Red Hat Ansible Automation Platform 2.4

在 OpenShift Container Platform 上部署 Red Hat Ansible Automation Platform Operator

在 OpenShift Container Platform 上安装和配置 Ansible Automation Platform Operator

Last Updated: 2024-07-04

Red Hat Ansible Automation Platform 2.4 在 OpenShift Container Platform 上部署 Red Hat Ansible Automation Platform Operator

在 OpenShift Container Platform 上安装和配置 Ansible Automation Platform Operator

法律通告

摘要

本指南为 OpenShift Container Platform 上的 Red Hat Ansible Automation Platform Operator 支持的安装场景提供步骤和参考信息。

目录

前言

感谢您对 Red Hat Ansible Automation Platform 的关注。Ansible Automation Platform 是一个商业产品，它可以帮助团队通过增加控制、知识、协调基于 Ansible 的环境来更好地管理多阶的复杂部署环境。

本指南帮助您了解在 OpenShift Container Platform 上部署 Ansible Automation Platform Operator 的安装、迁移和升级要求。

对红帽文档提供反馈

如果您对本文档有任何改进建议，或发现了任何错误，请通过 https://access.redhat.com 联系技术支持，以使用 **docs-product** 组件在 Ansible Automation Platform JIRA 项目中创建一个问题。

第 1 章 在 RED HAT OPENSHIFT CONTAINER PLATFORM 上规划 RED HAT ANSIBLE AUTOMATION PLATFORM OPERATOR 安装

Red Hat Ansible Automation Platform 在 Red Hat Enterprise Linux 和 Red Hat Openshift 中都被支持。

OpenShift Operator 可帮助在 Red Hat OpenShift Container Platform 上安装和自动化复杂分布式软件的第 2 天操作。Ansible Automation Platform Operator 可让您在 Red Hat OpenShift Container Platform 上部署和管理 Ansible Automation Platform 组件。

您可以使用本节来帮助规划 Red Hat OpenShift Container Platform 环境中的 Red Hat Ansible Automation Platform 安装。安装前，请查看受支持的安装场景以确定哪些满足您的要求。

1.1. 关于 ANSIBLE AUTOMATION PLATFORM OPERATOR

Ansible Automation Platform Operator 在 OpenShift 环境中提供新的 Ansible Automation Platform 实例的云原生、按钮式部署。Ansible Automation Platform Operator 包含用于部署和管理自动化控制器和私有自动化中心实例的资源类型。它还包括自动化控制器作业资源，用于在自动化控制器部署中定义和启动作业。

与从 Red Hat OpenShift Container Platform 上部署的 playbook 启动实例相比，使用 Kubernetes 原生 Operator 部署 Ansible Automation Platform 实例具有诸多优势，包括对 Red Hat Ansible Automation Platform 部署的升级和完整生命周期支持。

您可以从 OperatorHub 中的 Red Hat Operator 目录安装 Ansible Automation Platform Operator。

1.2. OPENSHIFT CONTAINER PLATFORM 版本兼容性

OpenShift Container Platform 4.9 及更新的版本提供了用于安装 Ansible Automation Platform 2.4 的 Ansible Automation Platform Operator。

其他资源

- 有关最新兼容性详情，请参阅 Red Hat Ansible Automation Platform 生命周期 。

1.3. RED HAT OPENSHIFT CONTAINER PLATFORM 支持的安装场景

您可以使用 Red Hat OpenShift Container Platform Web 控制台中的 OperatorHub 来安装 Ansible Automation Platform Operator。

另外，您还可以从 OpenShift Container Platform 命令行界面(CLI) **oc** 安装 Ansible Automation Platform Operator。

按照以下其中一个工作流来安装 Ansible Automation Platform Operator，并使用它来安装所需的 Ansible Automation Platform 组件。

- 首先是控制器自定义资源，然后为自动化中心自定义资源；
- 首先是中心自定义资源，然后为自动化控制器自定义资源；
- 自动化控制器自定义资源；
- 自动化中心自定义资源。

1.4. 自定义资源

您可以为每个主要安装工作流定义自定义资源。

1.5. 其他资源

- 请参阅 Understanding OperatorHub 以了解有关 OpenShift Container Platform OperatorHub 的更多信息。

第 2 章 在 RED HAT OPENSHIFT CONTAINER PLATFORM 上安装 RED HAT ANSIBLE AUTOMATION PLATFORM OPERATOR

先决条件

- 您已在 OperatorHub 中安装了 Red Hat Ansible Automation Platform 目录。
- 您已为平台创建一个 **StorageClass** 对象，以及一个带有 **ReadWriteMany** 访问模式的持久性卷声明(PVC)。详情请参阅 动态置备。
- 要使用 **ReadWriteMany** 访问模式的 Amazon Web Services (AWS) 上运行 Red Hat OpenShift Container Platform 集群，您必须添加 NFS 或其他存储。
  - 有关 AWS Elastic Block Store (EBS)或使用 **aws-ebs** 存储类的详情，请参考使用 AWS Elastic Block Store 的持久性存储。
  - 要对 AWS EBS 使用 multi-attach **ReadWriteMany** 访问模式，请参阅将卷附加到带有 Amazon EBS Multi-Attach 的多个实例。

流程

1. 登录到 Red Hat OpenShift Container Platform。
2. 进入到 **Operators → OperatorHub**。
3. 搜索 Red Hat Ansible Automation Platform operator 并点 **Install**。
4. 选择一个 **Update Channel**:
   - **stable-2.x**：安装命名空间范围的 operator，将自动化中心和自动化控制器实例部署到安装 Operator 的命名空间。这适用于大多数情况。stable-2.x 频道不需要管理员特权并使用较少的资源，因为它仅监控一个命名空间。
   - **stable-2.x-cluster-scoped**: 在集群中的多个命名空间中部署自动化中心和自动化控制器，并且需要集群中所有命名空间的管理员特权。
5. 选择 **Installation Mode**, **Installed Namespace**, 和 **Approval Strategy**.
6. 点 **Install**。

安装过程将开始。安装完成后，一个模态将会显示通知 Red Hat Ansible Automation Platform Operator 已安装在指定命名空间中。

- 点 **View Operator** 查看您新安装的 Red Hat Ansible Automation Platform operator。

第 3 章 在 RED HAT OPENSHIFT CONTAINER PLATFORM WEB 控制台中安装和配置自动化控制器

您可以使用以下说明在 Red Hat OpenShift Container Platform 上安装自动化控制器 Operator，指定自定义资源，并使用外部数据库部署 Ansible Automation Platform。

自动化控制器配置可以通过自动化控制器 extra_settings 或部署后直接在用户界面中进行。但请注意，在 extra_settings 中进行的配置优先于用户界面中进行的设置。

注意

删除自动化控制器实例时，关联的 PVC 不会被自动删除。如果新部署的名称与前一名称相同，这可能会导致迁移期间出现问题。因此，建议您在在同一命名空间中部署新自动化控制器实例前手动删除旧的 PVC。如需更多信息，请参阅查找和删除 PVC。

3.1. 先决条件

- 您已在 Operator Hub 中安装了 Red Hat Ansible Automation Platform 目录。
- 对于 Controller，必须在集群中配置默认 StorageClass，以便 Operator 动态创建所需的 PVC。如果配置了外部 PostgreSQL 数据库，则不需要此项。
- 对于 Hub，支持 ReadWriteMany 的 StorageClass 必须在集群中可用，才能动态创建内容、redis 和 api pod 所需的 PVC。如果不是集群中的默认 StorageClass，您可以在创建 AutomationHub 对象时指定它。

3.2. 安装自动化控制器 OPERATOR

使用这个流程安装自动化控制器 Operator。

流程

1. 进入到 **Operators → Installed Operators**，然后点 **Ansible Automation Platform** operator。
2. 找到 **Automation controller** 选项卡，然后点 **Create instance**。

您可以使用 Form View 或 YAML 视图来配置实例。

3.2.1. 创建自动化控制器表单

使用这个流程，使用 form-view 创建自动化控制器。

流程

1. 确定选择了 **Form view**。默认应选择它。
2. 输入新控制器的名称。
3. 可选：添加所需的任何标签。
4. 点 **Advanced configuration**。
5. 输入实例的**主机名**。主机名是可选的。默认主机名将根据您选择的部署名称生成。

6. 输入 **Admin 帐户用户名**。
7. 输入 **管理电子邮件地址**。
8. 在 **Admin password secret** 下拉菜单中，选择 secret。
9. 在 **Database configuration secret** 下拉菜单中，选择 secret。
10. 在 **Old Database configuration secret** 下拉菜单中，选择 secret。
11. 在 **Secret key secret** 下拉菜单中，选择 secret。
12. 在 **Broadcast Websocket Secret** 下拉菜单中，选择 secret。
13. 输入所需的任何**服务帐户注解**。

3.2.2. 配置控制器镜像拉取策略

使用这个流程在自动化控制器上配置镜像拉取策略。

流程

1. 在 **Image Pull Policy** 下，点单选按钮选择
   - **Always**
   - **Never**
   - **IfNotPresent**
2. 要显示 **Image Pull Secrets** 下的选项，点箭头。
   a. 点 **Add Image Pull Secret** 旁边的 **+** 输入值。
3. 要在 **Web 容器资源要求**下拉列表中显示字段，点箭头。
   a. 在 **Limits**、和 **Requests** 下，输入 **CPU cores**, **Memory**, 和 **Storage**。
4. 要在 **Task container resource requirements** 下拉列表中显示字段，点箭头。
   a. 在 **Limits**、和 **Requests** 下，输入 **CPU cores**, **Memory**, 和 **Storage**。
5. 要显示 **EE Control Plane 容器资源要求** 下拉列表下的字段，点箭头。
   a. 在 **Limits**、和 **Requests** 下，输入 **CPU cores**, **Memory**, 和 **Storage**。
6. 要在 **PostgreSQL init 容器资源要求下显示字段（使用受管服务）** 下拉列表，点箭头。
   a. 在 **Limits**、和 **Requests** 下，输入 **CPU cores**, **Memory**, 和 **Storage**。
7. 要显示 **Redis 容器资源要求**下拉列表下的字段，点箭头。
   a. 在 **Limits**、和 **Requests** 下，输入 **CPU cores**, **Memory**, 和 **Storage**。
8. 要显示 **PostgreSQL 容器资源要求下的字段（使用受管实例）*** 下拉列表，点箭头。
   a. 在 **Limits**、和 **Requests** 下，输入 **CPU cores**, **Memory**, 和 **Storage**。

