Red Hat Ansible Automation Platform 2.2

Red Hat Ansible Automation Platform 安装指南

本指南为 Red Hat Ansible Automation Platform 支持的安装场景提供了步骤和参考信息

Last Updated: 2023-12-14

Red Hat Ansible Automation Platform 2.2 Red Hat Ansible Automation Platform 安装指南

本指南为 Red Hat Ansible Automation Platform 支持的安装场景提供了步骤和参考信息

法律通告

摘要

提供反馈： 如果您对本文档有任何改进建议，或发现错误，请联系技术支持，使用 Docs组件在 Ansible Automation PlatformJIRA 项目中创建一个问题。

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。

本指南帮助您了解安装 Ansible Automation Platform 后的安装要求和流程。本文档已更新，以包含 Ansible Automation Platform 最新版本的信息。

使开源包含更多

红帽致力于替换我们的代码、文档和 Web 属性中存在问题的语言。我们从这四个术语开始：master、slave、黑名单和白名单。由于此项工作十分艰巨，这些更改将在即将推出的几个发行版本中逐步实施。有关更多详情，请参阅我们的首席技术官 Chris Wright 提供的消息。

第 1 章 规划 RED HAT ANSIBLE AUTOMATION PLATFORM 安装

Red Hat Ansible Automation Platform 在 Red Hat Enterprise Linux 和 Red Hat OpenShift 上都被支持。使用本指南规划在 Red Hat Enterprise Linux 上安装 Red Hat Ansible Automation Platform。

要在 Red Hat OpenShift Container Platform 环境中安装 Red Hat Ansible Automation Platform，请参阅在 OpenShift Container Platform 上部署 Red Hat Ansible Automation Platform Operator 。

1.1. RED HAT ANSIBLE AUTOMATION PLATFORM 系统要求

在规划 Red Hat Ansible Automation Platform 安装并设计适合您的用例的自动化网格拓扑时，请使用此信息。

您的系统必须满足以下最低系统要求才能安装和运行 Red Hat Ansible Automation Platform。

表 1.1. 基本系统

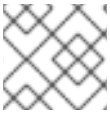
	必填	备注
订阅	有效的 Red Hat Ansible Automation Platform	
OS	Red Hat Enterprise Linux 8.4 或更高版本的 64 位版本(x86)	Red Hat Ansible Automation Platform 在 OpenShift 上也支持，请参阅在 OpenShift Container Platform 上部署 Red Hat Ansible Automation Platform Operator。
Ansible	2.11 版本（要安装的）	Ansible Automation Platform 附带包含 ansible-core 2.13 的执行环境。
Python	3.8 或更高版本	

在使用项目更新和集合时，需要满足以下条件：

- 确保以下域名是防火墙或代理的允许列表一部分，以便能成功连接并从自动化中心或 Galaxy 服务器下载集合：
  - **galaxy.ansible.com**
  - **cloud.redhat.com**
  - **console.redhat.com**
  - **sso.redhat.com**
- 在使用自签名证书或红帽域时，必须禁用 SSL 检查。

1.1.1. 自动化控制器

自动化控制器是一种分布式系统，不同的软件组件可以并置或部署到多个不同计算节点上。在安装程序中，节点类型控制、混合、执行和跃点作为抽象层提供，以帮助用户设计适合其用例的拓扑。下表提供了节点大小的建议：



注意

在控制和混合节点上，为执行环境存储分配至少 20 GB 的 **/var/lib/awx**。

执行节点	**必需**	**备注**
RAM	16 GB	• 建议 1500 或更多的存储卷 IOPS。
CPU	4	• 运行自动化。增加内存和 CPU 以增加容量来运行更多分叉
控制节点	**必需**	**备注**
RAM	16 GB	
CPU	4	• 处理事件并运行集群作业，包括项目更新和清理作业。增加 CPU 和内存有助于处理作业事件。
混合节点	**必需**	**备注**
RAM	16 GB	• 执行和控制节点的 RAM 上的备注也适用于此节点类型。
CPU	4	• 运行自动化和集群作业。有关执行和控制节点的 CPU 的备注也适用于此节点类型。
hop 节点	**必需**	**备注**
RAM	16 GB	

CPU	4	• 用于将流量从自动化网格的一部分路由到另一部分（例如，可以是 bastion 主机到另一网络）。RAM 可能会影响吞吐量，CPU 活动较低。网络带宽和延迟通常比 RAM/CPU 更重要。
磁盘：服务节点	40 GB 专用硬盘空间	• **自动化控制器**：至少 20 GB 的专用的 **/var/** 用于文件和工作目录存储 • 存储卷的最低基础线评级应该是 1500 IOPS。 • 项目存储在控制和混合环境中，作业持续时间也存储在执行节点上。如果集群有很多大型项目，请考虑在 /var/lib/awx/projects 中使用两倍的 GB，以避免磁盘空间错误。
数据库节点	**必需**	**备注**
RAM	16 GB	
CPU	4	
磁盘	20 GB 专用硬盘空间	• 最小专用硬盘空间为 20 GB • 建议大于 150 GB • 存储卷的基础线 IOPS 评级应该比较高（1500 或更高）。
浏览器	当前支持的 Mozilla FireFox 或 Google Chrome 版本	
数据库	PostgreSQL 版本 13	

其他资源

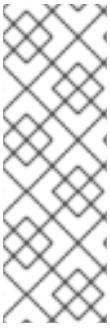
- 要授权使用自动化控制器，请参阅导入订阅。

1.1.2. Automation hub

通过自动化中心，您可以从 Red Hat Ansible 和认证合作伙伴发现并使用新的认证自动化内容。在 Ansible Automation Hub 上，您可以发现和管理由红帽及其合作伙伴开发的自动化内容的 Ansible 集合，用于云自动化、网络自动化和安全自动化等用例。

自动化中心有以下系统要求：

必填		备注
RAM	最小 8 GB	• 8 GB RAM（Vagrant trial 版本安装的最小和推荐值） • 8 GB RAM（外部独立 PostgreSQL 数据库最小值） • 有关您配置中基于 fork 的容量，请查看其他资源
CPU	最少 2 个	• 有关您配置中基于 fork 的容量，请查看其他资源
磁盘：服务节点	60 GB 专用硬盘空间	• 存储卷的最低基础线评级应该是 1500 IOPS。
数据库节点	**必需**	**备注**
RAM	16 GB	
CPU	4	
磁盘	20 GB 专用硬盘空间	• 最小专用硬盘空间为 20 GB • 建议大于 150 GB • 存储卷的基础线 IOPS 评级应该比较高（1500 或更高）。
浏览器	当前支持的 Mozilla FireFox 或 Google Chrome 版本	
数据库	PostgreSQL 版本 13	



注意

- 所有自动化控制器数据都存储在数据库中。通过管理的主机数量、作业运行数量、事实缓存中存储的 fact 数量以及单个作业中的任务数量，数据库存储会增加。例如，一个 playbook 在 250 个主机中每小时运行一次（一天 24 次），20 个任务每周会在数据库中存储超过 800000 个事件。
- 如果没有为数据库保留足够空间，旧的作业和 fact 将需要定期清理。如需更多信息，请参阅*自动控制器管理指南*中的管理作业

Amazon EC2

- 实例大小为 m5.large 或更大
- 如果超过 100 个主机，则实例大小为 m4.xlarge 或更于

有关 Red Hat Ansible Automation Platform 要求的额外备注

- 实际 RAM 的要求取决于同时管理的主机自动化控制器数量（这由作业模板或系统 **ansible.cfg** 文件中的 **forks** 参数控制）。为避免可能的资源冲突，Ansible 建议每 10 个 fork 需要 1 GB 内存再加上 2GB 保留用于自动化控制器，请参阅 Automation controller Capacity Determination and Job Impact。如果 **fork** 设为 400，则建议使用 42 GB 内存。
- 自动化控制器主机检查 **umask** 是否已设置为 0022。如果没有，则设置会失败。设置 **umask=0022** 以避免出现这个错误。
- 可以处理更多主机，但如果 fork 数量小于主机总数，则需要在主机间通过更多。使用滚动更新或使用内置于自动化控制器的调配回调系统时，可以避免这些 RAM 限制，其中每个请求配置的系统都会进入队列并尽可能快速处理；或者，当自动化控制器正在生成或部署镜像（如 AMI）时，可以避免这些 RAM 限制。所有这些都是管理更大环境的最佳方法。
- 如有疑问，请通过红帽客户门户网站联系 Ansible 支持。https://access.redhat.com/
- 由 Ansible Automation Platform 管理的系统的要求与 Ansible 相同。请参阅 Ansible *用户指南*中的快速入门部分。

PostgreSQL 要求

Red Hat Ansible Automation Platform 使用 PostgreSQL 13。

- 在将 PostgreSQL 用户密码保存到数据库前，会使用 SCRAM-SHA-256 安全散列算法对其进行处理。
- 要确定您的自动化控制器实例是否可以访问数据库，则可以使用 **awx-manage check_db** 命令。

PostgreSQL 配置

另外，您可以将 PostgreSQL 数据库配置为不由 Red Hat Ansible Automation Platform 安装程序管理的独立节点。当 Ansible Automation Platform 安装程序管理数据库服务器时，它会使用通常为大多数工作负载推荐的默认值配置服务器。但是，您可以调整独立数据库服务器节点的这些 PostgreSQL 设置，其中 **ansible_memtotal_mb** 是数据库服务器的总内存大小：

```
max_connections == 1024
shared_buffers == ansible_memtotal_mb*0.3
work_mem == ansible_memtotal_mb*0.03
maintenance_work_mem == ansible_memtotal_mb*0.04
```

-

有关调整 PostgreSQL 服务器的详情，请参阅 PostgreSQL 文档。

虽然 Red Hat Ansible Automation Platform 依赖于 Ansible Playbook，并且在安装自动化控制器前需要安装最新版本的 Ansible，但不再需要手动安装 Ansible。

在新安装时，自动化控制器会安装最新版本的 Ansible 2.2。

如果执行捆绑的 Ansible Automation Platform 安装，安装程序会尝试从捆绑包中安装 Ansible（及其依赖项）。

如果您选择自己手动安装 Ansible，Ansible Automation Platform 安装程序会检测到已安装的 Ansible，并且不会尝试重新安装它。

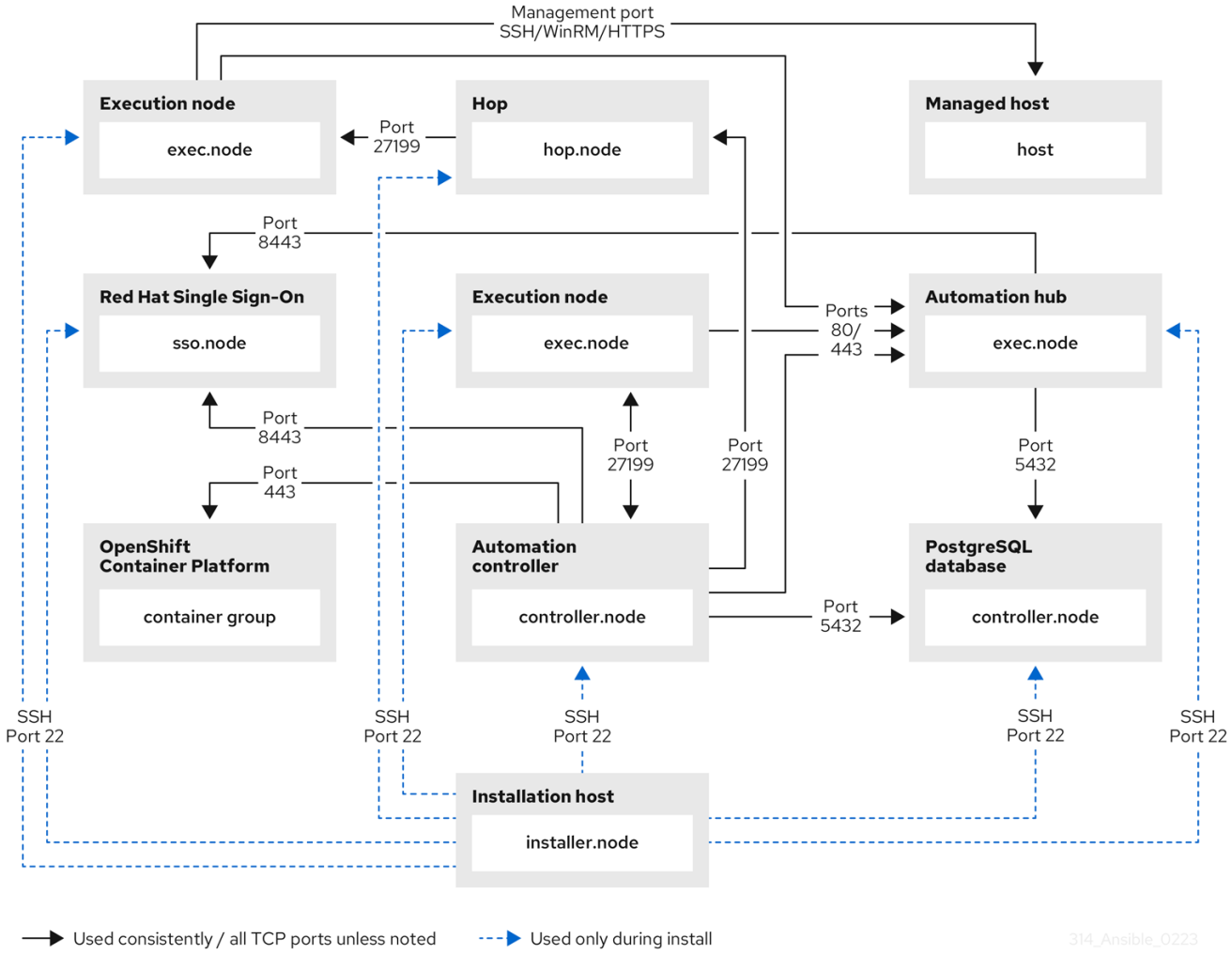
注意

您必须使用软件包管理器（如 **dnf** ）安装 Ansible，必须安装软件包管理器的最新稳定版本，以便 Red Hat Ansible Automation Platform 正常工作。版本 2.2 及更新的版本需要 Ansible 2.11 版本。

