があるのは、トポロジーによって通信パターンが変更された場合、つまりホップノードが変更された場合やノードを追加した場合のみです。

ボタンが無効になっている場合は、その操作を実行する権限がありません。必要なレベルのアクセスを許可するよう管理者に連絡してください。

インスタンスを削除できる場合は、確認を求めるプロンプトが表示されます。



注記

インスタンスがアクティブでジョブを実行中の場合でも、インスタンスを削除できます。Automation Controller が、このノードで実行されているジョブが完了するまで待機してから、このノードを削除します。