9. 要显示 **PostgreSQL 容器存储要求（使用受管实例时）** 下拉列表，点箭头。
    a. 在 **Limits**、和 **Requests** 下，输入 **CPU cores**, **Memory**, 和 **Storage**。
10. 在 Replicas 下，输入实例副本数量。
11. 在 **Remove used secrets on instance removal** 下，选择 **true** 或 **false**。默认值为 false。
12. 在 **Preload instance with data upon creation** 下，选择 **true** 或 **false**。默认值为 true。

3.2.3. 配置控制器 LDAP 安全性

使用这个流程为您的自动化控制器配置 LDAP 安全性。

流程

1. 如果您没有 **ldap_cacert_secret**，您可以使用以下命令创建一个：

```
$ oc create secret generic <resourcename>-custom-certs \
    --from-file=ldap-ca.crt=<PATH/TO/YOUR/CA/PEM/FILE>  \ 1
```

1 进行修改以指向存储您的 CA 证书的位置。

这将创建一个类似如下的 secret：

```
$ oc get secret/mycerts -o yaml
apiVersion: v1
data:
  ldap-ca.crt: <mysecret> 1
kind: Secret
metadata:
  name: mycerts
  namespace: awx
type: Opaque
```

1 在使用 **ldap_cacert_secret** 时，自动化控制器会在指定 secret 中查找 data 字段 **ldap-ca.crt**。

2. 在 **LDAP Certificate Authority Trust Bundle** 下，点下拉菜单并选择 **ldap_cacert_secret**。
3. 在 **LDAP Password Secret** 下，点下拉菜单并选择 secret。
4. 在 **EE Images Pull Credentials Secret** 下，点下拉菜单并选择 secret。
5. 在 **Bundle Cacert Secret** 下，点下拉菜单并选择一个 secret。
6. 在 **Service Type** 下，点下拉菜单并选择
    - **ClusterIP**
    - **LoadBalancer**
    - **NodePort**

3.2.4. 配置自动化控制器 Operator 路由选项

Red Hat Ansible Automation Platform Operator 安装表单允许您在**高级配置**下进一步配置自动化控制器 operator 路由选项。

流程

1. 点 **Advanced configuration**。
2. 在 **Ingress type** 下，点下拉菜单并选择 **Route**。
3. 在 **Route DNS host** 下，输入路由要回答的通用主机名。
4. 在 **Route TLS termination mechanism** 下，点下拉菜单并选择 **Edge** 或 **Passthrough**。对于大多数实例，应该选择 **Edge**。
5. 在 **Route TLS credential secret** 下，点下拉菜单并从列表中选择一个 secret。
6. 在 **Enable persistence for*/var/lib/projects* directory,** 选择 true 或 false（移动滑动条）。

3.2.5. 为自动化控制器 Operator 配置 Ingress 类型

Red Hat Ansible Automation Platform Operator 安装表单允许您在 **高级配置**下进一步配置 automation controller operator Ingress。

流程

1. 点 **Advanced Configuration**。
2. 在 **Ingress type** 下，点下拉菜单并选择 **Ingress**。
3. 在 **Ingress annotations** 下，输入要添加到 ingress 的任何注解。
4. 在 **Ingress TLS secret** 下，点下拉菜单并从列表中选择一个 secret。

配置自动化控制器 Operator 后，点表单视图底部的 **Create**。Red Hat OpenShift Container Platform 现在会创建 pod。这可能需要几分钟时间。

您可以通过进入到 **Workloads → Pods** 并查找新创建的实例来查看进度。

验证

验证来自自动化控制器的 Ansible Automation Platform Operator 安装提供的以下 Operator pod 是否正在运行：

Operator Manager 控制器	自动化控制器	Automation hub
每 3 个 operator 的 operator 管理器控制器，包括： • automation-controller-operator-controller-manager • automation-hub-operator-controller-manager • resource-operator-controller-manager	部署自动化控制器后，您将看到添加的 pod： • controller • controller-postgres	部署自动化中心后，您将看到添加的 pod： • hub-api • hub-content • hub-postgres • hub-redis • hub-worker

注意

缺少 pod 可以代表需要 pull secret。受保护的或私有镜像 registry 需要 pull secret。如需更多信息，请参阅使用镜像 pull secret。您可以通过运行 **oc describe pod <pod-name>** 来查看 该 pod 上是否有 ImagePullBackOff 错误来进一步诊断此问题。

3.3. 在 RED HAT ANSIBLE AUTOMATION PLATFORM OPERATOR 上为自动化控制器配置外部数据库

对于希望使用外部数据库部署 Ansible Automation Platform 的用户，可以通过使用实例凭证和连接信息配置 secret，然后使用 **oc create** 命令将其应用到其集群中。

默认情况下，Red Hat Ansible Automation Platform Operator 会在与 Ansible Automation Platform 部署相同的命名空间中创建并配置一个受管 PostgreSQL pod。您可以使用外部数据库部署 Ansible Automation Platform，而不是 Red Hat Ansible Automation Platform Operator 自动创建的受管 PostgreSQL pod。

使用外部数据库可让您共享和重复使用资源，并手动管理备份、升级和性能优化。

注意

只要数据库名称不同，同一个外部数据库（PostgreSQL 实例）可用于自动化 hub 和自动化控制器。换句话说，您可以在一个 PostgreSQL 实例中，带有使用不同名称的多个数据库。

以下部分概述了在 Ansible Automation Platform operator 上为自动化控制器配置外部数据库的步骤。

前提条件

外部数据库必须是 PostgreSQL 数据库，这是 Ansible Automation Platform 当前发行版本支持的版本。

注意

Ansible Automation Platform 2.4 支持 PostgreSQL 13。

流程

外部 postgres 实例凭证和连接信息必须存储在 secret 中，然后在自动化控制器 spec 中设置。

1. 按照下面的模板，创建一个 **postgres_configuration_secret** .yaml 文件：

```
apiVersion: v1
kind: Secret
metadata:
  name: external-postgres-configuration
  namespace: <target_namespace> (1)
stringData:
  host: "<external_ip_or_url_resolvable_by_the_cluster>" (2)
  port: "<external_port>" (3)
  database: "<desired_database_name>"
  username: "<username_to_connect_as>"
  password: "<password_to_connect_with>" (4)
  sslmode: "prefer" (5)
  type: "unmanaged"
type: Opaque
```

(1) 要创建 secret 的命名空间。这应该是您要部署到的同一命名空间。

(2) 数据库节点的可解析的主机名。

(3) 外部端口默认为 **5432**。

(4) 变量 **password** 的值不应包含单引号或双引号（'、"）或反斜杠(\)，以避免在部署、备份或恢复过程中出现任何问题。

(5) 变量 **sslmode** 仅适用于**外部**数据库。允许的值是：**prefer**, **disable**, **allow**, **require**, **verify-ca**, 和 **verify-full**。

2. 使用 **oc create** 命令将 **external-postgres-configuration-secret.yml** 应用到您的集群。

```
$ oc create -f external-postgres-configuration-secret.yml
```

3. 在创建 **AutomationController** 自定义资源对象时，在 spec 中指定 secret，参见以下示例：

```
apiVersion: automationcontroller.ansible.com/v1beta1
kind: AutomationController
metadata:
  name: controller-dev
spec:
  postgres_configuration_secret: external-postgres-configuration
```

3.4. 查找和删除 PVC

持久性卷声明 (PVC) 是一个存储卷，用于存储自动化 hub 和自动化控制器应用程序使用的数据。这些 PVC 独立于应用程序，即使应用程序被删除也是如此。如果您确信不再需要 PVC，或者已在其他位置备份它，您可以手动删除它们。

流程

1. 列出部署命名空间中的现有 PVC：

```
oc get pvc -n <namespace>
```

2. 通过比较旧部署名称和 PVC 名称来识别与之前部署关联的 PVC。

3. 删除旧的 PVC：

```
oc delete pvc -n <namespace> <pvc-name>
```

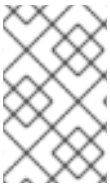
3.5. 其他资源

- 如需有关在 OpenShift Container Platform 上运行 Operator 的更多信息，请参阅 OpenShift Container Platform 产品文档，然后点 *OpenShift Container Platform 指南中的操作 Operator*。

第 4 章 在 RED HAT OPENSHIFT CONTAINER PLATFORM WEB 控制台中安装和配置自动化中心

您可以使用以下说明在 Red Hat OpenShift Container Platform 上安装自动化中心 Operator，指定自定义资源，并使用外部数据库部署 Ansible Automation Platform。

自动化中心配置可以通过自动化中心 pulp_settings 或部署后直接在用户界面中进行。但请注意，在 pulp_settings 中进行的配置优先于用户界面中进行的设置。Hub 自定义资源规格应始终将 hub 设置设置为小写。