1.2. 网络端口和协议

Red Hat Ansible Automation Platform 使用多个端口与其服务进行通信。这些端口必须处于打开状态，并可用于进入到 Red Hat Ansible Automation Platform 服务器的连接，以便它正常工作。确保这些端口可用，且服务器的防火墙没有阻断它们。

以下架构图是完全部署的 Ansible Automation Platform 的示例，其中包含所有可能的组件。



下表提供了每个应用程序所需的默认 Red Hat Ansible Automation Platform 目的地端口。

注意

下列默认目标端口和安装程序清单可以配置。如果您选择对其进行配置以适应您的环境，可能会发现一些行为的变化。

表 1.2. PostgreSQL

端口	协议	服务	方向	安装程序清单变量	需要的目的
22	TCP	SSH	入站和出站	**ansible_port**	在安装过程中远程访问
5432	TCP	Postgres	入站和出站	**pg_port**	默认端口 从控制器到数据库端口的 ALLOW 连接

表 1.3. 自动化控制器

端口	协议	服务	方向	安装程序清单变量	需要的目的
22	TCP	SSH	入站和出站	**ansible_port**	安装
80	TCP	HTTP	入站	**nginx_http_port**	UI/API
443	TCP	HTTPS	入站	**nginx_https_port**	UI/API
5432	TCP	PostgreSQL	入站和出站	**pg_port**	仅在将内部数据库与另一个组件一起使用时才开放。否则，不应该打开此端口 集群中的混合模式
27199	TCP	Receptor	入站和出站	**receptor_listener_port**	对所有用于必须的和自动控制平面集群的所有控制器，ALLOW receptor listener 端口

表 1.4. hop 节点

端口	协议	服务	方向	安装程序清单变量	需要的目的
22	TCP	SSH	入站和出站	**ansible_port**	安装
27199	TCP	Receptor	入站和出站	**receptor_listener_port**	Mesh（网格） 从控制器到 receptor 端口的 ALLOW 连接

表 1.5. 执行节点

端口	协议	服务	方向	安装程序清单变量	需要的目的
22	TCP	SSH	入站和出站	**ansible_port**	安装

端口	协议	服务	方向	安装程序清单变量	需要的目的
27199	TCP	Receptor	入站和出站	**receptor_listener_port**	Mesh - 直接连接到控制器的节点。没有涉及的跃点节点。27199 对于执行节点来说是双向的 从控制器到 receptor 端口的 ALLOW 连接（用于非连接的节点） 从跃点节点到 receptor 端口的 ALLOW 连接（如果通过跃点节点转发）

表 1.6. 控制节点

端口	协议	服务	方向	安装程序清单变量	需要的目的
22	TCP	SSH	入站和出站	**ansible_port**	安装
27199	TCP	Receptor	入站和出站	**receptor_listener_port**	Mesh - 直接连接到控制器的节点。涉及的直接节点。对于执行节点，27199 是双向的 从控制器到 receptor 端口的 ENABLE 连接（用于非连接的节点） 如果通过跃点节点转发，从 hop 节点到 Receptor 端口的ENABLE 连接

端口	协议	服务	方向	安装程序清单变量	需要的目的
443	TCP	Podman	入站	**nginx_https_port**	UI/API

表 1.7. 混合节点

端口	协议	服务	方向	安装程序清单变量	需要的目的
22	TCP	SSH	入站和出站	**ansible_port**	安装
27199	TCP	Receptor	入站和出站	**receptor_listener_port**	Mesh - 直接连接到控制器的节点。没有涉及的跃点节点。27199 对于执行节点来说是双向的 从控制器到 receptor 端口的 ENABLE 连接（用于非连接的节点） 如果通过跃点节点转发，从 hop 节点到 Receptor 端口的ENABLE 连接
443	TCP	Podman	入站	**nginx_https_port**	UI/API

表 1.8. Automation hub

端口	协议	服务	方向	安装程序清单变量	需要的目的
22	TCP	SSH	入站和出站	**ansible_port**	安装
80	TCP	HTTP	入站	固定的值	用户界面
443	TCP	HTTPS	入站	固定的值	用户界面

端口	协议	服务	方向	安装程序清单变量	需要的目的
5432	TCP	PostgreSQL	入站和出站	**automationhub_pg_port**	仅在将内部数据库与另一个组件一起使用时才开放。否则，不应该打开此端口

表 1.9. 服务目录（Services Catalog）

端口	协议	服务	方向	安装程序清单变量	需要的目的
22	TCP	SSH	入站和出站	**ansible_port**	安装
443	TCP	HTTPS	入站	**nginx_https_port**	访问 Service Catalog 用户界面
5432	TCP	PostgreSQL	入站和出站	**pg_port**	仅在使用内部数据库时打开。否则，不应该打开此端口

表 1.10. Red Hat Insights for Red Hat Ansible Automation Platform

URL	需要的目的
http://api.access.redhat.com:443	通用帐户服务、订阅
https://cert-api.access.redhat.com:443	Insights 数据上传
https://cert.cloud.redhat.com:443	清单上传和云连接器连接
https://cloud.redhat.com	访问 Insights 仪表板

表 1.11. Automation Hub

URL	需要的目的
https://console.redhat.com:443	通用帐户服务、订阅
https://sso.redhat.com:443	TCP

URL	需要的目的
https://automation-hub-prd.s3.amazonaws.com	
https://galaxy.ansible.com	Ansible 社区策展的 Ansible 内容
https://ansible-galaxy.s3.amazonaws.com	
https://registry.redhat.io:443	访问由红帽和合作伙伴提供的容器镜像
https://cert.cloud.redhat.com:443	红帽和合作伙伴策展的 Ansible 集合

表 1.12. 执行环境 (EE)

URL	需要的目的
https://registry.redhat.io:443	访问由红帽和合作伙伴提供的容器镜像
cdn.quay.io:443	访问由红帽和合作伙伴提供的容器镜像
cdn01.quay.io:443	访问由红帽和合作伙伴提供的容器镜像
cdn02.quay.io:443	访问由红帽和合作伙伴提供的容器镜像
cdn03.quay.io:443	访问由红帽和合作伙伴提供的容器镜像

重要

镜像清单和文件系统 Blob 直接从 **registry.redhat.io** 提供。但是，从 2023 年 5 月 1 日开始，文件系统 Blob 是从 **quay.io** 提供的。为了避免拉取容器镜像出现问题，您必须启用到列出的 **quay.io** 主机名的出站连接。此更改对专门用于启用到 **registry.redhat.io** 的出站连接的任何防火墙配置进行。在配置防火墙规则时使用主机名而不是 IP 地址。完成此更改后，您可以继续从 **registry.redhat.io** 拉取镜像。要继续拉取红帽容器镜像，您不需要 **quay.io** 登录，或者需要以任何方式直接与 **quay.io** registry 交互，。如需更多信息，请参阅此文档。

1.3. 附加 RED HAT ANSIBLE AUTOMATION PLATFORM 订阅

在安装 Red Hat Ansible Automation Platform 前，您必须在所有节点上附加了有效的订阅。在附加 Ansible Automation Platform 订阅后，您便可访问继续安装所需的需要订阅服务的子资源。

注意

如果您在红帽帐户中启用了简单内容访问模式，则需要附加订阅。启用后，您需要在安装 Ansible Automation Platform 前将您的系统注册到 Red Hat Subscription Management(RHSM)或 Satellite。如需更多信息，请参阅简单内容访问模式。

流程

1. 获取 Red Hat Ansible Automation Platform 订阅的 **pool_id**：

```
# subscription-manager list --available --all | grep "Ansible Automation Platform" -B 3 -A 6
```

示例

subsciption-manager list 命令的输出示例。获取 **Pool ID:** 部分中的 **pool_id** ：

```
Subscription Name: Red Hat Ansible Automation, Premium (5000 Managed Nodes)
  Provides: Red Hat Ansible Engine
  Red Hat Ansible Automation Platform
  SKU: MCT3695
  Contract: ````
  Pool ID: <pool_id>
  Provides Management: No
  Available: 4999
  Suggested: 1
```

2. 附加订阅：

```
# subscription-manager attach --pool=<pool_id>
```

现在，您已将 Red Hat Ansible Automation Platform 订阅附加到所有节点。

验证

- 验证订阅是否已成功附加：

```
# subscription-manager list --consumed
```

故障排除

- 如果您无法找到与 Ansible Automation Platform 安装程序捆绑的特定软件包，或者如果您看到 ***配置信息禁用了*** 存储库，请尝试使用以下命令启用存储库：
  Red Hat Ansible Automation Platform 2.2 for RHEL 8

```
subscription-manager repos --enable ansible-automation-platform-2.2-for-rhel-8-x86_64-rpms
```

  Red Hat Ansible Automation Platform 2.2 for RHEL 9

```
subscription-manager repos --enable ansible-automation-platform-2.2-for-rhel-9-x86_64-rpms
```

1.4. RED HAT ANSIBLE AUTOMATION PLATFORM 平台组件

Red Hat Ansible Automation Platform 由以下组件组成：

Ansible Automation hub

用于认证 Ansible 内容集合内容的存储库。Ansible Automation hub 是红帽及其合作伙伴的集中存储库，用于发布内容，并允许客户发现经过认证的 Ansible 内容集合。Red Hat Ansible 认证的内容为用户提供已经过测试并受红帽支持的内容。

私有自动化中心

私有自动化中心提供断开连接和内部解决方案，用于同步内容。您可以从 Red Hat 云自动化中心同步集合和执行环境镜像，存储和提供自己的自定义自动化集合和执行镜像。您还可以使用 Ansible Galaxy 或其他容器 registry 等其他源向私有自动化中心提供内容。私有自动化中心可以集成到您的企业目录和 CI/CD 管道中。

自动化控制器

使用用户界面 (UI) 和 RESTful 应用程序编程接口 (API)，控制、保护和管理 Ansible 自动化的企业框架。

自动化服务目录

自动化服务目录是 Red Hat Ansible Automation Platform 中的服务。自动化服务目录允许您在各种环境中在 Ansible 自动化控制器中组织和管理产品目录源。

使用自动服务目录，您可以：

- 对单个平台清单应用多级别批准。
- 将您的平台中的产品以产品组合（portfolios）的形式组织内容。
- 选择产品组合与特定用户组共享。
- 设定要执行用户请求的数值的界限。

Automation mesh

自动化网格是一个覆盖网络，旨在通过使用现有网络相互建立对等连接的节点，简化在大型且分散的 worker 集合中的工作分布。

Automation mesh 提供：

- 动态集群容量可独立扩展，允许您以最少的停机时间创建、注册、分组、分组和取消注册节点。
- 控制和执行平面分离，可让您独立于 control plane 容量来缩放 playbook 执行容量。
- 部署选择具有应对延迟、在没有中断中断的情况下可重新配置的部署选择，并在存在中断时动态重新路由以选择不同的路径。
- Mesh 路由更改。
- 包括符合联邦信息处理标准(FIPS)的双向多跃网格通信可能性的连接。

自动执行环境

包含 Ansible 执行引擎和数百个模块解决方案，可帮助用户自动化 IT 环境和流程的所有方面。执行环境自动化了常用的操作系统、基础架构平台、网络设备和云。

Ansible Galaxy

用于查找、重复利用和共享 Ansible 内容的中心。社区提供的 Galaxy 内容采用预打包角色的形式，可帮助启动自动化项目。用于配置基础架构、部署应用和完成其他任务的角色可被丢弃到 Ansible Playbook 中，并立即应用到客户环境。

自动化内容导航器

作为 *文本用户界面* (TUI)，它成为自动化平台的主要命令行界面，涵盖构建内容构建、在执行环境中运行自动化、在 Ansible Automation Platform 中运行自动化以及为未来*集成开发环境* (IDE)提供基础。

1.5. 选择并获取 RED HAT ANSIBLE AUTOMATION PLATFORM 安装程序

根据您的 Red Hat Enterprise Linux 环境互联网连接，选择您需要的 Red Hat Ansible Automation Platform 安装程序。查看以下场景，并确定哪个 Red Hat Ansible Automation Platform 安装程序满足您的需要。

注意

需要有效的红帽客户帐户才能访问红帽客户门户上的 Red Hat Ansible Automation Platform 安装程序下载。

使用互联网访问进行安装

如果您的 Red Hat Enterprise Linux 环境连接到互联网，请选择 Red Hat Ansible Automation Platform 安装程序。使用互联网访问进行安装会检索最新的软件仓库、软件包和依赖项。选择以下方法之一来设置 Ansible Automation Platform 安装程序。

Tarball 安装

1. 进入 https://access.redhat.com/downloads/content/480
2. 为 **Ansible Automation Platform <latest-version> Setup** 点 **Download Now**。
3. 解压文件：

```
$ tar xvzf ansible-automation-platform-setup-<latest-version>.tar.gz
```

RPM 安装

1. 安装 Ansible Automation Platform 安装程序软件包
   v.2.2 for RHEL 8 for x86_64

```
$ sudo dnf install --enablerepo=ansible-automation-platform-2.2-for-rhel-8-x86_64-rpms
ansible-automation-platform-installer
```

   v.2.2 for RHEL 9 for x86-64

```
$ sudo dnf install --enablerepo=ansible-automation-platform-2.2-for-rhel-9-x86_64-rpms
ansible-automation-platform-installer
```

注意

dnf install 启用存储库，因为默认禁用存储库。

使用 RPM 安装程序时，文件位于 **/opt/ansible-automation-platform/installer** 目录下。

在没有互联网访问的情况下安装

如果您无法访问互联网，或者不想从在线存储库安装独立的组件和依赖项，请使用 Red Hat Ansible Automation **PlatformBundle** 安装程序。仍然需要访问 Red Hat Enterprise Linux 软件仓库。所有其他依赖项都包含在 tar 归档中。