注意

当删除自动化 hub 实例时，PVC 不会被自动删除。如果新部署的名称与前一名称相同，这可能会导致迁移期间出现问题。因此，建议您在在同一命名空间中部署新自动化 hub 实例前手动删除旧的 PVC。如需更多信息，请参阅查找和删除 PVC。

4.1. 先决条件

- 您已在 Operator Hub 中安装了 Red Hat Ansible Automation Platform Operator。

4.2. 安装自动化 HUB OPERATOR

使用这个流程安装自动化中心 Operator。

流程

1. 进入到 **Operators → Installed Operators**。
2. 找到 **Automation hub** 条目，然后点 **Create instance**。

4.2.1. 在 Red Hat OpenShift Container Platform 上安装 Ansible Automation Platform Operator 的存储选项

自动化 hub 需要基于 **ReadWriteMany** 文件、Azure Blob 存储或 Amazon S3 兼容存储才能进行操作，以便多个 pod 可以访问共享内容，如集合。

在 **AutomationHub** CR 上配置对象存储的过程与 Amazon S3 和 Azure Blob Storage 类似。

如果您使用基于文件的存储，且安装场景包含自动化中心，请确保 Ansible Automation Platform Operator 的 storage 选项被设置为 **ReadWriteMany**。**ReadWriteMany** 是默认的存储选项。

另外，OpenShift Data Foundation 提供了一个 **ReadWriteMany** 或 S3 兼容实现。另外，您可以设置 NFS 存储配置来支持 **ReadWriteMany**。但是，这会将 NFS 服务器作为潜在的单点故障引入。

其他资源

- OpenShift Container Platform *Storage* 指南中的使用 NFS 的持久性存储
- IBM 如何在 OpenShift 环境中为 NFS 动态存储部署创建存储类？

4.2.1.1. 使用 ReadWriteMany 访问模式置备 OCP 存储

要确保成功安装 Ansible Automation Platform Operator，您必须为自动化 hub 置备存储类型，最初为 **ReadWriteMany** 访问模式。

流程

1. 单击 Provisioning 以更新访问模式。
2. 在第一步中，将 **accessModes** 从默认的 **ReadWriteOnce** 更新为 **ReadWriteMany**。
3. 完成本节中的额外步骤，以创建持久性卷声明 (PVC)。

4.2.1.2. 在 Amazon S3 中配置对象存储

红帽支持用于自动化中心的 Amazon Simple Storage Service (S3)。您可以在部署 **AutomationHub** 自定义资源 (CR) 时配置它，也可以为现有实例配置它。

先决条件

- 创建 Amazon S3 存储桶以存储对象。
- 请注意 S3 存储桶的名称。

流程

1. 创建包含 AWS 凭证和连接详情的 Kubernetes secret，以及 Amazon S3 存储桶的名称。以下示例创建一个名为 **test-s3** 的 secret：

```
$ oc -n $HUB_NAMESPACE apply -f- <<EOF
apiVersion: v1
kind: Secret
metadata:
  name: 'test-s3'
stringData:
  s3-access-key-id: $S3_ACCESS_KEY_ID
  s3-secret-access-key: $S3_SECRET_ACCESS_KEY
  s3-bucket-name: $S3_BUCKET_NAME
  s3-region: $S3_REGION
EOF
```

2. 将 secret 添加到自动化中心自定义资源 (CR) **spec** 中：

```
spec:
  object_storage_s3_secret: test-s3
```

3. 如果要将此 secret 应用到现有实例，请重启 API pod 以使更改生效。**<hub-name>** 是 hub 实例的名称。

```
$ oc -n $HUB_NAMESPACE delete pod -l app.kubernetes.io/name=<hub-name>-api
```

4.2.1.3. 在 Azure Blob 中配置对象存储

红帽支持用于自动化中心的 Azure Blob Storage。您可以在部署 **AutomationHub** 自定义资源 (CR) 时配置它，也可以为现有实例配置它。

先决条件

- 创建 Azure Storage blob 容器来存储对象。
- 请注意 blob 容器的名称。

流程

1. 创建一个包含 Azure 帐户凭证和连接详情的 Kubernetes secret，以及 Azure Storage blob 容器的名称。以下示例创建一个名为 **test-azure** 的 secret：

```
$ oc -n $HUB_NAMESPACE apply -f- <<EOF
apiVersion: v1
kind: Secret
metadata:
  name: 'test-azure'
stringData:
  azure-account-name: $AZURE_ACCOUNT_NAME
  azure-account-key: $AZURE_ACCOUNT_KEY
  azure-container: $AZURE_CONTAINER
  azure-container-path: $AZURE_CONTAINER_PATH
  azure-connection-string: $AZURE_CONNECTION_STRING
EOF
```

2. 将 secret 添加到自动化中心自定义资源 (CR) **spec** 中：

```
spec:
  object_storage_azure_secret: test-azure
```

3. 如果要将此 secret 应用到现有实例，请重启 API pod 以使更改生效。**<hub-name>** 是 hub 实例的名称。

```
$ oc -n $HUB_NAMESPACE delete pod -l app.kubernetes.io/name=<hub-name>-api
```

4.2.2. 配置自动化 hub Operator 路由选项

Red Hat Ansible Automation Platform Operator 安装表单允许您在 **高级配置**下进一步配置 Automation hub operator 路由选项。

流程

1. 点 **Advanced configuration**。
2. 在 **Ingress type** 下，点下拉菜单并选择 **Route**。
3. 在 **Route DNS host** 下，输入路由要回答的通用主机名。
4. 在 **Route TLS termination mechanism** 下，点下拉菜单并选择 **Edge** 或 **Passthrough**。
5. 在 **Route TLS credential secret** 下，点下拉菜单并从列表中选择一个 secret。

4.2.3. 为自动化中心 Operator 配置 Ingress 类型

Red Hat Ansible Automation Platform Operator 安装表单允许您在 **高级配置** 下进一步配置 Automation hub operator Ingress。

流程

1. 点 **Advanced Configuration**。
2. 在 **Ingress type** 下，点下拉菜单并选择 **Ingress**。
3. 在 **Ingress annotations** 下，输入要添加到 ingress 的任何注解。
4. 在 **Ingress TLS secret** 下，点下拉菜单并从列表中选择一个 secret。

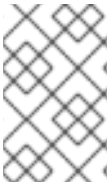
配置 Automation hub Operator 后，点表单视图底部的 **Create**。Red Hat OpenShift Container Platform 现在会创建 pod。这可能需要几分钟时间。

您可以通过进入到 **Workloads → Pods** 并查找新创建的实例来查看进度。

验证

验证来自自动化中心的 Ansible Automation Platform Operator 安装提供的以下 Operator pod 是否正在运行：

Operator Manager 控制器	自动化控制器	Automation hub
每 3 个 operator 的 operator 管理器控制器，包括： • automation-controller-operator-controller-manager • automation-hub-operator-controller-manager • resource-operator-controller-manager	部署自动化控制器后，您将看到添加的 pod： • controller • controller-postgres	部署自动化中心后，您将看到添加的 pod： • hub-api • hub-content • hub-postgres • hub-redis • hub-worker



注意

缺少 pod 可以代表需要 pull secret。受保护的或私有镜像 registry 需要 pull secret。如需更多信息，请参阅使用镜像 pull secret。您可以通过运行 **oc describe pod <pod-name>** 来查看 该 pod 上是否有 ImagePullBackOff 错误来进一步诊断此问题。

4.3. 在 OPENSHIFT CONTAINER PLATFORM 中为 ANSIBLE AUTOMATION HUB 配置 LDAP 身份验证

在 Hub 实例配置文件的 spec 部分中，为 OpenShift Container Platform 上的 Ansible Automation Platform 配置 LDAP 身份验证设置。

流程

- 使用以下示例在自动化中心实例中配置 LDAP。对于任何空白字段，请输入 ''。

```
spec:
  pulp_settings:
    auth_ldap_user_attr_map:
      email: "mail"
      first_name: "givenName"
      last_name: "sn"
    auth_ldap_group_search_base_dn: 'cn=groups,cn=accounts,dc=example,dc=com'
    auth_ldap_bind_dn: ' '
    auth_ldap_bind_password: ' '
    auth_ldap_group_search_filter: (objectClass=posixGroup)
    auth_ldap_user_search_scope: SUBTREE
    auth_ldap_server_uri: 'ldap://ldapserver:389'
    authentication_backend_preset: ldap
    auth_ldap_mirror_groups: 'True'
    auth_ldap_user_search_base_dn: 'cn=users,cn=accounts,dc=example,dc=com'
    auth_ldap_bind_password: 'ldappassword'
    auth_ldap_user_search_filter: (uid=%(user)s)
    auth_ldap_group_search_scope: SUBTREE
    auth_ldap_user_flags_by_group: '@json {"is_superuser": "cn=tower-
admin,cn=groups,cn=accounts,dc=example,dc=com"}'
```

注意

不要将任何字段留空。对于没有变量的字段，输入 '' 来表示默认值。

4.4. 访问自动化 HUB 用户界面

所有 pod 成功启动后，您可以访问自动化 hub 接口。

流程

1. 进入到 **Networking → Routes**。
2. 在 **Location** 下，点您的自动化 hub 实例的 URL。

Automation hub 用户界面会启动，您可以在其中使用 Operator 配置流程中指定的管理员凭证进行登录。

注意

如果您没有在配置期间指定管理员密码，则会为您自动创建密码。要找到此密码，访问您的项目，选择 **Workloads → Secrets**，打开 controller-admin-password。您可以从那里复制密码并将其粘贴到 Automation hub 密码字段中。

4.5. 在 RED HAT ANSIBLE AUTOMATION PLATFORM OPERATOR 上为自动化中心配置外部数据库

对于希望使用外部数据库部署 Ansible Automation Platform 的用户，可以通过使用实例凭证和连接信息配置 secret，然后使用 **oc create** 命令将其应用到其集群中。