1. 进入 https://access.redhat.com/downloads/content/480
2. 为 **Ansible Automation Platform <latest-version> Setup Bundle**点 **Download Now**。
3. 解压文件：

```
$ tar xvzf ansible-automation-platform-setup-bundle-<latest-version>.tar.gz
```

1.6. 关于安装程序清单文件

Red Hat Ansible Automation Platform 使用清单文件，根据您以逻辑方式组织的基础架构中的受管节点或主机列表进行工作。您可以使用 Red Hat Ansible Automation Platform 安装程序清单文件指定您的安装场景，并描述 Ansible 的主机部署。通过使用清单文件，Ansible 可以通过一个命令管理大量主机。清单还可以通过减少您指定的命令行选项数目来更有效地使用 Ansible。

根据您拥有的清单插件，清单文件可以采用多种格式。最常见的格式是 **INI** 和 **YAML**。本文档中列出的清单文件以 INI 格式显示。

清单文件的位置取决于您使用的安装程序。下表显示了可能的位置：

安装程序	位置
捆绑包 tar	**/ansible-automation-platform-setup-bundle-<latest-version>**
非捆绑包 tar	**/ansible-automation-platform-setup-<latest-version>**
RPM	**/opt/ansible-automation-platform/installer**

您可以使用以下命令验证清单中的主机：

```
ansible all -i <path-to-inventory-file. --list-hosts
```

清单文件示例

```
[automationcontroller]
host1.example.com
host2.example.com
Host4.example.com

[automationhub]
host3.example.com

[database]
```

```
Host5.example.com

[all:vars]
admin_password='<password>'

pg_host=''
pg_port=''

pg_database='awx'
pg_username='awx'
pg_password='<password>'

registry_url='registry.redhat.io'
registry_username='<registry username>'
registry_password='<registry password>'
```

清单文件的第一个部分指定 Ansible 可使用的主机或主机组。

1.6.1. 主机和组指南

数据库

- 使用外部数据库时，请确保正确设置清单文件的 **[database]** 部分。
- 要提高性能，请不要在同一服务器上并置数据库和自动化控制器。

Automation hub

- 在 **[automationhub]** 组中添加 Ansible Automation hub 信息。
- 不要在同一节点上安装 Ansible Automation hub 和自动化控制器。
- 为 **[automationhub]** 主机提供可访问 IP 地址或完全限定域名(FQDN)，以确保用户可以从不同节点从 Ansible Automation hub 同步和安装内容。不要使用 **localhost**。

重要

您必须分隔自动化控制器和 Ansible 自动化中心的安装，因为 **[database]** 组无法区分这两个（如果同时安装了这两个组）。

如果您在 **[database]** 和自动化控制器和 Ansible 自动化中心中使用一个值，则它们将使用相同的数据库。

自动化控制器

- 自动化控制器将不会为其使用的数据库配置复制或故障转移功能。自动化控制器应该与已有的任何复制一起工作。

集群安装

- 在升级现有集群时，您还可以重新配置集群来忽略现有的实例或实例组。从清单文件中省略实例或实例组不足以将其从集群中移除。除了从清单文件中省略实例或实例组外，还必须在开始升级前取消置备实例或实例组。请参阅取消置备节点或组。否则，忽略的实例或实例组会继续与集群通信，这可能会在升级过程中造成自动化控制器服务的问题。

- 如果要创建集群安装设置，则必须使用所有实例的主机名或 IP 地址替换 **[localhost]**。自动化控制器、自动化中心和自动服务目录的安装程序不接受 **[localhost]** 所有节点和实例，且实例必须能够使用这个主机名或地址访问任何其他节点。您不能在其中一个节点上使用 localhost **ansible_connection=local**。对所有节点的主机名使用相同的格式。
  因此，这无法正常工作：

```
[automationhub]
localhost ansible_connection=local
hostA
hostB.example.com
172.27.0.4
```

需要使用以下格式：

```
[automationhub]
hostA
hostB
hostC
```

或

```
[automationhub]
hostA.example.com
hostB.example.com
hostC.example.com
```

1.6.2. 取消置备节点或组

您可以使用 Ansible Automation Platform 安装程序取消置备节点和实例组。运行安装程序将删除附加到组中节点的所有配置文件和日志。

注意

您可以取消置备清单中的任何主机，但 **[automationcontroller]** 组中指定的第一个主机除外。

要取消置备节点，请将 **node_state=deprovision** 附加到清单文件中的节点或组。

例如：

从部署中删除单个节点：

```
[automationcontroller]
host1.example.com
host2.example.com
host4.example.com   node_state=deprovision
```

或

从部署中删除整个实例组：

```
[instance_group_restrictedzone]
host4.example.com
```

```
host5.example.com

[instance_group_restrictedzone:vars]
node_state=deprovision
```

1.6.3. 清单变量

示例清单文件的第二部分（在 **[all:vars]** 后面）是安装程序使用的变量列表。使用 **all** 表示变量适用于所有主机。

要将变量应用到特定的主机，请使用 **[hostname:vars]**。例如，**[automationhub:vars]**。

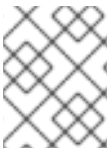
1.6.4. 在清单文件中声明变量的规则

字符串变量的值在引号中声明。例如：

```
pg_database='awx'
pg_username='awx'
pg_password='<password>'
```

在 **:vars** 部分中声明时，INII 值将解释为字符串。例如，**var=FALSE** 创建一个等于 **FALSE** 的字符串。与主机行不同，**:vars** 部分每行只接受一个条目，因此 **=** 后的所有内容都必须是条目的值。主机行接受每行多个 **key=value** 参数。因此，它们需要使用一种方式来指出空格是值的一部分而不是分隔符。包含空格的值可以加上引号（单引号或双引号）。详情请查看 Python shlex 解析规则。

如果 INI 清单中设置的变量值必须是特定的类型（如字符串或布尔值），则始终在任务中使用过滤器指定类型。在消耗变量时，请勿依赖 INI 清单中设置的类型。



注意

考虑将 YAML 格式用于清单源，以避免在变量的实际类型上产生混淆。YAML 清单插件会一致且正确处理变量值。

如果 Ansible 清单文件中的参数值包含特殊字符，如 #、{ 或 }，则必须双重转义（用单引号和双引号包括该值）。

例如，要将 **mypasswordwith#hashsigns** 用作变量 **pg_password** 的值，在 Ansible 主机清单文件中请将其声明为 **pg_password='"mypasswordwith#hashsigns"'**。

1.6.5. 保护清单文件中的 secret

您可以使用 Ansible Vault 加密敏感或机密的变量。但是，对变量名称和变量值进行加密可能会难以查找值的来源。为了缓解这个问题，您可以使用 **ansible-vault encrypt_string** 加密单独的变量，或者加密包含变量的文件。

流程

1. 创建一个标记为 **credentials.yml** 的文件来存储加密的凭证。

```
$ cat credentials.yml

admin_password: my_long_admin_pw
```

```
pg_password: my_long_pg_pw
registry_password: my_long_registry_pw
```

2. 使用 **ansible-vault** 加密 **credentials.yml** 文件。

```
$ ansible-vault encrypt credentials.yml
New Vault password:
Confirm New Vault password:
Encryption successful
```

重要

将加密的 vault 密码存储在安全的地方。

3. 验证 **credentials.yml** 文件是否加密。

```
$ cat credentials.yml
$ANSIBLE_VAULT;1.1;
AES256363836396535623865343163333339613833363064653364656138313534353135303
764646165393765393063303065323466663330646232363065316666310a37306230313337
633963383130303334313534383962613632303761636632623932653062343839613639653
635643365616233313365363661663931386430 0a353239373433313339613465326339313 0
356335653534643565386536316334643438353464323766386235336136663261363433323
131633436393939646132656164333634306335343039356462646330343839663623230330
65383763
```

4. 运行 **setup.sh** 以安装 Ansible Automation Platform 2.2，并传递 **credentials.yml** 和 **--ask-vault-pass** 选项。

```
$ ANSIBLE_BECOME_METHOD='sudo' ANSIBLE_BECOME=True
ANSIBLE_HOST_KEY_CHECKING=False ./setup.sh -e @credentials.yml -- --ask-vault-pass
```

1.6.6. 其他清单文件变量

您可以通过在清单文件中包含额外变量来进一步配置 Red Hat Ansible Automation Platform 安装。这些配置添加了用于管理 Red Hat Ansible Automation Platform 的各种可选功能。使用文本编辑器编辑清单文件来添加这些变量。

清单文件变量的预定义值表可在 附录 A: 清单文件变量 中找到

1.7. 支持的安装场景

红帽支持以下 Red Hat Ansible Automation Platform 安装场景

1.7.1. 带有同一节点上的数据库的独立自动化控制器，或者一个非安装程序管理的数据库

此场景包括在单一计算机上安装自动化控制器，包括 Web 前端、REST API 后端和数据库。它安装 PostgreSQL，并将自动化控制器配置为将其用作其数据库。这被视为标准自动化控制器安装场景。

请参阅在一台机器上安装 Red Hat Ansible Automation Platform 组件 中的*在同一节点上安装带有数据库的自动化控制器*。

1.7.2. 带有外部管理数据库的独立自动化控制器

此场景包括在一台机器上安装自动化控制器服务器，并将与远程 PostgreSQL 实例的通信配置为其数据库。此远程 PostgreSQL 可以是您管理的服务器，也可以由 Amazon RDS 等云服务提供。

请参阅 *在一台机器上安装 Red Hat Ansible Automation Platform 组件* 中的安装带有外部管理的数据库的自动化控制器。

1.7.3. 带有同一节点上的数据库的独立自动化中心，或者一个非安装程序管理的数据库

此场景包括安装自动化中心，包括 Web 前端、REST API 后端和单一计算机上的数据库。它安装 PostgreSQL，并将自动化中心配置为将其用作其数据库。

请参阅 *在一台机器上安装 Red Hat Ansible Automation Platform 组件* 中的在同一节点上安装带有数据库的自动化中心 。

1.7.4. 带有外部管理数据库的独立自动化中心

此场景包括在一台机器上安装自动化中心服务器，并安装由 Red Hat Ansible Automation Platform 安装程序管理的远程 PostgreSQL 数据库。

请参阅 *在一台机器上安装 Red Hat Ansible Automation Platform 组件* 中的在同一节点上安装带有外部数据库的自动化中心 。

1.7.5. 在自动化控制器节点或非安装程序管理数据库上使用数据库进行平台安装

这个场景包括在自动化控制器节点上安装带有数据库的自动化控制器和自动化中心，或者一个非安装程序管理的数据库。

请参阅 *安装 Red Hat Ansible Automation Platform* 中的安装使用自动化控制器节点中的数据库或非安装程序管理的数据库的 Red Hat Ansible Automation Platform。

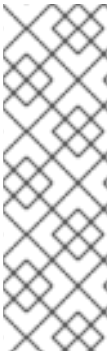
1.7.6. 带有外部管理数据库的平台安装

此场景包括安装自动化控制器和自动化中心，并将与远程 PostgreSQL 实例的通信配置为其数据库。此远程 PostgreSQL 可以是您管理的服务器，也可以由 Amazon RDS 等云服务提供。

请参阅 *安装 Red Hat Ansible Automation Platform* 中的安装带有外部管理的数据库的 Red Hat Ansible Automation Platform。

1.7.7. 使用外部管理的数据库安装多机器集群

此场景包括安装多个自动化控制器节点和一个自动化中心实例，并将与远程 PostgreSQL 实例的通信配置为其数据库。此远程 PostgreSQL 可以是您管理的服务器，也可以由 Amazon RDS 等云服务提供。在这种情况下，所有自动化控制器都是活跃的，并可执行作业，任何节点都可以接收 HTTP 请求。



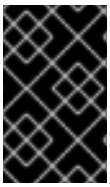
注意

- 在集群设置中运行需要自动化控制器使用外部的PostgreSQL 数据库安装到不是主或从 tower 节点之一的机器上。在带有冗余功能的设置中，远程 PostgreSQL 版本要求为 **PostgreSQL 13**。
  - 有关配置集群设置的更多信息，请参阅集群。
- 为 **[automationhub]** 主机提供一个可访问的 IP 地址，以确保用户可以从不同节点的 Private Automation Hub 同步内容。

请参阅 *多机器集群安装*中的安装带有外部管理数据库的多节点 Red Hat Ansible Automation Platform 。

第 2 章 安装 RED HAT ANSIBLE AUTOMATION PLATFORM

Red Hat Ansible Automation Platform 安装涉及部署自动化控制器和自动化中心。



重要

安装程序不需要用户以 root 身份登录来运行 **./setup.sh**。用户需要正确配置环境变量 **ANSIBLE_BECOME_METHOD** 以使用首选的特权升级方法升级到 root。默认方法是 **sudo**。

这个安装选项包括两个支持的情况：

2.1. 使用自动化控制器节点或非安装程序管理的数据库安装 RED HAT ANSIBLE AUTOMATION PLATFORM

您可以使用以下说明在自动化控制器节点或非安装程序管理的数据库上安装带有数据库的 Red Hat Ansible Automation Platform（自动化控制器和自动化中心）。

2.1.1. 先决条件

- 您已经从 Red Hat Ansible Automation Platform 产品软件 中选择并获取了平台安装程序。
- 您需要在满足基本系统要求的机器上安装。
- 您已将所有软件包更新至 RHEL 节点的当前版本。

警告

如果您在 Ansible Automation Platform 安装前没有完全升级 RHEL 节点，您可能会遇到错误。

- 您已创建了 Red Hat Registry Service Account，按照创建 Registry 服务账户指南 中的说明进行操作。
- 安装 Ansible Automation Platform 需要容器 registry 服务。通过访问容器 registry，您可以将自动化执行环境加载到 Ansible 自动化平台上，为您提供一致、容器化的环境，用于执行 Ansible playbook 和角色。默认情况下，Ansible Automation Platform 使用 **registry.redhat.io**，它需要 Red Hat registry 服务帐户。请参阅创建 Registry 服务账户 指南来创建 registry 服务帐户。
- 您必须在所有节点上配置 NTP 客户端。如需更多信息，请参阅使用 Chrony 配置 NTP 服务器 。

2.1.2. 编辑 Red Hat Ansible Automation Platform 安装程序清单文件

您可以使用 Red Hat Ansible Automation Platform 安装程序清单文件指定您的安装场景。

流程

1. 进入安装程序

a. [Bundled installer]

```
$ cd ansible-automation-platform-setup-bundle-<latest-version>
```

b. [Online installer]

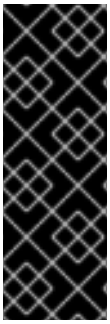
```
$ cd ansible-automation-platform-setup-<latest-version>
```

2. 使用文本编辑器打开**清**单文件。

3. 编辑**清**单文件参数，以指定您的安装场景。请按照以下示例操作。

2.1.3. 自动化控制器节点或非安装程序管理的数据库的清单文件示例

本例描述了如何填充清单文件以安装 Red Hat Ansible Automation Platform。此安装清单文件包含自动化控制器和自动化中心，以及自动化控制器节点或非安装程序管理的数据库上的数据库。



重要

- 您不能在同一节点上安装自动化控制器和自动化 hub。
- 为 **[automationhub]** 主机提供一个可访问的 IP 地址，以确保用户可以从不同节点的 Private Automation Hub 同步内容。
- 在 **registry_username** 和 **registry_password** 中输入 Red Hat Registry Service account 凭证以链接到 Red Hat 容器 registry。

```
[automationcontroller]
controller.acme.org

[automationhub]
automationhub.acme.org

[all:vars]
admin_password='<password>'
pg_host=''
pg_port=''
pg_database='awx'
pg_username='awx'
pg_password='<password>'
pg_sslmode='prefer'  # set to 'verify-full' for client-side enforced SSL

registry_url='registry.redhat.io'
registry_username='<registry username>'
registry_password='<registry password>'

# Automation Hub Configuration
#
automationhub_admin_password='<password>'
automationhub_pg_host='controller.acme.org'
automationhub_pg_port='5432'
automationhub_pg_database='automationhub'
automationhub_pg_username='automationhub'
automationhub_pg_password='<password>'
```

```
automationhub_pg_sslmode='prefer'

# The default install will deploy a TLS enabled Automation Hub.
# If for some reason this is not the behavior wanted one can
# disable TLS enabled deployment.
#
# automationhub_disable_https = False
# The default install will generate self-signed certificates for the Automation
# Hub service. If you are providing valid certificate via automationhub_ssl_cert
# and automationhub_ssl_key, one should toggle that value to True.
#
# automationhub_ssl_validate_certs = False
# SSL-related variables
# If set, this will install a custom CA certificate to the system trust store.
# custom_ca_cert=/path/to/ca.crt
# Certificate and key to install in nginx for the web UI and API
# web_server_ssl_cert=/path/to/tower.cert
# web_server_ssl_key=/path/to/tower.key
# Certificate and key to install in Automation Hub node
# automationhub_ssl_cert=/path/to/automationhub.cert
# automationhub_ssl_key=/path/to/automationhub.key
# Server-side SSL settings for PostgreSQL (when we are installing it).
# postgres_use_ssl=False
# postgres_ssl_cert=/path/to/pgsql.crt
# postgres_ssl_key=/path/to/pgsql.key
```

2.1.4. 设置脚本标志和额外变量

在运行设置脚本以安装自动化控制器时，您还可以传递标志和额外变量：

表 2.1. 标记

参数	描述
-h	显示帮助信息并退出
-i INVENTORY_FILE	到 Ansible 清单文件的路径（默认：**inventory**）
-e EXTRA_VARS	将额外的 Ansible 变量设置为 key=value 或 YAML/JSON
-b	在安装时执行数据库备份
-r	在安装时执行数据库恢复
-k	生成 **并分发** SECRET_KEY。 当使用此参数执行 setup.sh 脚本时，默认会为自动化控制器生成并分发一个新的 secret 密钥，但不用于自动化服务目录。 要为自动化控制器和自动服务目录生成并分发新的 secret 密钥，请指定变量 **rekey_catalog: true**。

使用 **--** 分隔符添加您希望应用的任何 Ansible 参数。例如：**./setup.sh -i my_awesome_inventory.yml -e matburt_is_country_gold=True — -K**。

注意

- 在进行数据库恢复时使用了 **-r**，则会使用默认的恢复目录，除非使用 EXTRA_VARS 提供了一个非默认路径。请参阅以下通过 EXTRA_VAR 指定恢复路径的示例：

```
./setup.sh -e 'restore_backup_file=/path/to/nondefault/location' -r
```

- 您可以通过使用 **-e bundle_install=false** 来强制在线安装：

```
$ ./setup.sh -e bundle_install=false
```

表 2.2. 额外变量

变量	Description	default
upgrade_ansible_with_tower	安装自动化控制器时，请确保 Ansible 也为最新版本	**False**
create_preload_data	安装 Tower 时还创建 Demo Org 、项目 、凭证 、作业模板等。	**True**
bundle_install_folder	在进行捆绑安装时获得捆绑软件仓库的目录	**var/lib/tower-bundle**
nginx_disable_https	禁用使用 nginx 的 HTTPS 数据，这在将 HTTPS 数据卸载到负载均衡器时非常有用	**False**
nginx_disable_hsts	禁用 HSTS web-security policy 机制	**False**
nginx_http_port	nginx 为 HTTP 侦听的端口	**80**
nginx_https_port	nginx 为 HTTPS 侦听的端口	**443**
backup_dir	备份时要使用的临时位置	**/var/backups/tower/**
restore_backup_file	指定要从中恢复的备用文件	None
required_ram	安装 Tower 所需的最小 RAM（应只在测试安装时修改）	**3750**
min_open_fds	最小打开文件描述（应只在测试安装时修改）	None

变量	Description	default
ignore_preflight_errors	忽略 preflight 检查，在安装到一个模板或其他非系统镜像时很有用（覆盖 **required_ram** 和 **min_open_fds**）	**False**
rekey_catalog	当使用此变量设置为 **true** 来执行 **setup.sh** 脚本时，默认会生成一个新的 secret 密钥，并为自动化控制器和自动服务目录发布。	**False**

例子

- 要升级内核：

```
./setup.sh -e upgrade_ansible_with_tower=1
```

- 在 nginx 中禁用 https 处理：

```
./setup.sh -e nginx_disable_https=true
```

- 在从备份文件中恢复时指定非默认路径：

```
./setup.sh -e 'restore_backup_file=/path/to/nondefault/location' -r
```

2.1.5. 运行 Red Hat Ansible Automation Platform 安装程序设置脚本

在完成使用安装私有 Automation Hub 所需的参数更新**清**单文件后，您可以运行设置脚本。

流程

1. 运行 **setup.sh** 脚本

```
$ ./setup.sh
```

安装将开始。

2.1.6. 验证自动化控制器安装

安装完成后，您可以通过使用插入到**清**单文件中的 admin 凭据登录来验证自动化控制器是否已成功安装。

流程

1. 进入**清**单文件中为自动化控制器节点指定的 IP 地址。
2. 使用您在**清**单文件中设置的 Admin 凭据登录。

注意

自动化控制器服务器可从端口 80（https://<TOWER_SERVER_NAME>/）访问，但会被重定向到端口 443，因此端口 443 需要可以被使用。

重要

如果安装失败，且您购买了 Red Hat Ansible Automation Platform 的有效许可证，请通过红帽客户门户网站 https://access.redhat.com/ 联络 Ansible。

当可以成功登录到自动化控制器后，则代表您的 Red Hat Ansible Automation Platform 2.2 安装已完成。

2.1.6.1. 其他自动化控制器配置和资源

请参阅以下资源来探索其他自动化控制器配置。

表 2.3. 配置自动化控制器的资源

Link	描述
自动化控制器快速设置指南	设置自动化控制器并运行第一个 playbook
自动化控制器管理指南	通过客户脚本、管理作业等配置自动化控制器管理。
配置 Red Hat Ansible Automation Platform 的代理支持	使用代理服务器设置自动化控制器
从自动化控制器管理可用性分析和数据收集	管理您与红帽共享的自动化控制器信息
自动化控制器用户指南	更详细地查看自动化控制器功能

2.1.7. 验证自动化中心安装

安装完成后，您可以通过使用插入到**清单**文件中的 admin 凭据登录来验证自动化中心是否已成功安装。

流程

1. 进入**清单**文件中为自动化中心节点指定的 IP 地址。
2. 使用您在**清单**文件中设置的 Admin 凭据登录。

重要

如果安装失败，且您购买了 Red Hat Ansible Automation Platform 的有效许可证，请通过红帽客户门户网站 https://access.redhat.com/ 联络 Ansible。

当可以成功登录到自动化 hub 后，则代表您的 Red Hat Ansible Automation Platform 2.2 安装已完成。

2.1.7.1. 其他自动化中心配置和资源

请参阅以下资源来探索额外的自动化中心配置。

表 2.4. 配置自动化控制器的资源

Link	描述
在私有自动化 hub 中管理用户访问权限	为自动化 hub 配置用户访问
在自动 Hub 中管理红帽认证的集合和 Ansible Galaxy 集合	在自动化 hub 中添加内容
在自动化 hub 中发布专有内容集合	在自动化 hub 中发布内部开发的集合

2.1.8. Ansible Automation Platform 2.2 的下一步是什么

无论您是希望开始使用自动化的新 Ansible Automation Platform 用户，还是希望将旧的 Ansible 内容迁移到最新版本的 Red Hat Ansible Automation Platform，充分利用 Ansible Automation Platform 2.2 的新功能：

2.1.8.1. 将数据迁移到 Ansible Automation Platform 2.2

对于希望完成升级到 Ansible Automation Platform 2.2 的平台管理员，可能需要额外步骤将数据迁移到新实例：

2.1.8.1.1. 从旧的虚拟环境 (venvs) 迁移到自动化执行环境

通过 Ansible Automation Platform 2.2，您可以从自定义 Python 虚拟环境 (venvs) 转换为使用自动化执行环境 - 它是容器化镜像，包括了执行和扩展 Ansible 自动化所需的组件。这包括 Ansible Core、Ansible 内容集合、Python 依赖项、Red Hat Enterprise Linux UBI 8 以及任何其他软件包依赖项。

如果您希望将 venvs 迁移到执行环境，则需要(1)使用 **awx-manage** 命令列出并从原始实例中导出 venvs 列表，然后（2）使用 **ansible-builder** 创建执行环境。如需更多信息，请参阅 升级到自动化执行环境指南 和 创建和恢复执行环境。

2.1.8.1.2. 使用 Ansible Builder 迁移至 Ansible Engine 2.9 镜像

要迁移 Ansible Engine 2.9 镜像以用于 Ansible Automation Platform 2.2，**ansible-builder** 工具会自动重建镜像（包括其自定义插件和依赖项）的过程，供自动化执行环境使用。有关使用 Ansible Builder 构建执行环境的更多信息，请参阅 创建和恢复执行环境。

2.1.8.1.3. 迁移到 Ansible Core 2.13

升级到 Ansible Core 2.13 时，您需要更新您的 playbook、插件或其他 Ansible 基础架构的一部分，以便由最新版本的 Ansible Core 支持。有关更新 Ansible Core 2.13 兼容性的步骤，请参阅 Ansible 内核 2.13 端口指南。

2.1.8.2. 使用自动化网格扩展自动化

Red Hat Ansible Automation Platform 的自动化网格组件简化了在多站点部署之间分布自动化的过程。对于具有多个隔离的 IT 环境的企业，自动化网格提供了一个一致且可靠的方法，使用对等对网格通信网络在执行节点上部署和扩展自动化。

当从版本 1.x 升级到最新版本的 Ansible Automation Platform 时，您需要将旧隔离节点中的数据迁移到自动化网格所需的执行节点。您可以通过规划混合和控制节点网络来实施自动化中心，然后编辑 Ansible Automation Platform 安装程序中找到的清单文件，为每个执行节点分配与网格相关的值。

有关如何从隔离节点迁移到执行节点的步骤，请参阅**升级和迁移指南**。

有关自动化网格以及为您的环境设计自动化网格的各种方法的信息，请参阅 Red Hat Ansible Automation Platform Automation mesh。

2.2. 使用外部管理的数据库安装 RED HAT ANSIBLE AUTOMATION PLATFORM

您可以使用以下说明安装带有外部管理数据库的 Red Hat Ansible Automation Platform（自动化控制器和自动化中心）。

2.2.1. 先决条件

- 您已经从 Red Hat Ansible Automation Platform 产品软件 中选择并获取了平台安装程序。
- 您需要在满足基本系统要求的机器上安装。
- 您已将所有软件包更新至 RHEL 节点的当前版本。

警告

如果您在 Ansible Automation Platform 安装前没有完全升级 RHEL 节点，您可能会遇到错误。

- 您已创建了 Red Hat Registry Service Account，按照创建 Registry 服务账户指南 中的说明进行操作。
- 您必须在所有节点上配置 NTP 客户端。如需更多信息，请参阅使用 Chrony 配置 NTP 服务器 。

2.2.2. 编辑 Red Hat Ansible Automation Platform 安装程序清单文件

您可以使用 Red Hat Ansible Automation Platform 安装程序清单文件指定您的安装场景。

流程

1. 进入安装程序
   a. [Bundled installer]

   ```
   $ cd ansible-automation-platform-setup-bundle-<latest-version>
   ```

   b. [Online installer]

   ```
   $ cd ansible-automation-platform-setup-<latest-version>
   ```

2. 使用文本编辑器打开**清单**文件。
3. 编辑**清单**文件参数，以指定您的安装场景。请按照以下示例操作。

2.2.3. 带有外部管理数据库的 Red Hat Ansible Automation Platform 清单文件示例

本例描述了如何填充清单文件以安装 Red Hat Ansible Automation Platform。此安装清单文件包含带有外部管理数据库的自动化控制器和自动化中心。