默认情况下，Red Hat Ansible Automation Platform Operator 会在与 Ansible Automation Platform 部署相同的命名空间中创建并配置一个受管 PostgreSQL pod。

如果用户更喜欢使用专用节点来确保专用资源或手动管理备份、升级或性能调整，则用户可能会选择使用外部数据库。

注意

只要数据库名称不同，同一个外部数据库（PostgreSQL 实例）可用于自动化 hub 和自动化控制器。换句话说，您可以在一个 PostgreSQL 实例中，带有使用不同名称的多个数据库。

以下概述了在 Ansible Automation Platform operator 上为您的自动化中心配置外部数据库的步骤。

前提条件

外部数据库必须是 PostgreSQL 数据库，这是 Ansible Automation Platform 当前发行版本支持的版本。

注意

Ansible Automation Platform 2.4 支持 PostgreSQL 13。

流程

外部 postgres 实例凭证和连接信息需要存储在 secret 中，然后在自动化中心 spec 中设置。

1. 按照下面的模板，创建一个 **postgres_configuration_secret** .yaml 文件：

```
apiVersion: v1
kind: Secret
metadata:
  name: external-postgres-configuration
  namespace: <target_namespace> (1)
stringData:
  host: "<external_ip_or_url_resolvable_by_the_cluster>" (2)
  port: "<external_port>" (3)
  database: "<desired_database_name>"
  username: "<username_to_connect_as>"
  password: "<password_to_connect_with>" (4)
  sslmode: "prefer" (5)
  type: "unmanaged"
type: Opaque
```

(1) 要创建 secret 的命名空间。这应该是您要部署到的同一命名空间。

(2) 数据库节点的可解析的主机名。

(3) 外部端口默认为 **5432**。

(4) 变量 **password** 的值不应包含单引号或双引号（'、"）或反斜杠(\)，以避免在部署、备份或恢复过程中出现任何问题。

(5) 变量 **sslmode** 仅适用于**外部**数据库。允许的值是：**prefer**, **disable**, **allow**, **require**, **verify-ca**, 和 **verify-full**。

2. 使用 **oc create** 命令将 **external-postgres-configuration-secret.yml** 应用到您的集群。

```
$ oc create -f external-postgres-configuration-secret.yml
```

3. 在创建 **AutomationHub** 自定义资源对象时，在 spec 中指定 secret，如下例所示：

```
apiVersion: automationhub.ansible.com/v1beta1
kind: AutomationHub
metadata:
  name: hub-dev
spec:
  postgres_configuration_secret: external-postgres-configuration
```

4.5.1. 为自动化中心 PostgreSQL 数据库启用 hstore 扩展

在 Ansible Automation Platform 2.4 中，数据库迁移脚本使用 **hstore** 字段来存储信息，因此必须启用对自动化中心 PostgreSQL 数据库的 **hstore** 扩展。

使用 Ansible Automation Platform 安装程序和受管 PostgreSQL 服务器时，此过程是自动的。

如果 PostgreSQL 数据库是外部的，则必须在自动化中心安装前手动为自动化中心 PostreSQL 数据库启用 **hstore** 扩展。

如果在自动化中心安装前没有启用 **hstore** 扩展，在数据库迁移过程中会引发故障。

流程

1. 检查 PostgreSQL 服务器上是否有扩展（自动化 hub 数据库）。

```
$ psql -d <automation hub database> -c "SELECT * FROM pg_available_extensions WHERE
name='hstore'"
```

其中 **<automation hub database>** 的默认值为 **automationhub**。

带有 hstore 可用的输出示例 ：

```
name  | default_version | installed_version |comment
------+-----------------+-------------------+----------------------------------------------------
 hstore | 1.7           |                   | data type for storing sets of (key, value) pairs
(1 row)
```

带有 hstore 不可用的输出示例 ：

```
 name | default_version | installed_version | comment
------+-----------------+-------------------+---------
(0 rows)
```

2. 在基于 RHEL 的服务器上，**hstore** 扩展包含在 **postgresql-contrib** RPM 软件包中，该软件包在安装 PostgreSQL 服务器 RPM 软件包时不会自动安装。
要安装 RPM 软件包，请使用以下命令：

```
dnf install postgresql-contrib
```

3. 使用以下命令，在自动化中心数据库上创建 **hstore** PostgreSQL 扩展：

```
$ psql -d <automation hub database> -c "CREATE EXTENSION hstore;"
```

输出：

```
CREATE EXTENSION
```

4. 在以下输出中，**installed_version** 字段包含使用的 **hstore** 扩展，表示启用了 **hstore**。

```
name | default_version | installed_version | comment
-----+-----------------+-------------------+-----------------------------------------------------
hstore |    1.7     |      1.7          | data type for storing sets of (key, value) pairs
(1 row)
```

4.6. 查找和删除 PVC

持久性卷声明 (PVC) 是一个存储卷，用于存储自动化 hub 和自动化控制器应用程序使用的数据。这些 PVC 独立于应用程序，即使应用程序被删除也是如此。如果您确信不再需要 PVC，或者已在其他位置备份它，您可以手动删除它们。

流程

1. 列出部署命名空间中的现有 PVC：

```
oc get pvc -n <namespace>
```

2. 通过比较旧部署名称和 PVC 名称来识别与之前部署关联的 PVC。

3. 删除旧的 PVC：

```
oc delete pvc -n <namespace> <pvc-name>
```

4.7. 其他配置

集合下载数可帮助您了解集合使用情况。要在自动化中心中添加集合下载计数，请设置以下配置：

```
spec:
  pulp_settings:
    ansible_collect_download_count: true
```

启用 **ansible_collect_download_count** 后，自动化中心将按集合显示下载计数。

4.8. 其他资源

- 如需有关在 OpenShift Container Platform 上运行 Operator 的更多信息，请参阅 OpenShift Container Platform 产品文档，然后点 *OpenShift Container Platform 指南中的操作 Operator*。

第 5 章 通过 OPENSHIFT CONTAINER PLATFORM CLI 安装 ANSIBLE AUTOMATION PLATFORM OPERATOR

根据这些说明，使用 **oc** 命令从 OpenShift Container Platform 命令行界面(CLI)在 Red Hat OpenShift Container Platform 上安装 Ansible Automation Platform Operator。

5.1. 先决条件

- 使用具有 operator 安装权限的账户访问 Red Hat OpenShift Container Platform。
- OpenShift Container Platform CLI **oc** 命令安装在本地系统中。如需更多信息，请参阅 Red Hat OpenShift Container Platform 产品文档中的安装 OpenShift CLI。

5.2. 使用 OPENSHIFT CONTAINER PLATFORM CLI 将命名空间订阅到 OPERATOR

使用这个流程为 Operator 订阅命名空间。

流程

1. 为 operator 创建项目

```
oc new-project ansible-automation-platform
```

2. 创建名为 **sub.yaml** 的文件。
3. 将以下 YAML 代码添加到 **sub.yaml** 文件中。

```
---
apiVersion: v1
kind: Namespace
metadata:
  labels:
    openshift.io/cluster-monitoring: "true"
  name: ansible-automation-platform
---
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: ansible-automation-platform-operator
  namespace: ansible-automation-platform
spec:
  targetNamespaces:
    - ansible-automation-platform
---
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: ansible-automation-platform
  namespace: ansible-automation-platform
spec:
  channel: 'stable-2.4'
  installPlanApproval: Automatic
```

```
  name: ansible-automation-platform-operator
  source: redhat-operators
  sourceNamespace: openshift-marketplace
---
apiVersion: automationcontroller.ansible.com/v1beta1
kind: AutomationController
metadata:
  name: example
  namespace: ansible-automation-platform
spec:
  replicas: 1
```

此文件创建一个名为 ***ansible-automation-platform*** 的 **Subscription** 对象，它将 **ansible-automation-platform** 命名空间订阅到 **ansible-automation-platform-operator** operator。

然后，它会在 **ansible-automation-platform** 命名空间中创建一个名为 ***example*** 的 **AutomationController** 对象。

要从 ***示例*** 更改自动化控制器名称，请编辑 **sub.yaml** 的 **kind: AutomationController** 部分中的 *name* 字段，并将 & ***lt;automation_controller_name*** > 替换为您要使用的名称：

```
apiVersion: automationcontroller.ansible.com/v1beta1
kind: AutomationController
metadata:
  name: <automation_controller_name>
  namespace: ansible-automation-platform
```

4. 运行 **oc apply** 命令来创建 **sub.yaml** 文件中指定的对象：

```
oc apply -f sub.yaml
```

要验证命名空间是否已成功订阅到 **ansible-automation-platform-operator** operator，请运行 **oc get subs** 命令：

```
$ oc get subs -n ansible-automation-platform
```

如需有关将命名空间订阅到 Operator 的更多信息，请参阅 Red Hat OpenShift Container Platform *Operator* 指南中的使用 CLI 从 OperatorHub 安装。

您可以使用 OpenShift Container Platform CLI 获取您创建的自动化控制器的 Web 地址和密码。

5.3. 从 OPENSHIFT CONTAINER PLATFORM CLI 获取自动化控制器登录详情

要登录到 Automation 控制器，您需要 web 地址和密码。

5.3.1. 获取自动化控制器 web 地址

Red Hat OpenShift Container Platform 路由以主机名的形式公开服务，以便外部客户端可根据名称访问该服务。创建自动化控制器实例时，为其创建一个路由。路由会继承您分配给 YAML 文件中的自动化控制器对象的名称。