重要

- 您不能在同一节点上安装自动化控制器和自动化 hub。
- 为 **[automationhub]** 主机提供一个可访问的 IP 地址，以确保用户可以从不同节点的 Private Automation Hub 同步内容。
- 在 **registry_username** 和 **registry_password** 中输入 Red Hat Registry Service account 凭证以链接到 Red Hat 容器 registry。

```
[automationcontroller]
controller.acme.org

[automationhub]
automationhub.acme.org

[database]
database-01.acme.org

[all:vars]
admin_password='<password>'
pg_host='database-01.acme.org'
pg_port='5432'
pg_database='awx'
pg_username='awx'
pg_password='<password>'
pg_sslmode='prefer'  # set to 'verify-full' for client-side enforced SSL

registry_url='registry.redhat.io'
registry_username='<registry username>'
registry_password='<registry password>'

# Automation Hub Configuration
#
automationhub_admin_password='<password>'
automationhub_pg_host='database-01.acme.org'
automationhub_pg_port='5432'
automationhub_pg_database='automationhub'
automationhub_pg_username='automationhub'
automationhub_pg_password='<password>'
automationhub_pg_sslmode='prefer'

# The default install will deploy a TLS enabled Automation Hub.
# If for some reason this is not the behavior wanted one can
# disable TLS enabled deployment.
#
# automationhub_disable_https = False
# The default install will generate self-signed certificates for the Automation
# Hub service. If you are providing valid certificate via automationhub_ssl_cert
# and automationhub_ssl_key, one should toggle that value to True.
```

```
#
# automationhub_ssl_validate_certs = False
# SSL-related variables
# If set, this will install a custom CA certificate to the system trust store.
# custom_ca_cert=/path/to/ca.crt
# Certificate and key to install in nginx for the web UI and API
# web_server_ssl_cert=/path/to/tower.cert
# web_server_ssl_key=/path/to/tower.key
# Certificate and key to install in Automation Hub node
# automationhub_ssl_cert=/path/to/automationhub.cert
# automationhub_ssl_key=/path/to/automationhub.key
# Server-side SSL settings for PostgreSQL (when we are installing it).
# postgres_use_ssl=False
# postgres_ssl_cert=/path/to/pgsql.crt
# postgres_ssl_key=/path/to/pgsql.key
```

2.2.4. 设置脚本标志和额外变量

在运行设置脚本以安装自动化控制器时，您还可以传递标志和额外变量：

表 2.5. 标记

参数	描述
-h	显示帮助信息并退出
-i INVENTORY_FILE	到 Ansible 清单文件的路径（默认：**inventory**）
-e EXTRA_VARS	将额外的 Ansible 变量设置为 key=value 或 YAML/JSON
-b	在安装时执行数据库备份
-r	在安装时执行数据库恢复
-k	生成 **并分发** SECRET_KEY。 当使用此参数执行 setup.sh 脚本时，默认会为自动化控制器生成并分发一个新的 secret 密钥，但不用于自动化服务目录。 要为自动化控制器和自动服务目录生成并分发新的 secret 密钥，请指定变量 **rekey_catalog: true**。

使用 **--** 分隔符添加您希望应用的任何 Ansible 参数。例如：**./setup.sh -i my_awesome_inventory.yml -e matburt_is_country_gold=True — -K**。

注意

- 在进行数据库恢复时使用了 **-r**，则会使用默认的恢复目录，除非使用 EXTRA_VARS 提供了一个非默认路径。请参阅以下通过 EXTRA_VAR 指定恢复路径的示例：

```
./setup.sh -e 'restore_backup_file=/path/to/nondefault/location' -r
```

- 您可以通过使用 **-e bundle_install=false** 来强制在线安装：

```
$ ./setup.sh -e bundle_install=false
```

表 2.6. 额外变量

变量	Description	default
upgrade_ansible_with_tower	安装自动化控制器时，请确保 Ansible 也为最新版本	**False**
create_preload_data	安装 Tower 时还创建 Demo Org 、项目 、凭证 、作业模板等。	**True**
bundle_install_folder	在进行捆绑安装时获得捆绑软件仓库的目录	**var/lib/tower-bundle**
nginx_disable_https	禁用使用 nginx 的 HTTPS 数据，这在将 HTTPS 数据卸载到负载均衡器时非常有用	**False**
nginx_disable_hsts	禁用 HSTS web-security policy 机制	**False**
nginx_http_port	nginx 为 HTTP 侦听的端口	**80**
nginx_https_port	nginx 为 HTTPS 侦听的端口	**443**
backup_dir	备份时要使用的临时位置	**/var/backups/tower/**
restore_backup_file	指定要从中恢复的备用文件	None
required_ram	安装 Tower 所需的最小 RAM（应只在测试安装时修改）	**3750**
min_open_fds	最小打开文件描述（应只在测试安装时修改）	None
ignore_preflight_errors	忽略 preflight 检查，在安装到一个模板或其他非系统镜像时很有用（覆盖 **required_ram** 和 **min_open_fds**）	**False**

变量	Description	default
rekey_catalog	当使用此变量设置为 **true** 来执行 **setup.sh** 脚本时，默认会生成一个新的 secret 密钥，并为自动化控制器和自动服务目录发布。	**False**

例子

- 要升级内核：

```
./setup.sh -e upgrade_ansible_with_tower=1
```

- 在 nginx 中禁用 https 处理：

```
./setup.sh -e nginx_disable_https=true
```

- 在从备份文件中恢复时指定非默认路径：

```
./setup.sh -e 'restore_backup_file=/path/to/nondefault/location' -r
```

2.2.5. 运行 Red Hat Ansible Automation Platform 安装程序设置脚本

在完成使用安装私有 Automation Hub 所需的参数更新**清**单文件后，您可以运行设置脚本。

流程

1. 运行 **setup.sh** 脚本

```
$ ./setup.sh
```

安装将开始。

2.2.6. 验证自动化控制器安装

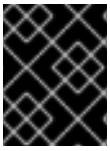
安装完成后，您可以通过使用插入到**清**单文件中的 admin 凭据登录来验证自动化控制器是否已成功安装。

流程

1. 进入**清**单文件中为自动化控制器节点指定的 IP 地址。
2. 使用您在**清**单文件中设置的 Admin 凭据登录。

注意

自动化控制器服务器可从端口 80（https://<TOWER_SERVER_NAME>/）访问，但会被重定向到端口 443，因此端口 443 需要可以被使用。



重要

如果安装失败，且您购买了 Red Hat Ansible Automation Platform 的有效许可证，请通过红帽客户门户网站 https://access.redhat.com/ 联络 Ansible。

当可以成功登录到自动化控制器后，则代表您的 Red Hat Ansible Automation Platform 2.2 安装已完成。

2.2.6.1. 其他自动化控制器配置和资源

请参阅以下资源来探索其他自动化控制器配置。

表 2.7. 配置自动化控制器的资源

Link	描述
自动化控制器快速设置指南	设置自动化控制器并运行第一个 playbook
自动化控制器管理指南	通过客户脚本、管理作业等配置自动化控制器管理。
配置 Red Hat Ansible Automation Platform 的代理支持	使用代理服务器设置自动化控制器
从自动化控制器管理可用性分析和数据收集	管理您与红帽共享的自动化控制器信息
自动化控制器用户指南	更详细地查看自动化控制器功能

2.2.7. 验证自动化中心安装

安装完成后，您可以通过使用插入到**清单**文件中的 admin 凭据登录来验证自动化中心是否已成功安装。

流程

1. 进入**清单**文件中为自动化中心节点指定的 IP 地址。
2. 使用您在**清单**文件中设置的 Admin 凭据登录。

重要

如果安装失败，且您购买了 Red Hat Ansible Automation Platform 的有效许可证，请通过红帽客户门户网站 https://access.redhat.com/ 联络 Ansible。

当可以成功登录到自动化 hub 后，则代表您的 Red Hat Ansible Automation Platform 2.2 安装已完成。

2.2.7.1. 其他自动化中心配置和资源

请参阅以下资源来探索额外的自动化中心配置。

表 2.8. 配置自动化控制器的资源

Link	描述
在私有自动化 hub 中管理用户访问权限	为自动化 hub 配置用户访问
在自动 Hub 中管理红帽认证的集合和 Ansible Galaxy 集合	在自动化 hub 中添加内容
在自动化 hub 中发布专有内容集合	在自动化 hub 中发布内部开发的集合

2.2.8. Ansible Automation Platform 2.2 的下一步是什么

无论您是希望开始使用自动化的新 Ansible Automation Platform 用户，还是希望将旧的 Ansible 内容迁移到最新版本的 Red Hat Ansible Automation Platform，充分利用 Ansible Automation Platform 2.2 的新功能：

2.2.8.1. 将数据迁移到 Ansible Automation Platform 2.2

对于希望完成升级到 Ansible Automation Platform 2.2 的平台管理员，可能需要额外步骤将数据迁移到新实例：

2.2.8.1.1. 从旧的虚拟环境 (venvs) 迁移到自动化执行环境

通过 Ansible Automation Platform 2.2，您可以从自定义 Python 虚拟环境 (venvs) 转换为使用自动化执行环境 - 它是容器化镜像，包括了执行和扩展 Ansible 自动化所需的组件。这包括 Ansible Core、Ansible 内容集合、Python 依赖项、Red Hat Enterprise Linux UBI 8 以及任何其他软件包依赖项。

如果您希望将 venvs 迁移到执行环境，则需要(1)使用 **awx-manage** 命令列出并从原始实例中导出 venvs 列表，然后（2）使用 **ansible-builder** 创建执行环境。如需更多信息，请参阅 升级到自动化执行环境指南 和 创建和恢复执行环境。

2.2.8.1.2. 使用 Ansible Builder 迁移至 Ansible Engine 2.9 镜像

要迁移 Ansible Engine 2.9 镜像以用于 Ansible Automation Platform 2.2，**ansible-builder** 工具会自动重建镜像（包括其自定义插件和依赖项）的过程，供自动化执行环境使用。有关使用 Ansible Builder 构建执行环境的更多信息，请参阅 创建和恢复执行环境。

2.2.8.1.3. 迁移到 Ansible Core 2.13

升级到 Ansible Core 2.13 时，您需要更新您的 playbook、插件或其他 Ansible 基础架构的一部分，以便由最新版本的 Ansible Core 支持。有关更新 Ansible Core 2.13 兼容性的步骤，请参阅 Ansible 内核 2.13 端口指南。

2.2.8.2. 使用自动化网格扩展自动化

Red Hat Ansible Automation Platform 的自动化网格组件简化了在多站点部署之间分布自动化的过程。对于具有多个隔离的 IT 环境的企业，自动化网格提供了一个一致且可靠的方法，使用对等对网格通信网络在执行节点上部署和扩展自动化。

当从版本 1.x 升级到最新版本的 Ansible Automation Platform 时，您需要将旧隔离节点中的数据迁移到自动化网格所需的执行节点。您可以通过规划混合和控制节点网络来实施自动化中心，然后编辑 Ansible Automation Platform 安装程序中找到的清单文件，为每个执行节点分配与网格相关的值。

有关如何从隔离节点迁移到执行节点的步骤，请参阅**升级和迁移指南**。

有关自动化网格以及为您的环境设计自动化网格的各种方法的信息，请参阅 Red Hat Ansible Automation Platform Automation mesh。

第 3 章 在一台机器上安装 RED HAT ANSIBLE AUTOMATION PLATFORM 组件

您可以在以下受支持的场景之一的单个机器上安装 Red Hat Ansible Automation Platform 组件。

3.1. 使用同一节点上的数据库安装自动化控制器

您可以使用以下说明在同一个节点上安装带有数据库的自动化控制器独立实例，或者安装一个非安装程序管理的数据库。此场景包括在单一计算机上安装自动化控制器，包括 Web 前端、REST API 后端和数据库。它安装 PostgreSQL，并将自动化控制器配置为将其用作其数据库。这被视为标准自动化控制器安装场景。

3.1.1. 先决条件

- 您已经从 Red Hat Ansible Automation Platform 产品软件 中选择并获取了平台安装程序。
- 您需要在满足基本系统要求的机器上安装。
- 您已将所有软件包更新至 RHEL 节点的当前版本。

警告

如果您在 Ansible Automation Platform 安装前没有完全升级 RHEL 节点，您可能会遇到错误。

- 您已创建了 Red Hat Registry Service Account，按照创建 Registry 服务账户指南 中的说明进行操作。
- 您必须在所有节点上配置 NTP 客户端。如需更多信息，请参阅使用 Chrony 配置 NTP 服务器 。

3.1.2. 编辑 Red Hat Ansible Automation Platform 安装程序清单文件

您可以使用 Red Hat Ansible Automation Platform 安装程序清单文件指定您的安装场景。

流程

1. 进入安装程序
   a. [Bundled installer]

   ```
   $ cd ansible-automation-platform-setup-bundle-<latest-version>
   ```

   b. [Online installer]

   ```
   $ cd ansible-automation-platform-setup-<latest-version>
   ```

2. 使用文本编辑器打开清单文件。

3. 编辑**清单**文件参数，以指定您的安装场景。请按照以下示例操作。

3.1.3. Red Hat Ansible Automation Platform 单节点清单文件示例

本例描述了如何为自动化控制器的单一节点安装填充清单文件。

重要

- 对于 **forpg_password**，不要使用特殊字符。可能会导致设置失败。
- 在 **registry_username** 和 **registry_password** 中输入 Red Hat Registry Service account 凭证以链接到 Red Hat 容器 registry。

```
[automationcontroller]
controller.example.com (1)

[database]

[all:vars]
admin_password='<password>'

pg_host=''
pg_port=''

pg_database='awx'
pg_username='awx'
pg_password='<password>'

registry_url='registry.redhat.io'
registry_username='<registry username>'
registry_password='<registry password>'
```

(1) 这应该设置为 FQDN/IP。

3.1.4. 设置脚本标志和额外变量

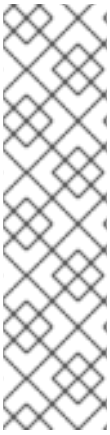
在运行设置脚本以安装自动化控制器时，您还可以传递标志和额外变量：

表 3.1. 标记

参数	描述
-h	显示帮助信息并退出
-i INVENTORY_FILE	到 Ansible 清单文件的路径（默认：**inventory**）
-e EXTRA_VARS	将额外的 Ansible 变量设置为 key=value 或 YAML/JSON
-b	在安装时执行数据库备份