使用以下命令获取路由：

```
oc get routes -n <controller_namespace>
```

在以下示例中，自动化控制器 ***example*** 在 ***ansible-automation-platform*** 命名空间中运行。

```
$ oc get routes -n ansible-automation-platform

NAME      HOST/PORT                                        PATH   SERVICES         PORT   TERMINATION
WILDCARD
example   example-ansible-automation-platform.apps-crc.testing          example-service   http
edge/Redirect   None
```

自动化控制器实例的地址为 **example-ansible-automation-platform.apps-crc.testing**。

5.3.2. 获取自动化控制器密码

sub.yaml 中自动化控制器实例的 YAML 块为 *name* 和 *admin_user* 键分配值。在以下命令中使用这些值来获取自动化控制器实例的密码。

```
oc get secret/<controller_name>-<admin_user>-password -o yaml
```

admin_user 的默认值为 ***admin***。如果您在 **sub.yaml** 中更改了 admin 用户名，请修改该命令。

以下示例为名为 ***example*** 的自动化控制器对象检索密码：

```
oc get secret/example-admin-password -o yaml
```

自动化控制器实例的密码列在输出的 **metadata** 字段中：

```
$ oc get secret/example-admin-password -o yaml

apiVersion: v1
data:
  password: ODzLODzLODzLODzLODzLODzLODzLODzLODzLODzLODzL
kind: Secret
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: '{"apiVersion":"v1","kind":"Secret","metadata":
{"labels":{"app.kubernetes.io/component":"automationcontroller","app.kubernetes.io/managed-
by":"automationcontroller-
operator","app.kubernetes.io/name":"example","app.kubernetes.io/operator-
version":"","app.kubernetes.io/part-of":"example"},"name":"example-admin-
password","namespace":"ansible-automation-platform"},"stringData":
{"password":"88TG88TG88TG88TG88TG88TG88TG88TG"}}'
  creationTimestamp: "2021-11-03T00:02:24Z"
  labels:
    app.kubernetes.io/component: automationcontroller
    app.kubernetes.io/managed-by: automationcontroller-operator
    app.kubernetes.io/name: example
    app.kubernetes.io/operator-version: ""
    app.kubernetes.io/part-of: example
  name: example-admin-password
  namespace: ansible-automation-platform
  resourceVersion: "185185"
  uid: 39393939-5252-4242-b929-665f665f665f
```

对于这个实例，密码是 **88TG88TG88TG88TG88TG88TG88TG88TG**。

5.4. 其他资源

- 如需有关在 OpenShift Container Platform 上运行 Operator 的更多信息，请参阅 OpenShift Container Platform 产品文档，然后点 *OpenShift Container Platform 指南中的操作 Operator*。

第 6 章 在 OPENSHIFT CONTAINER PLATFORM 上使用 ANSIBLE AUTOMATION PLATFORM OPERATOR 部署 EVENT-DRIVEN ANSIBLE 控制器

Event-Driven Ansible 控制器是事件驱动的自动化的接口，并引入对 IT 请求的自动解析。此组件可帮助您连接到事件的来源，并使用规则手册对这些事件执行操作。当您部署 Event-Driven Ansible 控制器时，您可以自动做出决策，使用多个事件源，在多个 IT 用例和跨多个 IT 用例中实施事件驱动的自动化，并实现更高效的服务交付。

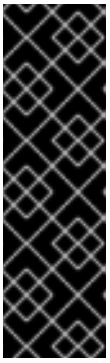
使用以下说明在 OpenShift Container Platform 上使用 Ansible Automation Platform Operator 安装 Event-Driven Ansible。

先决条件

- 您已在 OpenShift Container Platform 上安装了 Ansible Automation Platform Operator。
- 已安装并配置了自动化控制器。

流程

1. 选择 **Operators → Installed Operators**。
2. 找到并选择您的 Ansible Automation Platform 安装。
3. 在 **Provided APIs** 下，找到 Event-Driven Ansible modal，再点 **Create instance**。
   这会带您进入 Form View 来自定义安装。



重要

为确保您可以有效地运行并发事件(Driven Ansible 激活)，您必须将激活的最大数量设置为集群中可用的资源。您可以通过在 YAML 视图中调整 Event-Driven Ansible 设置来完成此操作。

当您在标准条件下激活 Event-Driven Ansible rulebook 时，它使用大约 250 MB 内存。但是，根据规则的复杂性以及处理事件的卷和大小，实际内存消耗可能会很大不同。在预计大量事件或规则手册复杂性很高的情况下，对暂存环境中的资源使用情况进行初步评估。这样可确保您的激活的最大数量取决于您的资源容量。

4. 点 **YAML 视图** 来更新 YAML 键值。
5. 在 **spec** key value 部分的末尾复制并粘贴以下字符串：

```
extra_settings:
  - setting: EDA_MAX_RUNNING_ACTIVATIONS
    value: '12'
```

6. 单击 **Reload** 和 **Save**。返回到 **Form** 视图。
7. 在 **Name** 字段中，为新的 Event-Driven Ansible 控制器部署输入您想要的名称。

重要

如果您在当前 OpenShift Container Platform 命名空间中安装了其他 Ansible Automation Platform 组件，请确保在创建 Event-Driven Ansible 自定义资源时为 Event-Driven Ansible 控制器提供唯一名称。否则，可能会发生命名冲突，并影响 Event-Driven Ansible 控制器部署。

8. 指定控制器 URL。
   如果您在 Openshift 中也部署了自动化控制器，您可以在 **Networking → Routes** 下找到导航面板中的 URL。

注意

这是唯一必需的自定义，但您可以根据需要使用 UI 表单或直接在 YAML 配置选项卡中自定义其他选项。

9. 点 **Create**。这会在您指定的命名空间中部署 Event-Driven Ansible 控制器。
   安装标记为 **Successful** 后，您可以在 Openshift UI 的 **Routes** 页面中找到 Event-Driven Ansible UI 的 URL。

10. 在导航面板中，选择 **Networking → Routes** 来查找为您创建的新 Route URL。
    路由会根据自定义资源的名称列出。

11. 点新 URL 以导航到浏览器中的 Event-Driven Ansible。

12. 在导航面板中，选择 **Workloads → Secrets** 并找到为您创建的 Admin Password k8s secret，除非您指定了自定义 secret。
    secret 根据自定义资源的名称列出，并附加 **-admin-password。**

注意

您可以使用 secret 中的 password 值登录到 Event-Driven Ansible 控制器 UI。默认用户为 **admin**。

第 7 章 使用带有自动化中心的 RED HAT SINGLE SIGN-ON OPERATOR

私有自动化中心使用 Red Hat Single Sign-On 进行验证。

Red Hat Single Sign-On Operator 会创建和管理资源。使用此 Operator 创建自定义资源，以在 Openshift 中自动执行 Red Hat Single Sign-On 管理。

- 在*虚拟机* (VM) 上安装 Ansible Automation Platform 时，安装程序可以自动安装和配置 Red Hat Single Sign-On，以用于私有自动化中心。
- 在 Red Hat OpenShift Container Platform 上安装 Ansible Automation Platform 时，您必须单独安装单点登录。

本章论述了在 OpenShift Container Platform 上安装 Ansible Automation Platform 时配置 Red Hat Single Sign-On 并将其与私有自动化中心集成的过程。

先决条件

- 您具有 operator 安装权限的账户访问 Red Hat OpenShift Container Platform。
- 已安装包含 Red Hat Ansible Automation Platform operator 的目录。
- 已安装 Red Hat Single Sign-On Operator。要安装 Red Hat Single Sign-On Operator，请按照 Red Hat Single Sign-On 文档中的使用自定义资源安装 Red Hat Single Sign-On 中的步骤操作。

7.1. 创建 KEYCLOAK 实例

安装 Red Hat Single Sign-On Operator 时，您可以创建一个 Keycloak 实例以用于 Ansible Automation Platform。

在此，您提供一个外部 Postgres 或会为您创建一个。

流程

1. 进入 **Operator → Installed Operators**。
2. 选择 **rh-sso** 项目。
3. 选择 **Red Hat Single Sign-On Operator**。
4. 在 Red Hat Single Sign-On Operator 详情页面中选择 **Keycloak**。
5. 单击 **Create instance**。
6. 点 **YAML** 视图。
   默认 Keycloak 自定义资源如下：

```
apiVersion: keycloak.org/v1alpha1
kind: Keycloak
metadata:
  name: example-keycloak
  labels:
 app: sso
  namespace: aap
```

```
spec:
  externalAccess:
 enabled: true
  instances: 1
```

7. 点 **Create**。
8. 部署完成后，您可以使用此凭证登录到管理控制台。
9. 您可以在命名空间中的 **credential-<custom-resource>** (example keycloak) secret 中找到管理员的凭证。

7.2. 为 ANSIBLE AUTOMATION PLATFORM 创建 KEYCLOAK 域

创建域来管理一组用户、凭据、角色和组。用户从属于并登陆到的域。域彼此隔离，只能管理和验证其控制的用户。

流程

1. 进入 **Operator → Installed Operators**。
2. 选择 **Red Hat Single Sign-On Operator**项目。
3. 选择 **Keycloak Realm** 选项卡，再点 **Create Keycloak Realm**。
4. 在 **Keycloak Realm** 表单中，选择 **YAML 视图**。编辑 YAML 文件，如下所示：

```
kind: KeycloakRealm
apiVersion: keycloak.org/v1alpha1
metadata:
  name: ansible-automation-platform-keycloakrealm
  namespace: rh-sso
  labels:
    app: sso
    realm: ansible-automation-platform
spec:
  realm:
    id: ansible-automation-platform
    realm: ansible-automation-platform
    enabled: true
    displayName: Ansible Automation Platform
  instanceSelector:
    matchLabels:
      app: sso
```