参数	描述
-r	在安装时执行数据库恢复
-k	生成 **并分发** SECRET_KEY。 当使用此参数执行 setup.sh 脚本时，默认会为自动化控制器生成并分发一个新的 secret 密钥，但不用于自动化服务目录。 要为自动化控制器和自动服务目录生成并分发新的 secret 密钥，请指定变量 **rekey_catalog: true**。

使用 **--** 分隔符添加您希望应用的任何 Ansible 参数。例如：**./setup.sh -i my_awesome_inventory.yml -e matburt_is_country_gold=True — -K**。



注意

- 在进行数据库恢复时使用了 **-r**，则会使用默认的恢复目录，除非使用 EXTRA_VARS 提供了一个非默认路径。请参阅以下通过 EXTRA_VAR 指定恢复路径的示例：

```
./setup.sh -e 'restore_backup_file=/path/to/nondefault/location' -r
```

- 您可以通过使用 **-e bundle_install=false** 来强制在线安装：

```
$ ./setup.sh -e bundle_install=false
```

表 3.2. 额外变量

变量	Description	default
upgrade_ansible_with_tower	安装自动化控制器时，请确保 Ansible 也为最新版本	**False**
create_preload_data	安装 Tower 时还创建 Demo Org 、项目 、凭证 、作业模板等。	**True**
bundle_install_folder	在进行捆绑安装时获得捆绑软件仓库的目录	**var/lib/tower-bundle**
nginx_disable_https	禁用使用 nginx 的 HTTPS 数据，这在将 HTTPS 数据卸载到负载均衡器时非常有用	**False**
nginx_disable_hsts	禁用 HSTS web-security policy 机制	**False**
nginx_http_port	nginx 为 HTTP 侦听的端口	**80**

变量	Description	default
nginx_https_port	nginx 为 HTTPS 侦听的端口	**443**
backup_dir	备份时要使用的临时位置	**/var/backups/tower/**
restore_backup_file	指定要从中恢复的备用文件	None
required_ram	安装 Tower 所需的最小 RAM（应只在测试安装时修改）	**3750**
min_open_fds	最小打开文件描述（应只在测试安装时修改）	None
ignore_preflight_errors	忽略 preflight 检查，在安装到一个模板或其他非系统镜像时很有用（覆盖 **required_ram** 和 **min_open_fds**）	**False**
rekey_catalog	当使用此变量设置为 **true** 来执行 **setup.sh** 脚本时，默认会生成一个新的 secret 密钥，并为自动化控制器和自动服务目录发布。	**False**

例子

- 要升级内核：

```
./setup.sh -e upgrade_ansible_with_tower=1
```

- 在 nginx 中禁用 https 处理：

```
./setup.sh -e nginx_disable_https=true
```

- 在从备份文件中恢复时指定非默认路径：

```
./setup.sh -e 'restore_backup_file=/path/to/nondefault/location' -r
```

3.1.5. 运行 Red Hat Ansible Automation Platform 安装程序设置脚本

在完成使用安装私有 Automation Hub 所需的参数更新**清**单文件后，您可以运行设置脚本。

流程

1. 运行 **setup.sh** 脚本

```
$ ./setup.sh
```

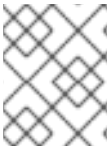
安装将开始。

3.1.6. 验证自动化控制器安装

安装完成后，您可以通过使用插入到**清**单文件中的 admin 凭据登录来验证自动化控制器是否已成功安装。

流程

1. 进入**清**单文件中为自动化控制器节点指定的 IP 地址。
2. 使用您在**清**单文件中设置的 Admin 凭据登录。



注意

自动化控制器服务器可从端口 80（https://<TOWER_SERVER_NAME>/）访问，但会被重定向到端口 443，因此端口 443 需要可以被使用。

重要

如果安装失败，且您购买了 Red Hat Ansible Automation Platform 的有效许可证，请通过红帽客户门户网站 https://access.redhat.com/ 联络 Ansible。

当可以成功登录到自动化控制器后，则代表您的 Red Hat Ansible Automation Platform 2.2 安装已完成。

3.1.6.1. 其他自动化控制器配置和资源

请参阅以下资源来探索其他自动化控制器配置。

表 3.3. 配置自动化控制器的资源

Link	描述
自动化控制器快速设置指南	设置自动化控制器并运行第一个 playbook
自动化控制器管理指南	通过客户脚本、管理作业等配置自动化控制器管理。
配置 Red Hat Ansible Automation Platform 的代理支持	使用代理服务器设置自动化控制器
从自动化控制器管理可用性分析和数据收集	管理您与红帽共享的自动化控制器信息
自动化控制器用户指南	更详细地查看自动化控制器功能

3.1.7. Ansible Automation Platform 2.2 的下一步是什么

无论您是希望开始使用自动化的新 Ansible Automation Platform 用户，还是希望将旧的 Ansible 内容迁移到最新版本的 Red Hat Ansible Automation Platform，充分利用 Ansible Automation Platform 2.2 的新功能：

3.1.7.1. 将数据迁移到 Ansible Automation Platform 2.2

对于希望完成升级到 Ansible Automation Platform 2.2 的平台管理员，可能需要额外步骤将数据迁移到新实例：

3.1.7.1.1. 从旧的虚拟环境 (venvs) 迁移到自动化执行环境

通过 Ansible Automation Platform 2.2，您可以从自定义 Python 虚拟环境 (venvs) 转换为使用自动化执行环境 - 它是容器化镜像，包括了执行和扩展 Ansible 自动化所需的组件。这包括 Ansible Core、Ansible 内容集合、Python 依赖项、Red Hat Enterprise Linux UBI 8 以及任何其他软件包依赖项。

如果您希望将 venvs 迁移到执行环境，则需要(1)使用 **awx-manage** 命令列出并从原始实例中导出 venvs 列表，然后（2）使用 **ansible-builder** 创建执行环境。如需更多信息，请参阅 升级到自动化执行环境指南 和 创建和恢复执行环境。

3.1.7.1.2. 使用 Ansible Builder 迁移至 Ansible Engine 2.9 镜像

要迁移 Ansible Engine 2.9 镜像以用于 Ansible Automation Platform 2.2，**ansible-builder** 工具会自动重建镜像（包括其自定义插件和依赖项）的过程，供自动化执行环境使用。有关使用 Ansible Builder 构建执行环境的更多信息，请参阅 创建和恢复执行环境。

3.1.7.1.3. 迁移到 Ansible Core 2.13

升级到 Ansible Core 2.13 时，您需要更新您的 playbook、插件或其他 Ansible 基础架构的一部分，以便由最新版本的 Ansible Core 支持。有关更新 Ansible Core 2.13 兼容性的步骤，请参阅 Ansible 内核 2.13 端口指南。

3.1.7.2. 使用自动化网格扩展自动化

Red Hat Ansible Automation Platform 的自动化网格组件简化了在多站点部署之间分布自动化的过程。对于具有多个隔离的 IT 环境的企业，自动化网格提供了一个一致且可靠的方法，使用对等对网格通信网络在执行节点上部署和扩展自动化。

当从版本 1.x 升级到最新版本的 Ansible Automation Platform 时，您需要将旧隔离节点中的数据迁移到自动化网格所需的执行节点。您可以通过规划混合和控制节点网络来实施自动化中心，然后编辑 Ansible Automation Platform 安装程序中找到的清单文件，为每个执行节点分配与网格相关的值。

有关如何从隔离节点迁移到执行节点的步骤，请参阅**升级和迁移指南**。

有关自动化网格以及为您的环境设计自动化网格的各种方法的信息，请参阅 Red Hat Ansible Automation Platform Automation mesh。

3.2. 使用外部管理的数据库安装自动化控制器

您可以使用这些说明在配置的单一机器上安装独立自动化控制器服务器，以与远程 PostgreSQL 实例通信作为其数据库。此远程 PostgreSQL 可以是您管理的服务器，也可以由 Amazon RDS 等云服务提供。

3.2.1. 先决条件

- 您已经从 Red Hat Ansible Automation Platform 产品软件 中选择并获取了平台安装程序。
- 您需要在满足基本系统要求的机器上安装。
- 您已将所有软件包更新至 RHEL 节点的当前版本。

警告

如果您在 Ansible Automation Platform 安装前没有完全升级 RHEL 节点，您可能会遇到错误。

- 您已创建了 Red Hat Registry Service Account，按照创建 Registry 服务账户指南 中的说明进行操作。
- 您必须在所有节点上配置 NTP 客户端。如需更多信息，请参阅使用 Chrony 配置 NTP 服务器 。

3.2.2. 编辑 Red Hat Ansible Automation Platform 安装程序清单文件

您可以使用 Red Hat Ansible Automation Platform 安装程序清单文件指定您的安装场景。

流程

1. 进入安装程序
   a. [Bundled installer]

```
$ cd ansible-automation-platform-setup-bundle-<latest-version>
```

   b. [Online installer]

```
$ cd ansible-automation-platform-setup-<latest-version>
```

2. 使用文本编辑器打开**清**单文件。
3. 编辑**清**单文件参数，以指定您的安装场景。请按照以下示例操作。

3.2.3. 带有外部管理数据库的独立自动化控制器的清单文件示例

本例描述了如何填充清单文件，以使用外部数据库部署自动化控制器的安装。

重要

- 对于 **forpg_password**，不要使用特殊字符。可能会导致设置失败。
- 在 **registry_username** 和 **registry_password** 中输入 Red Hat Registry Service account 凭证以链接到 Red Hat 容器 registry。

```
[automationcontroller]
controller.example.com 1


[database]
database.example.com

[all:vars]
```

```
admin_password='<password>'
pg_password='<password>'

pg_host='database.example.com'
pg_port='5432'

pg_database='awx'
pg_username='awx'

registry_url='registry.redhat.io'
registry_username='<registry username>'
registry_password='<registry password>'
```

1 这应该设置为 FQDN/IP。

3.2.4. 设置脚本标志和额外变量

在运行设置脚本以安装自动化控制器时，您还可以传递标志和额外变量：

表 3.4. 标记

参数	描述
-h	显示帮助信息并退出
-i INVENTORY_FILE	到 Ansible 清单文件的路径（默认：**inventory**）
-e EXTRA_VARS	将额外的 Ansible 变量设置为 key=value 或 YAML/JSON
-b	在安装时执行数据库备份
-r	在安装时执行数据库恢复
-k	生成 **并分发** SECRET_KEY。 当使用此参数执行 setup.sh 脚本时，默认会为自动化控制器生成并分发一个新的 secret 密钥，但不用于自动化服务目录。 要为自动化控制器和自动服务目录生成并分发新的 secret 密钥，请指定变量 **rekey_catalog: true**。

使用 **--** 分隔符添加您希望应用的任何 Ansible 参数。例如：**./setup.sh -i my_awesome_inventory.yml -e matburt_is_country_gold=True — -K**。

注意

- 在进行数据库恢复时使用了 **-r**，则会使用默认的恢复目录，除非使用 EXTRA_VARS 提供了一个非默认路径。请参阅以下通过 EXTRA_VAR 指定恢复路径的示例：

```
./setup.sh -e 'restore_backup_file=/path/to/nondefault/location' -r
```

- 您可以通过使用 **-e bundle_install=false** 来强制在线安装：

```
$ ./setup.sh -e bundle_install=false
```

表 3.5. 额外变量

变量	Description	default
upgrade_ansible_with_tower	安装自动化控制器时，请确保 Ansible 也为最新版本	**False**
create_preload_data	安装 Tower 时还创建 Demo Org 、项目 、凭证 、作业模板等。	**True**
bundle_install_folder	在进行捆绑安装时获得捆绑软件仓库的目录	**var/lib/tower-bundle**
nginx_disable_https	禁用使用 nginx 的 HTTPS 数据，这在将 HTTPS 数据卸载到负载均衡器时非常有用	**False**
nginx_disable_hsts	禁用 HSTS web-security policy 机制	**False**
nginx_http_port	nginx 为 HTTP 侦听的端口	**80**
nginx_https_port	nginx 为 HTTPS 侦听的端口	**443**
backup_dir	备份时要使用的临时位置	**/var/backups/tower/**
restore_backup_file	指定要从中恢复的备用文件	None
required_ram	安装 Tower 所需的最小 RAM（应只在测试安装时修改）	**3750**
min_open_fds	最小打开文件描述（应只在测试安装时修改）	None
ignore_preflight_errors	忽略 preflight 检查，在安装到一个模板或其他非系统镜像时很有用（覆盖 **required_ram** 和 **min_open_fds**）	**False**

变量	Description	default
rekey_catalog	当使用此变量设置为 **true** 来执行 **setup.sh** 脚本时，默认会生成一个新的 secret 密钥，并为自动化控制器和自动服务目录发布。	**False**

例子

- 要升级内核：

```
./setup.sh -e upgrade_ansible_with_tower=1
```

- 在 nginx 中禁用 https 处理：

```
./setup.sh -e nginx_disable_https=true
```

- 在从备份文件中恢复时指定非默认路径：

```
./setup.sh -e 'restore_backup_file=/path/to/nondefault/location' -r
```

3.2.5. 运行 Red Hat Ansible Automation Platform 安装程序设置脚本

在完成使用安装私有 Automation Hub 所需的参数更新**清**单文件后，您可以运行设置脚本。

流程

1. 运行 **setup.sh** 脚本

```
$ ./setup.sh
```

安装将开始。

3.2.6. 验证自动化控制器安装

安装完成后，您可以通过使用插入到**清**单文件中的 admin 凭据登录来验证自动化控制器是否已成功安装。

流程

1. 进入**清**单文件中为自动化控制器节点指定的 IP 地址。
2. 使用您在**清**单文件中设置的 Admin 凭据登录。

注意

自动化控制器服务器可从端口 80（https://<TOWER_SERVER_NAME>/）访问，但会被重定向到端口 443，因此端口 443 需要可以被使用。

重要

如果安装失败，且您购买了 Red Hat Ansible Automation Platform 的有效许可证，请通过红帽客户门户网站 https://access.redhat.com/ 联络 Ansible。

当可以成功登录到自动化控制器后，则代表您的 Red Hat Ansible Automation Platform 2.2 安装已完成。

3.2.6.1. 其他自动化控制器配置和资源

请参阅以下资源来探索其他自动化控制器配置。

表 3.6. 配置自动化控制器的资源

Link	描述
自动化控制器快速设置指南	设置自动化控制器并运行第一个 playbook
自动化控制器管理指南	通过客户脚本、管理作业等配置自动化控制器管理。
配置 Red Hat Ansible Automation Platform 的代理支持	使用代理服务器设置自动化控制器
从自动化控制器管理可用性分析和数据收集	管理您与红帽共享的自动化控制器信息
自动化控制器用户指南	更详细地查看自动化控制器功能

3.2.7. Ansible Automation Platform 2.2 的下一步是什么

无论您是希望开始使用自动化的新 Ansible Automation Platform 用户，还是希望将旧的 Ansible 内容迁移到最新版本的 Red Hat Ansible Automation Platform，充分利用 Ansible Automation Platform 2.2 的新功能：

3.2.7.1. 将数据迁移到 Ansible Automation Platform 2.2

对于希望完成升级到 Ansible Automation Platform 2.2 的平台管理员，可能需要额外步骤将数据迁移到新实例：

3.2.7.1.1. 从旧的虚拟环境 (venvs) 迁移到自动化执行环境

通过 Ansible Automation Platform 2.2，您可以从自定义 Python 虚拟环境 (venvs) 转换为使用自动化执行环境 - 它是容器化镜像，包括了执行和扩展 Ansible 自动化所需的组件。这包括 Ansible Core、Ansible 内容集合、Python 依赖项、Red Hat Enterprise Linux UBI 8 以及任何其他软件包依赖项。

如果您希望将 venvs 迁移到执行环境，则需要(1)使用 **awx-manage** 命令列出并从原始实例中导出 venvs 列表，然后（2）使用 **ansible-builder** 创建执行环境。如需更多信息，请参阅 升级到自动化执行环境指南 和 创建和恢复执行环境。

3.2.7.1.2. 使用 Ansible Builder 迁移至 Ansible Engine 2.9 镜像

要迁移 Ansible Engine 2.9 镜像以用于 Ansible Automation Platform 2.2，**ansible-builder** 工具会自动重建镜像（包括其自定义插件和依赖项）的过程，供自动化执行环境使用。有关使用 Ansible Builder 构建执行环境的更多信息，请参阅 创建和恢复执行环境。

3.2.7.1.3. 迁移到 Ansible Core 2.13

升级到 Ansible Core 2.13 时，您需要更新您的 playbook、插件或其他 Ansible 基础架构的一部分，以便由最新版本的 Ansible Core 支持。有关更新 Ansible Core 2.13 兼容性的步骤，请参阅 Ansible 内核 2.13 端口指南。

3.2.7.2. 使用自动化网格扩展自动化

Red Hat Ansible Automation Platform 的自动化网格组件简化了在多站点部署之间分布自动化的过程。对于具有多个隔离的 IT 环境的企业，自动化网格提供了一个一致且可靠的方法，使用对等对网格通信网络在执行节点上部署和扩展自动化。

当从版本 1.x 升级到最新版本的 Ansible Automation Platform 时，您需要将旧隔离节点中的数据迁移到自动化网格所需的执行节点。您可以通过规划混合和控制节点网络来实施自动化中心，然后编辑 Ansible Automation Platform 安装程序中找到的清单文件，为每个执行节点分配与网格相关的值。

有关如何从隔离节点迁移到执行节点的步骤，请参阅**升级和迁移指南**。

有关自动化网格以及为您的环境设计自动化网格的各种方法的信息，请参阅 Red Hat Ansible Automation Platform Automation mesh。

3.3. 使用同一节点上的数据库安装自动化中心（AUTOMATION HUB）

您可以使用以下说明，安装带有在同一个节点上的数据库或非安全程序管理的数据库的自动化中心独立实例。

3.3.1. 先决条件

- 您已经从 Red Hat Ansible Automation Platform 产品软件 中选择并获取了平台安装程序。
- 您需要在满足基本系统要求的机器上安装。
- 您已将所有软件包更新至 RHEL 节点的当前版本。

警告

如果您在 Ansible Automation Platform 安装前没有完全升级 RHEL 节点，您可能会遇到错误。

- 您已创建了 Red Hat Registry Service Account，按照创建 Registry 服务账户指南 中的说明进行操作。
- 您必须在所有节点上配置 NTP 客户端。如需更多信息，请参阅使用 Chrony 配置 NTP 服务器 。

3.3.2. 编辑 Red Hat Ansible Automation Platform 安装程序清单文件

您可以使用 Red Hat Ansible Automation Platform 安装程序清单文件指定您的安装场景。

流程

1. 进入安装程序

   a. [Bundled installer]

   ```
   $ cd ansible-automation-platform-setup-bundle-<latest-version>
   ```

   b. [Online installer]

   ```
   $ cd ansible-automation-platform-setup-<latest-version>
   ```

2. 使用文本编辑器打开**清**单文件。

3. 编辑**清**单文件参数，以指定您的安装场景。请按照以下示例操作。

3.3.3. 独立自动化中心清单文件示例

本例描述了如何填充清单文件来部署自动化中心的独立实例。

重要

- Red Hat Ansible Automation Platform 或 Automation hub：在 **[automationhub]** 组中添加一个 Automation hub 主机。您不能在同一节点上安装自动化控制器和自动化 hub。
- 为 **[automationhub]** 主机提供可访问 IP 地址或完全限定域名(FQDN)，以确保用户可以从不同节点从自动化中心同步并安装内容。不要使用 'localhost'。
- 在 **registry_username** 和 **registry_password** 中输入 Red Hat Registry Service account 凭证以链接到 Red Hat 容器 registry。

```
[automationcontroller]


[automationhub]
<FQDN> ansible_connection=local

[all:vars]
registry_url='registry.redhat.io'
registry_username='<registry username>'
registry_password='<registry password>'

automationhub_admin_password= <PASSWORD>

automationhub_pg_host=''
automationhub_pg_port=''

automationhub_pg_database='automationhub'
automationhub_pg_username='automationhub'
automationhub_pg_password=<PASSWORD>
automationhub_pg_sslmode='prefer'

# The default install will deploy a TLS enabled Automation Hub.
# If for some reason this is not the behavior wanted one can
# disable TLS enabled deployment.
```

```
#
# automationhub_disable_https = False
# The default install will generate self-signed certificates for the Automation
# Hub service. If you are providing valid certificate via automationhub_ssl_cert
# and automationhub_ssl_key, one should toggle that value to True.
#
# automationhub_ssl_validate_certs = False
# SSL-related variables
# If set, this will install a custom CA certificate to the system trust store.
# custom_ca_cert=/path/to/ca.crt
# Certificate and key to install in Automation Hub node
# automationhub_ssl_cert=/path/to/automationhub.cert
# automationhub_ssl_key=/path/to/automationhub.key
```

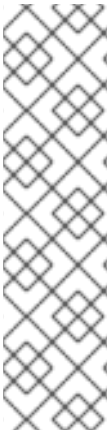
3.3.4. 设置脚本标志和额外变量

在运行设置脚本以安装自动化控制器时，您还可以传递标志和额外变量：

表 3.7. 标记

参数	描述
-h	显示帮助信息并退出
-i INVENTORY_FILE	到 Ansible 清单文件的路径（默认：**inventory**）
-e EXTRA_VARS	将额外的 Ansible 变量设置为 key=value 或 YAML/JSON
-b	在安装时执行数据库备份
-r	在安装时执行数据库恢复
-k	生成 **并分发** SECRET_KEY。 当使用此参数执行 setup.sh 脚本时，默认会为自动化控制器生成并分发一个新的 secret 密钥，但不用于自动化服务目录。 要为自动化控制器和自动服务目录生成并分发新的 secret 密钥，请指定变量 **rekey_catalog: true**。

使用 **--** 分隔符添加您希望应用的任何 Ansible 参数。例如：**./setup.sh -i my_awesome_inventory.yml -e matburt_is_country_gold=True — -K**。



注意

- 在进行数据库恢复时使用了 **-r**，则会使用默认的恢复目录，除非使用 EXTRA_VARS 提供了一个非默认路径。请参阅以下通过 EXTRA_VAR 指定恢复路径的示例：

```
./setup.sh -e 'restore_backup_file=/path/to/nondefault/location' -r
```

- 您可以通过使用 **-e bundle_install=false** 来强制在线安装：

```
$ ./setup.sh -e bundle_install=false
```

表 3.8. 额外变量

变量	Description	default
upgrade_ansible_with_tower	安装自动化控制器时，请确保 Ansible 也为最新版本	**False**
create_preload_data	安装 Tower 时还创建 Demo Org 、项目 、凭证 、作业模板等。	**True**
bundle_install_folder	在进行捆绑安装时获得捆绑软件仓库的目录	**var/lib/tower-bundle**
nginx_disable_https	禁用使用 nginx 的 HTTPS 数据，这在将 HTTPS 数据卸载到负载均衡器时非常有用	**False**
nginx_disable_hsts	禁用 HSTS web-security policy 机制	**False**
nginx_http_port	nginx 为 HTTP 侦听的端口	**80**
nginx_https_port	nginx 为 HTTPS 侦听的端口	**443**
backup_dir	备份时要使用的临时位置	**/var/backups/tower/**
restore_backup_file	指定要从中恢复的备用文件	None
required_ram	安装 Tower 所需的最小 RAM（应只在测试安装时修改）	**3750**
min_open_fds	最小打开文件描述（应只在测试安装时修改）	None
ignore_preflight_errors	忽略 preflight 检查，在安装到一个模板或其他非系统镜像时很有用（覆盖 **required_ram** 和 **min_open_fds**）	**False**

变量	Description	default
rekey_catalog	当使用此变量设置为 **true** 来执行 **setup.sh** 脚本时，默认会生成一个新的 secret 密钥，并为自动化控制器和自动服务目录发布。	**False**

例子

- 要升级内核：

```
./setup.sh -e upgrade_ansible_with_tower=1
```

- 在 nginx 中禁用 https 处理：

```
./setup.sh -e nginx_disable_https=true
```

- 在从备份文件中恢复时指定非默认路径：

```
./setup.sh -e 'restore_backup_file=/path/to/nondefault/location' -r
```

3.3.5. 运行 Red Hat Ansible Automation Platform 安装程序设置脚本

在完成使用安装私有 Automation Hub 所需的参数更新**清**单文件后，您可以运行设置脚本。

流程

1. 运行 **setup.sh** 脚本

```
$ ./setup.sh
```

安装将开始。

3.3.6. 验证自动化中心安装

安装完成后，您可以通过使用插入到**清**单文件中的 admin 凭据登录来验证自动化中心是否已成功安装。

流程

1. 进入**清**单文件中为自动化中心节点指定的 IP 地址。
2. 使用您在**清**单文件中设置的 Admin 凭据登录。

重要

如果安装失败，且您购买了 Red Hat Ansible Automation Platform 的有效许可证，请通过红帽客户门户网站 https://access.redhat.com/ 联络 Ansible。

当可以成功登录到自动化 hub 后，则代表您的 Red Hat Ansible Automation Platform 2.2 安装已完成。

3.3.6.1. 其他自动化中心配置和资源

请参阅以下资源来探索额外的自动化中心配置。

表 3.9. 配置自动化控制器的资源

Link	描述
在私有自动化 hub 中管理用户访问权限	为自动化 hub 配置用户访问
在自动 Hub 中管理红帽认证的集合和 Ansible Galaxy 集合	在自动化 hub 中添加内容
在自动化 hub 中发布专有内容集合	在自动化 hub 中发布内部开发的集合

3.3.7. Ansible Automation Platform 2.2 的下一步是什么

无论您是希望开始使用自动化的新 Ansible Automation Platform 用户，还是希望将旧的 Ansible 内容迁移到最新版本的 Red Hat Ansible Automation Platform，充分利用 Ansible Automation Platform 2.2 的新功能：

3.3.7.1. 将数据迁移到 Ansible Automation Platform 2.2

对于希望完成升级到 Ansible Automation Platform 2.2 的平台管理员，可能需要额外步骤将数据迁移到新实例：

3.3.7.1.1. 从旧的虚拟环境 (venvs) 迁移到自动化执行环境

通过 Ansible Automation Platform 2.2，您可以从自定义 Python 虚拟环境 (venvs) 转换为使用自动化执行环境 - 它是容器化镜像，包括了执行和扩展 Ansible 自动化所需的组件。这包括 Ansible Core、Ansible 内容集合、Python 依赖项、Red Hat Enterprise Linux UBI 8 以及任何其他软件包依赖项。

如果您希望将 venvs 迁移到执行环境，则需要(1)使用 **awx-manage** 命令列出并从原始实例中导出 venvs 列表，然后（2）使用 **ansible-builder** 创建执行环境。如需更多信息，请参阅 升级到自动化执行环境指南 和 创建和恢复执行环境。

3.3.7.1.2. 使用 Ansible Builder 迁移至 Ansible Engine 2.9 镜像

要迁移 Ansible Engine 2.9 镜像以用于 Ansible Automation Platform 2.2，**ansible-builder** 工具会自动重建镜像（包括其自定义插件和依赖项）的过程，供自动化执行环境使用。有关使用 Ansible Builder 构建执行环境的更多信息，请参阅 创建和恢复执行环境。

3.3.7.1.3. 迁移到 Ansible Core 2.13

升级到 Ansible Core 2.13 时，您需要更新您的 playbook、插件或其他 Ansible 基础架构的一部分，以便由最新版本的 Ansible Core 支持。有关更新 Ansible Core 2.13 兼容性的步骤，请参阅 Ansible 内核 2.13 端口指南。