字段	描述
metadata.name	在元数据中为配置资源 (CR) 的名称设置唯一值。
metadata.namespace	在元数据中为配置资源 (CR) 的名称设置唯一值。
metadata.labels.app	将标签设置为唯一值。这在创建客户端 CR 时使用。

metadata.labels.realm	将标签设置为唯一值。这在创建客户端 CR 时使用。
spec.realm.id	设置 realm 名称和 id。这些名称必须相同。
spec.realm.realm	设置 realm 名称和 id。这些名称必须相同。
spec.realm.displayname	设置显示的名称。

5. 点 **Create** 并等待进程完成。

7.3. 创建 KEYCLOAK 客户端

Keycloak 客户端使用 Red Hat Single Sign-On 验证 hub 用户。当用户验证请求时，请求会通过 Keycloak 客户端。当 Single Sign-On 验证或发出 **OAuth** 令牌时，客户端会向自动化中心提供总结，用户可以登录。

流程

1. 进入 **Operator → Installed Operators**。
2. 选择 Red Hat Single Sign-On Operator 项目。
3. 选择 **Keycloak Client** 选项卡，然后点 **Create Keycloak Client**。
4. 在 Keycloak Realm 表单中，选择 **YAML** 视图。
5. 使用以下内容替换默认 YAML 文件：

```
kind: KeycloakClient
apiVersion: keycloak.org/v1alpha1
metadata:
  name: automation-hub-client-secret
  labels:
    app: sso
    realm: ansible-automation-platform
  namespace: rh-sso
spec:
  realmSelector:
    matchLabels:
      app: sso
      realm: ansible-automation-platform
  client:
    name: Automation Hub
    clientId: automation-hub
    secret: <client-secret>                    (1)
    clientAuthenticatorType: client-secret
    description: Client for automation hub
    attributes:
      user.info.response.signature.alg: RS256
      request.object.signature.alg: RS256
    directAccessGrantsEnabled: true
```

```
publicClient: true
protocol: openid-connect
standardFlowEnabled: true
protocolMappers:
  - config:
      access.token.claim: "true"
      claim.name: "family_name"
      id.token.claim: "true"
      jsonType.label: String
      user.attribute: lastName
      userinfo.token.claim: "true"
    consentRequired: false
    name: family name
    protocol: openid-connect
    protocolMapper: oidc-usermodel-property-mapper
  - config:
      userinfo.token.claim: "true"
      user.attribute: email
      id.token.claim: "true"
      access.token.claim: "true"
      claim.name: email
      jsonType.label: String
    name: email
    protocol: openid-connect
    protocolMapper: oidc-usermodel-property-mapper
    consentRequired: false
  - config:
      multivalued: "true"
      access.token.claim: "true"
      claim.name: "resource_access.${client_id}.roles"
      jsonType.label: String
    name: client roles
    protocol: openid-connect
    protocolMapper: oidc-usermodel-client-role-mapper
    consentRequired: false
  - config:
      userinfo.token.claim: "true"
      user.attribute: firstName
      id.token.claim: "true"
      access.token.claim: "true"
      claim.name: given_name
      jsonType.label: String
    name: given name
    protocol: openid-connect
    protocolMapper: oidc-usermodel-property-mapper
    consentRequired: false
  - config:
      id.token.claim: "true"
      access.token.claim: "true"
      userinfo.token.claim: "true"
    name: full name
    protocol: openid-connect
    protocolMapper: oidc-full-name-mapper
    consentRequired: false
  - config:
      userinfo.token.claim: "true"
```

```
        user.attribute: username
        id.token.claim: "true"
        access.token.claim: "true"
        claim.name: preferred_username
        jsonType.label: String
      name: <username>
      protocol: openid-connect
      protocolMapper: oidc-usermodel-property-mapper
      consentRequired: false
    - config:
        access.token.claim: "true"
        claim.name: "group"
        full.path: "true"
        id.token.claim: "true"
        userinfo.token.claim: "true"
      consentRequired: false
      name: group
      protocol: openid-connect
      protocolMapper: oidc-group-membership-mapper
    - config:
        multivalued: 'true'
        id.token.claim: 'true'
        access.token.claim: 'true'
        userinfo.token.claim: 'true'
        usermodel.clientRoleMapping.clientId:  'automation-hub'
        claim.name: client_roles
        jsonType.label: String
      name: client_roles
      protocolMapper: oidc-usermodel-client-role-mapper
      protocol: openid-connect
    - config:
        id.token.claim: "true"
        access.token.claim: "true"
        included.client.audience: 'automation-hub'
      protocol: openid-connect
      name: audience mapper
      protocolMapper: oidc-audience-mapper
  roles:
    - name: "hubadmin"
      description: "An administrator role for automation hub"
```

1 使用唯一值替换它。

6. 点 **Create** 并等待进程完成。

部署自动化中心时，您必须使用 "Valid Redirect URI" 和 "Web Origins" 更新客户端，如更新 Red Hat Single Sign-On 客户端中所述。另外，客户端会预先配置令牌映射程序，如果您的身份验证供应商未向 Red Hat SSO 提供组数据，则必须更新组映射来反映传递的信息。这通常由用户属性实现。

7.4. 创建 KEYCLOAK 用户

此流程会创建一个有 **hubadmin** 角色的 Keycloak 用户，它可以使用 Super Administration 权限登录到自动化中心。

流程

1. 进入 **Operator → Installed Operators**。
2. 选择 Red Hat Single Sign-On Operator 项目。
3. 选择 **Keycloak Realm** 选项卡，再点 **Create Keycloak User**。
4. 在 **Keycloak User** 表单中，选择 **YAML** 视图。
5. 使用以下内容替换默认 YAML 文件：

```
apiVersion: keycloak.org/v1alpha1
kind: KeycloakUser
metadata:
  name: hubadmin-user
  labels:
    app: sso
    realm: ansible-automation-platform
  namespace: rh-sso
spec:
  realmSelector:
    matchLabels:
      app: sso
      realm: ansible-automation-platform
  user:
    username: hub_admin
    firstName: Hub
    lastName: Admin
    email: hub_admin@example.com
    enabled: true
    emailVerified: false
    credentials:
      - type: password
        value: <ch8ngeme>
    clientRoles:
      automation-hub:
        - hubadmin
```

6. 点 **Create** 并等待进程完成。

创建用户时，Operator 会创建一个包含用户名和密码的 Secret，格式为：**credential-<realm name>-<username>-<namespace>**。在本例中，凭据名为 **credentials-ansible-automation-platform-hub-admin-rh-sso**。创建用户时，Operator 不会更新用户密码。密码的改变不会反映在 Secret 中。

7.5. 安装 ANSIBLE AUTOMATION PLATFORM OPERATOR

流程

1. 进入 **Operator → Operator Hub**，搜索 Ansible Automation Platform Operator。
2. 选择 Ansible Automation Platform Operator 项目。
3. 点 Operator 标题。

4. 点 **Install**。
5. 选择要将 Operator 安装到的项目。红帽建议使用 Operator 推荐的命名空间名称。
   a. 如果要将 Operator 安装到推荐的项目里，请从下拉菜单中选择 **Create Project**。
   b. 输入项目名称。
   c. 点 **Create**。
6. 点 **Install**。
7. 安装 Operator 后，点 **View Operator**。

7.6. 创建 RED HAT SINGLE SIGN-ON 连接 SECRET

使用这个流程为 Red Hat Single Sign-On 创建连接 secret。

流程

1. 进入 **https://<sso_host>/auth/realms/ansible-automation-platform**。
2. 复制 **public_key** 值。
3. 在 OpenShift Web UI 中，进入到 **Workloads → Secrets**。
4. 选择 **ansible-automation-platform** 项目。
5. 点 **Create**，然后选择 **From YAML**。
6. 编辑以下 YAML 以创建 secret

```
apiVersion: v1
kind: Secret
metadata:
  name: automation-hub-sso                    (1)
  namespace: ansible-automation-platform
type: Opaque
stringData:
  keycloak_host: "keycloak-rh-sso.apps-crc.testing"
  keycloak_port: "443"
  keycloak_protocol: "https"
  keycloak_realm: "ansible-automation-platform"
  keycloak_admin_role: "hubadmin"
  social_auth_keycloak_key: "automation-hub"
  social_auth_keycloak_secret: "client-secret"   (2)
  social_auth_keycloak_public_key: >-            (3)
```

(1) 创建自动化中心实例时使用此名称。

(2) 如果在为自动化中心创建 Keycloak 客户端时更改 secret，请务必更改此值以匹配。

(3) 输入在安装 Ansible Automation Platform Operator 中复制的 **public_key** 值。

7. 点 **Create** 并等待进程完成。

7.7. 使用 OPERATOR 安装自动化中心

按照以下流程使用 Operator 安装自动化中心。

流程

1. 进入 **Operator → Installed Operators**。
2. 选择 Ansible Automation Platform。
3. 选择 Automation hub 选项卡，再点 **Create Automation hub**。
4. 选择 **YAML 视图**。YAML 应该类似于：

```
apiVersion: automationhub.ansible.com/v1beta1
kind: AutomationHub
metadata:
  name: private-ah                    (1)
  namespace: aap
spec:
  sso_secret: automation-hub-sso      (2)
  pulp_settings:
    verify_ssl: false
  route_tls_termination_mechanism: Edge
  ingress_type: Route
  loadbalancer_port: 80
  file_storage_size: 100Gi
  image_pull_policy: IfNotPresent
  replicas: 1                         (3)
  web_replicas: N
  task_replicas: N
  file_storage_access_mode: ReadWriteMany
  content:
    log_level: INFO
    replicas: 2
  postgres_storage_requirements:
    limits:
      storage: 50Gi
    requests:
      storage: 8Gi
  api:
    log_level: INFO
    replicas: 1
  postgres_resource_requirements:
    limits:
      cpu: 1000m
      memory: 8Gi
    requests:
      cpu: 500m
      memory: 2Gi
  loadbalancer_protocol: http
  resource_manager:
    replicas: 1
  worker:
    replicas: 2
```

1. 将 metadata.name 设置为要用于实例的名称。
2. 将 spec.sso_secret 设置为创建一个 Secret 来保存 Red Hat Single Sign On 连接详情 过程中创建的 secret 名称。
3. 分别使用 **web_replicas** 或 **task_replicas** 来扩展副本或缩减副本，其中 N 代表您要创建的副本数。或者，您也可以使用 **副本** 在这两个部署中扩展所有容器集。详情请参阅 独立扩展 Web 和任务 Pod。

注意

此 YAML 关闭 SSL 验证 (**ssl_verify: false**)。如果对于 OpenShift 没有使用自签名的证书，则这个设置可以被忽略。

5. 点 **Create** 并等待进程完成。

7.8. 确定自动化中心路由

使用以下步骤确定 hub 路由。

流程

1. 进入到 **Networking → Routes**。
2. 选择您用于安装的项目。
3. 复制 **private-ah-web-svc** 服务的位置。如果您在创建自动化中心实例时使用不同的名称，则服务名称会有所不同。这用于在以后更新 Red Hat Single Sign-On 客户端。

7.9. 更新 RED HAT SINGLE SIGN-ON 客户端

在自动化中心已被安装，并且您知道实例的 URL，您需要更新 Red Hat Single Sign-On，设置 Valid Redirect URI 和 Web Origins。

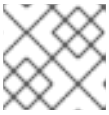
流程

1. 进入 **Operator → Installed Operators**。
2. 选择 RH-SSO 项目。
3. 点 **Red Hat Single Sign-On Operator**。
4. 选择 **Keycloak Client**。
5. 点 automation-hub-client-secret 客户端。
6. 选择 **YAML**。
7. 更新 Client YAML 以添加 Valid Redirect URI 和 Web Origins 设置。

```
redirectUris:
  - 'https://private-ah-ansible-automation-platform.apps-crc.testing/*'
webOrigins:
```

```
- 'https://private-ah-ansible-automation-platform.apps-crc.testing'
```

字段	描述
redirectURIs	这是确定 Automation Hub Route 中确定的位置。确定在 **redirectUris** 设置的末尾添加了 /*。
webOrigins	这是确定 Automation Hub Route 中确定的位置。



注意

在输入这些设置时，确保缩进正确。

8. 点击 **Save**。

验证连接

1. 进入自动化中心路由。
2. 输入 **hub_admin** 用户凭证并登录。
3. Red Hat Single Sign-On 处理身份验证并重定向到自动化中心。

7.10. 其他资源

- 如需有关在 OpenShift Container Platform 上运行 Operator 的更多信息，请参阅 OpenShift Container Platform 产品文档中的在 OpenShift Container Platform 中使用 Operator。

第 8 章 将 RED HAT ANSIBLE AUTOMATION PLATFORM 迁移到 ANSIBLE AUTOMATION PLATFORM OPERATOR

将 Red Hat Ansible Automation Platform 部署迁移到 Ansible Automation Platform Operator 允许您利用 Kubernetes 原生 Operator 所提供的优势，包括简化的升级和 Red Hat Ansible Automation Platform 部署的完整生命周期支持。

使用以下步骤将以下任何部署迁移到 Ansible Automation Platform Operator：

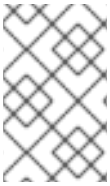
- 基于虚拟机的 Ansible Tower 3.8.6 安装、自动化控制器或自动化中心
- Ansible Tower 3.8.6 的 Openshift 实例 (Ansible Automation Platform 1.2)

8.1. 迁移考虑

如果您要从 OpenShift Container Platform 3 上的 Ansible Automation Platform 1.2 升级到 OpenShift Container Platform 4 上的 Ansible Automation Platform 2.x，您必须置备一个全新的 OpenShift Container Platform 版本 4 集群，然后将 Ansible Automation Platform 迁移到新集群。

8.2. 准备迁移

在将当前的 Ansible Automation Platform 部署迁移到 Ansible Automation Platform Operator 之前，您需要备份您的现有数据，为 secret 密钥和 postgresql 配置创建 k8s secret。



注意

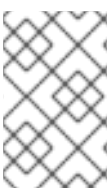
如果要迁移自动化控制器和自动化中心实例，请重复 创建 secret 密钥 secret 的步骤并为这两者 创建一个 postgresql 配置 secret，然后继续 将数据迁移到 Ansible Automation Platform Operator。

8.2.1. 迁移到 Ansible Automation Platform Operator

先决条件

要将 Ansible Automation Platform 部署迁移到 Ansible Automation Platform Operator，您必须有以下几项：

- Secret 密钥 secret
- PostgreSQL 配置
- 新 OpenShift 集群上命名空间的基于角色的访问控制
- 新的 OpenShift 集群必须能够连接到前面的 PostgreSQL 数据库



注意

您可以在初始 Red Hat Ansible Automation Platform 安装前将 secret 密钥信息存储在清单文件中。如果您无法记住您的 secret 密钥或发现您的清单文件时遇到问题，请通过红帽客户门户网站联系 Ansible 支持。

在从 Ansible Automation Platform 2.x 或更早版本迁移数据前，您必须备份您的数据以防丢失。要备份数据，请执行以下操作：

流程

1. 登录您当前的部署项目。
2. 运行 **setup.sh** 来创建当前数据或部署的备份：
   对于版本 2.x 或更早版本的前文部署：

```
$ ./setup.sh -b
```

   对于版本 2.0 之前的 OpenShift 部署（非 Operator 部署）：

```
./setup_openshift.sh -b
```

8.2.2. 创建 secret 密钥 secret

要将数据迁移到 OpenShift Container Platform 上的 Ansible Automation Platform Operator，您必须创建一个 secret 密钥，该 secret 键与清单文件在初始安装过程中定义的 secret 键匹配。否则，迁移后，迁移的数据将保持加密且不可用。

流程

1. 在之前安装中用于部署 Ansible Automation Platform 的清单文件中找到旧的 secret 密钥。
2. 为您的 secret 密钥创建一个 yaml 文件：

```
apiVersion: v1
kind: Secret
metadata:
  name: <resourcename>-secret-key
  namespace: <target-namespace>
stringData:
  secret_key: <old-secret-key>
type: Opaque
```

3. 将 secret key yaml 应用到集群：

```
oc apply -f <secret-key.yml>
```

8.2.3. 创建 postgresql 配置 secret

要成功迁移，您必须为现有部署提供数据库的访问权限。

流程

1. 为您的 postgresql 配置 secret 创建 yaml 文件：

```
apiVersion: v1
kind: Secret
metadata:
  name: <resourcename>-old-postgres-configuration
  namespace: <target namespace>
stringData:
  host: "<external ip or url resolvable by the cluster>"
```

```
  port: "<external port, this usually defaults to 5432>"
  database: "<desired database name>"
  username: "<username to connect as>"
  password: "<password to connect with>"
type: Opaque
```

2. 将 postgresql 配置 yaml 应用到集群：

```
oc apply -f <old-postgres-configuration.yml>
```

8.2.4. 验证网络连接

要确保成功迁移数据，请验证您是否有从新操作器部署到旧部署数据库的网络连接。

先决条件

记录现有部署的主机和端口信息。此信息位于位于 conf.d 目录中的 postgres.py 文件中。

流程

1. 创建一个 yaml 文件来验证新部署和旧部署数据库之间的连接：

```
apiVersion: v1
kind: Pod
metadata:
   name: dbchecker
spec:
  containers:
    - name: dbchecker
      image: registry.redhat.io/rhel8/postgresql-13:latest
      command: ["sleep"]
      args: ["600"]
```

2. 将 connection checker yaml 文件应用到您的新项目更新中：

```
oc project ansible-automation-platform
oc apply -f connection_checker.yaml
```

3. 验证连接检查程序 pod 是否正在运行：

```
oc get pods
```

4. 连接到 pod shell：

```
oc rsh dbchecker
```

5. 在 pod 中打开 shell 会话后，验证新项目是否可以连接到您的旧项目集群：

```
pg_isready -h <old-host-address> -p <old-port-number> -U awx
```

示例

```
<old-host-address>:<old-port-number> - accepting connections
```

8.3. 将数据迁移到 ANSIBLE AUTOMATION PLATFORM OPERATOR

设置 secret 密钥后，postgresql 凭证后，验证网络连接并安装 Ansible Automation Platform Operator，您必须先创建一个自定义资源控制器对象，然后才能迁移数据。

8.3.1. 创建 AutomationController 对象

使用以下步骤创建 AutomationController 自定义资源对象。

流程

1. 登录到 **Red Hat OpenShift Container Platform**。
2. 进入到 **Operators → Installed Operators**。
3. 选择项目命名空间中安装的 Ansible Automation Platform Operator。
4. 选择 **Automation Controller** 选项卡。
5. 点 **Create AutomationController**。
6. 输入新部署的名称。
7. 在 **高级配置** 中，执行以下操作：
   a. 从 **Admin Password Secret** 列表中，选择您的 secret 密钥 secret。
   b. 从 **Database Configuration Secret** 列表中，选择 postgres 配置 secret。
8. 点 **Create**。

8.3.2. 创建 AutomationHub 对象

使用以下步骤创建 AutomationHub 自定义资源对象。

流程

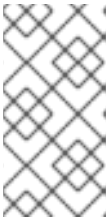
1. 登录到 **Red Hat OpenShift Container Platform**。
2. 进入到 **Operators → Installed Operators**。
3. 选择项目命名空间中安装的 Ansible Automation Platform Operator。
4. 选择 **Automation Hub** 选项卡。
5. 点 **Create AutomationHub**。
6. 输入新部署的名称。
7. 在 **Advanced configurations** 中，选择您的 secret key secret 和 postgres configuration secret。
8. 点 **Create**。

8.4. 迁移后清理

数据迁移完成后，您必须删除所有不再需要的实例组。

流程

1. 使用您在迁移过程中创建的密码以管理员身份登录到 Red Hat Ansible Automation Platform。



注意

注：如果您在迁移过程中没有创建管理员密码，则会为您自动创建密码。要找到此密码，访问您的项目，选择 **Workloads → Secrets**，打开 controller-admin-password。您可以从那里复制密码并将其粘贴到 Red Hat Ansible Automation Platform 密码字段中。

2. 选择 **Administration → Instance Groups**。
3. 选择除 controlplane 和 default 以外的所有实例组。
4. 单击 **Delete**。

第 9 章 在 OPENSHIFT CONTAINER PLATFORM 上升级 ANSIBLE AUTOMATION PLATFORM OPERATOR

Ansible Automation Platform Operator 简化了 OpenShift Container Platform 环境中的新 Red Hat Ansible Automation Platform 实例的安装、升级和部署。

9.1. 升级注意事项

Red Hat Ansible Automation Platform 版本 2.0 是 Ansible Automation Platform Operator 的第一个版本。如果您要从版本 2.0 升级，请继续 升级 Ansible Automation Platform Operator 的步骤。

如果您使用不是由要升级到的 Red Hat Ansible Automation Platform 版本支持的 OpenShift Container Platform 版本，则必须在升级前将 OpenShift Container Platform 集群升级到受支持的版本。

请参阅 Red Hat Ansible Automation Platform 生命周期，以确定所需的 OpenShift Container Platform 版本。

有关升级集群的详情，请参考 更新集群。

9.2. 先决条件

要升级到较新版本的 Ansible Automation Platform Operator，建议您进行以下操作：

- 创建 AutomationControllerBackup 和 AutomationHubBackup 对象。有关此帮助，请参阅创建 Red Hat Ansible Automation Platform 备份资源
- 有关升级和任何中间版本的新 Ansible Automation Platform 版本，请参阅发行注记。

9.3. 升级 ANSIBLE AUTOMATION PLATFORM OPERATOR

要升级到 OpenShift Container Platform 上 Ansible Automation Platform Operator 的最新版本，请执行以下操作：

操作过程

1. 登录 OpenShift Container Platform。
2. 进入到 **Operators → Installed Operators**。
3. 选择 **Subscriptions** 选项卡。
4. 在 **Upgrade status** 下，点 **Upgrade Available**。
5. 点 **Preview InstallPlan**。
6. 点 **Approve**。

第 10 章 在 ANSIBLE AUTOMATION PLATFORM OPERATOR 中添加执行节点

您可以通过下载并安装安装捆绑包来启用带有执行节点的 Ansible Automation Platform Operator。

先决条件

- 自动化控制器实例
- 已安装 receptor 集合软件包
- **ansible-runner** 软件包已安装

流程

1. 登录到 Red Hat Ansible Automation Platform。
2. 在导航面板中，选择 **Administration → Instances**。
3. 点**Add**。
4. 在 **Host Name** 字段中输入虚拟机名称。
5. 可选：在 **Listener Port** 字段中输入端口号。
6. 点击 **Save**。
7. 点 **Install Bundle** 旁边的下载图标 。这将开始下载，记录您保存文件的位置
8. 解压 gz 文件。

注意

要运行 **install_receptor.yml** playbook，您需要从 Ansible Galaxy 安装 receptor 集合： **Ansible-galaxy collection install -r requirements.txt**

9. 使用您的用户名和 SSH 私钥文件更新 playbook。请注意，**ansible_host** 会预先填充您之前输入的主机名。

```
all:
  hosts:
    remote-execution:
      ansible_host: example_host_name
      ansible_user: <username> #user provided
      Ansible_ssh_private_key_file: ~/.ssh/id_example
```

10. 打开一个终端，再前往保存 playbook 的目录。
11. 安装捆绑包运行：

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

12. 现在，可以通过下载并重新运行您创建的实例的 playbook 来升级执行节点。

验证

要验证您的 playbook 是否在新节点上正确运行，请运行以下命令：

```
watch podman ps
```

其他资源

- 有关管理实例组的更多信息，请参阅自动化控制器 用户指南中的管理实例组部分。

第 11 章 ANSIBLE AUTOMATION PLATFORM RESOURCE OPERATOR

11.1. 资源 OPERATOR 概述

Resource Operator 是一个自定义资源(CR)，您可以在创建自动化控制器部署后进行部署。使用 Resource Operator，您可以使用 YAML 文件定义项目、作业模板和清单。然后，自动化控制器使用这些 YAML 文件来创建这些资源。您可以通过 **Form 视图** 创建 YAML，该视图会提示您输入 YAML 代码的键和值。或者，若要直接使用 YAML，您可以选择 **YAML 视图**。

Resource Operator 当前提供了两个自定义资源：

- AnsibleJob：在 Kubernetes secret 中指定的自动化控制器实例中启动作业（自动化控制器主机 URL、令牌）。
- jobTemplate：在指定的自动化控制器实例中创建一个作业模板。

11.2. 使用资源 OPERATOR

在用户创建对象前，Resource Operator 本身不会进行任何操作。用户创建 **AutomationControllerProject** 或 **AnsibleJob** 资源后，Resource Operator 将开始处理该对象。

先决条件

- 安装您选择的基于 Kubernetes 的集群。
- 使用 **automation-controller-operator** 部署自动化控制器。

在集群中安装 **automation-controller-resource-operator** 后，您必须使用自动化控制器实例的连接信息创建一个 Kubernetes (k8s) secret。然后，您可以使用 Resource Operator 创建一个 k8s 资源来管理自动化控制器实例。

11.3. 将 RESOURCE OPERATOR 连接到自动化控制器

要将 Resource Operator 与自动化控制器连接，您需要使用自动化控制器实例的连接信息创建一个 k8s secret。

流程

在自动化控制器 UI 中为您的用户创建 OAuth2 令牌：

1. 在导航面板中，选择 **Access → Users**。
2. 选择您要为其创建令牌的用户名。
3. 单击 **Tokens**，然后单击 **Add**。
4. 您可以将 **应用程序** 留空。添加描述，并为 **Scope** 选择 **Read** 或 **Write**。

另外，您可以使用 **create_oauth2_token** manage 命令在命令行中创建 OAuth2 令牌：

```
$ controller-manage create_oauth2_token --user example_user
New OAuth2 token for example_user: j89ia8OO79te6IAZ97L7E8bMgXCON2
```

注意

确保您在创建令牌时提供有效的用户。否则，您会收到一条错误消息，您在未指定用户的情况下尝试发出命令，或者提供的用户名不存在。

11.4. 为 RESOURCE OPERATOR 创建自动化控制器连接 SECRET

要使连接信息可供 Resource Operator 使用，请使用 token 和 host 值创建一个 k8s secret。

流程

1. 以下是连接 secret 的 YAML 示例。将以下示例保存到文件中，如 **automation-controller-connection-secret.yml**。

```
apiVersion: v1
kind: Secret
metadata:
  name: controller-access
  type: Opaque
stringData:
  token: <generated-token>
  host: https://my-controller-host.example.com/
```

2. 使用您的主机和令牌值编辑该文件。

3. 运行 **kubectl create** 命令将其应用到集群：

```
kubectl create -f controller-connection-secret.yml
```

11.5. 创建 ANSIBLEJOB

通过创建 AnsibleJob 资源，在自动化控制器上启动自动化作业。

流程

1. 指定您要启动的连接 secret 和作业模板。

```
apiVersion: tower.ansible.com/v1alpha1
kind: AnsibleJob
metadata:
  generateName: demo-job-1 # generate a unique suffix per 'kubectl create'
spec:
  connection_secret: controller-access
  job_template_name: Demo Job Template
```

2. 配置作业的实时功能，如清单、额外变量和时间。

```
spec:
  connection_secret: controller-access
  job_template_name: Demo Job Template
  inventory: Demo Inventory                  # Inventory prompt on launch needs to be enabled
  runner_image: quay.io/ansible/controller-resource-runner
  runner_version: latest
```

```
  job_ttl: 100
  extra_vars:                          # Extra variables prompt on launch needs to be enabled
    test_var: test
  job_tags: "provision,install,configuration"  # Specify tags to run
  skip_tags: "configuration,restart"           # Skip tasks with a given tag
```

注意

如果要配置它们，您必须在启动时启用清单和额外变量的提示。要启用 **启动时提示，在** 自动化控制器 UI 中：从 **Resources → Templates** 页面中，选择模板并选择 **Inventory** 和 **Variables** 部分旁边的 **Prompt on launch** 复选框。

3. 通过指定 **workflow_template_name** 而不是 **job_template_name**，使用 AnsibleJob 对象启动工作流作业模板：

```
apiVersion: tower.ansible.com/v1alpha1
kind: AnsibleJob
metadata:
  generateName: demo-job-1 # generate a unique suffix per 'kubectl create'
spec:
  connection_secret: controller-access
  workflow_template_name: Demo Workflow Template
```

11.6. 创建 JOBTEMPLATE

- 通过创建 JobTemplate 资源，在自动化控制器上创建作业模板：

```
apiVersion: tower.ansible.com/v1alpha1
kind: JobTemplate
metadata:
  name: jobtemplate-4
spec:
  connection_secret: controller-access
  job_template_name: ExampleJobTemplate4
  job_template_project: Demo Project
  job_template_playbook: hello_world.yml
  job_template_inventory: Demo Inventory
```