3.3.7.2. 使用自动化网格扩展自动化

Red Hat Ansible Automation Platform 的自动化网格组件简化了在多站点部署之间分布自动化的过程。对于具有多个隔离的 IT 环境的企业，自动化网格提供了一个一致且可靠的方法，使用对等对网格通信网络在执行节点上部署和扩展自动化。

当从版本 1.x 升级到最新版本的 Ansible Automation Platform 时，您需要将旧隔离节点中的数据迁移到自动化网格所需的执行节点。您可以通过规划混合和控制节点网络来实施自动化中心，然后编辑 Ansible Automation Platform 安装程序中找到的清单文件，为每个执行节点分配与网格相关的值。

有关如何从隔离节点迁移到执行节点的步骤，请参阅**升级和迁移指南**。

有关自动化网格以及为您的环境设计自动化网格的各种方法的信息，请参阅 Red Hat Ansible Automation Platform Automation mesh。

3.4. 使用外部数据库安装自动化中心

您可以使用以下说明安装带有外部管理数据库的独立自动化中心实例。这会在一台机器上安装自动化 hub 服务器，并使用 Ansible Automation Platform 安装程序安装远程 PostgreSQL 数据库。

3.4.1. 先决条件

- 您已经从 Red Hat Ansible Automation Platform 产品软件 中选择并获取了平台安装程序。
- 您需要在满足基本系统要求的机器上安装。
- 您已将所有软件包更新至 RHEL 节点的当前版本。

警告

如果您在 Ansible Automation Platform 安装前没有完全升级 RHEL 节点，您可能会遇到错误。

- 您已创建了 Red Hat Registry Service Account，按照创建 Registry 服务账户指南 中的说明进行操作。
- 您必须在所有节点上配置 NTP 客户端。如需更多信息，请参阅使用 Chrony 配置 NTP 服务器 。

3.4.2. 编辑 Red Hat Ansible Automation Platform 安装程序清单文件

您可以使用 Red Hat Ansible Automation Platform 安装程序清单文件指定您的安装场景。

流程

1. 进入安装程序
   a. [Bundled installer]

```
$ cd ansible-automation-platform-setup-bundle-<latest-version>
```

   b. [Online installer]

```
$ cd ansible-automation-platform-setup-<latest-version>
```

2. 使用文本编辑器打开清单文件。

3. 编辑**清**单文件参数，以指定您的安装场景。请按照以下示例操作。

3.4.3. 独立自动化中心清单文件示例

本例描述了如何填充清单文件来部署自动化中心的独立实例。

重要

- Red Hat Ansible Automation Platform 或 Automation hub：在 **[automationhub]** 组中添加一个 Automation hub 主机。您不能在同一节点上安装自动化控制器和自动化 hub。
- 为 **[automationhub]** 主机提供可访问 IP 地址或完全限定域名(FQDN)，以确保用户可以从不同节点从自动化中心同步并安装内容。不要使用 'localhost'。
- 在 **registry_username** 和 **registry_password** 中输入 Red Hat Registry Service account 凭证以链接到 Red Hat 容器 registry。

```
[automationcontroller]


[automationhub]
<FQDN> ansible_connection=local

[database]
host2

[all:vars]
registry_url='registry.redhat.io'
registry_username='<registry username>'
registry_password='<registry password>'

automationhub_admin_password= <PASSWORD>

automationhub_pg_host=''
automationhub_pg_port=''

automationhub_pg_database='automationhub'
automationhub_pg_username='automationhub'
automationhub_pg_password=<PASSWORD>
automationhub_pg_sslmode='prefer'

# The default install will deploy a TLS enabled Automation Hub.
# If for some reason this is not the behavior wanted one can
# disable TLS enabled deployment.
#
# automationhub_disable_https = False
# The default install will generate self-signed certificates for the Automation
# Hub service. If you are providing valid certificate via automationhub_ssl_cert
# and automationhub_ssl_key, one should toggle that value to True.
#
# automationhub_ssl_validate_certs = False
# SSL-related variables
# If set, this will install a custom CA certificate to the system trust store.
# custom_ca_cert=/path/to/ca.crt
```

```
# Certificate and key to install in Automation Hub node
# automationhub_ssl_cert=/path/to/automationhub.cert
# automationhub_ssl_key=/path/to/automationhub.key
```

3.4.4. 私有自动化中心上的 LDAP 配置

您必须在 Red Hat Ansible Automation Platform 安装程序清单文件中设置以下六个变量，以配置用于 LDAP 身份验证的私有自动化中心：

- **automationhub_authentication_backend**
- **automationhub_ldap_server_uri**
- **automationhub_ldap_bind_dn**
- **automationhub_ldap_bind_password**
- **automationhub_ldap_user_search_base_dn**
- **automationhub_ldap_group_search_base_dn**

如果缺少其中的任何变量，Ansible Automation 安装程序将无法完成安装。

3.4.4.1. 设置清单文件变量

当使用 LDAP 身份验证配置私有自动化中心时，您必须在安装过程中在清单文件中设置正确的变量。

先决条件

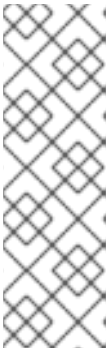
- 确定您的系统使用 Red Hat Ansible Automation Platform 2.2.1 或更高版本。
- 确定您使用私有自动化中心 4.5.2 或更高版本。

流程

1. 根据编辑 Red Hat Ansible Automation Platform 安装程序清单文件 中的步骤访问您的清单文件。
2. 使用以下示例来设置 Ansible Automation Platform 清单文件：

```
automationhub_authentication_backend = "ldap"

automationhub_ldap_server_uri = "ldap://ldap:389"   (for LDAPs use
automationhub_ldap_server_uri = "ldaps://ldap-server-fqdn")
automationhub_ldap_bind_dn = "cn=admin,dc=ansible,dc=com"
automationhub_ldap_bind_password = "GoodNewsEveryone"
automationhub_ldap_user_search_base_dn = "ou=people,dc=ansible,dc=com"
automationhub_ldap_group_search_base_dn = "ou=people,dc=ansible,dc=com"
```



注意

以下变量将使用默认值设置，除非您使用其他选项进行了设置。

```
auth_ldap_user_search_scope= `SUBTREE'
auth_ldap_user_search_filter= `(uid=%(user)s)`
auth_ldap_group_search_scope= 'SUBTREE'
auth_ldap_group_search_filter= '(objectClass=Group)`
auth_ldap_group_type_class= 'django_auth_ldap.config:GroupOfNamesType'
```

3. 如果您计划在私有自动化中心中设置额外的参数（如用户组、超级用户访问、镜像等），请继续下一部分。

3.4.4.2. 配置额外的 LDAP 参数

如果您计划设置超级用户访问权限、用户组、镜像或其他额外参数，您可以创建一个在 **ldap_extra_settings** 字典中组成它们的 YAML 文件。

流程

1. 创建一个包含 **ldap_extra_settings** 的 YAML 文件，如下所示：

```
#ldapextras.yml
---
ldap_extra_settings:
  AUTH_LDAP_USER_ATTR_MAP: '{"first_name": "givenName", "last_name": "sn", "email":
"mail"}'
...
```

然后在私有自动化中心安装过程中运行 **setup.sh -e @ldapextras.yml**。

2. 使用本示例根据 LDAP 组中的成员资格设置超级用户标志。

```
#ldapextras.yml
---
ldap_extra_settings:
  AUTH_LDAP_USER_FLAGS_BY_GROUP: {"is_superuser": "cn=pah-
admins,ou=groups,dc=example,dc=com",}
...
```

然后在私有自动化中心安装过程中运行 **setup.sh -e @ldapextras.yml**。

3. 使用本示例设置超级用户访问权限。

```
#ldapextras.yml
---
ldap_extra_settings:
  AUTH_LDAP_USER_FLAGS_BY_GROUP: {"is_superuser": "cn=pah-
admins,ou=groups,dc=example,dc=com",}
...
```

然后在私有自动化中心安装过程中运行 **setup.sh -e @ldapextras.yml**。

4. 使用本示例来镜像（mirror）您属于的所有 LDAP 组。

```
#ldapextras.yml
---
ldap_extra_settings:
  AUTH_LDAP_MIRROR_GROUPS: True
...
```

然后在私有自动化中心安装过程中运行 **setup.sh -e @ldapextras.yml**。

5. 使用本示例映射 LDAP 用户属性（如用户名、姓氏和电子邮件地址）。

```
#ldapextras.yml
---
ldap_extra_settings:
  AUTH_LDAP_USER_ATTR_MAP: {"first_name": "givenName", "last_name": "sn", "email":
"mail",}
...
```

然后在私有自动化中心安装过程中运行 **setup.sh -e @ldapextras.yml**。

6. 使用以下示例根据 LDAP 组成员资格授予或拒绝访问权限。

   a. 要授予私有自动化中心访问权限（例如，**cn=pah-nosoupforyou,ou=groups,dc=example,dc=com** 组的成员）：

   ```
   #ldapextras.yml
   ---
   ldap_extra_settings:
     AUTH_LDAP_REQUIRE_GROUP: "cn=pah-users,ou=groups,dc=example,dc=com'
   ...
   ```

   b. 拒绝私有自动化中心访问（例如，**cn=pah-nosoupforyou,ou=groups,dc=example,dc=com** 组的成员）：

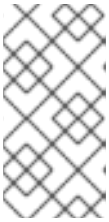
   ```
   #ldapextras.yml
   ---
   ldap_extra_settings:
     AUTH_LDAP_DENY_GROUP: 'cn=pah-nosoupforyou,ou=groups,dc=example,dc=com'
   ...
   ```

   然后在私有自动化中心安装过程中运行 **setup.sh -e @ldapextras.yml**。

7. 使用本示例启用 LDAP 调试日志记录。

```
#ldapextras.yml
---
ldap_extra_settings:
  GALAXY_LDAP_LOGGING: True
...
```

然后在私有自动化中心安装过程中运行 **setup.sh -e @ldapextras.yml**。



注意

如果重新运行 **setup.sh** 或在短时间内启用了调试日志，则可以将包含 **GALAXY_LDAP_LOGGING: True** 的行手动添加到私有自动化中心上的 **/etc/pulp/settings.py** 文件。重启 **pulpcore-api.service** 和 **nginx.service** 以使更改生效。为了避免因为人为错误而失败，请仅在需要时使用此方法。

8. 通过设置变量 **AUTH_LDAP_CACHE_TIMEOUT**，使用本示例配置 LDAP 缓存。

```
#ldapextras.yml
---
ldap_extra_settings:
  AUTH_LDAP_CACHE_TIMEOUT: 3600
...
```

然后在私有自动化中心安装过程中运行 **setup.sh -e @ldapextras.yml**。

您可以查看私有自动化中心上的 **/etc/pulp/settings.py** 文件中的所有设置。

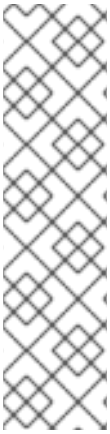
3.4.5. 设置脚本标志和额外变量

在运行设置脚本以安装自动化控制器时，您还可以传递标志和额外变量：

表 3.10. 标记

参数	描述
-h	显示帮助信息并退出
-i INVENTORY_FILE	到 Ansible 清单文件的路径（默认：**inventory**）
-e EXTRA_VARS	将额外的 Ansible 变量设置为 key=value 或 YAML/JSON
-b	在安装时执行数据库备份
-r	在安装时执行数据库恢复
-k	生成 **并分发** SECRET_KEY。 当使用此参数执行 setup.sh 脚本时，默认会为自动化控制器生成并分发一个新的 secret 密钥，但不用于自动化服务目录。 要为自动化控制器和自动服务目录生成并分发新的 secret 密钥，请指定变量 **rekey_catalog: true**。

使用 **--** 分隔符添加您希望应用的任何 Ansible 参数。例如：**./setup.sh -i my_awesome_inventory.yml -e matburt_is_country_gold=True — -K**。



注意

- 在进行数据库恢复时使用了 **-r**，则会使用默认的恢复目录，除非使用 EXTRA_VARS 提供了一个非默认路径。请参阅以下通过 EXTRA_VAR 指定恢复路径的示例：

```
./setup.sh -e 'restore_backup_file=/path/to/nondefault/location' -r
```

- 您可以通过使用 **-e bundle_install=false** 来强制在线安装：

```
$ ./setup.sh -e bundle_install=false
```

表 3.11. 额外变量

变量	Description	default
upgrade_ansible_with_tower	安装自动化控制器时，请确保 Ansible 也为最新版本	**False**
create_preload_data	安装 Tower 时还创建 Demo Org 、项目 、凭证 、作业模板等。	**True**
bundle_install_folder	在进行捆绑安装时获得捆绑软件仓库的目录	**var/lib/tower-bundle**
nginx_disable_https	禁用使用 nginx 的 HTTPS 数据，这在将 HTTPS 数据卸载到负载均衡器时非常有用	**False**
nginx_disable_hsts	禁用 HSTS web-security policy 机制	**False**
nginx_http_port	nginx 为 HTTP 侦听的端口	**80**
nginx_https_port	nginx 为 HTTPS 侦听的端口	**443**
backup_dir	备份时要使用的临时位置	**/var/backups/tower/**
restore_backup_file	指定要从中恢复的备用文件	None
required_ram	安装 Tower 所需的最小 RAM（应只在测试安装时修改）	**3750**
min_open_fds	最小打开文件描述（应只在测试安装时修改）	None
ignore_preflight_errors	忽略 preflight 检查，在安装到一个模板或其他非系统镜像时很有用（覆盖 **required_ram** 和 **min_open_fds**）	**False**

变量	Description	default
rekey_catalog	当使用此变量设置为 **true** 来执行 **setup.sh** 脚本时，默认会生成一个新的 secret 密钥，并为自动化控制器和自动服务目录发布。	**False**

例子

- 要升级内核：

```
./setup.sh -e upgrade_ansible_with_tower=1
```

- 在 nginx 中禁用 https 处理：

```
./setup.sh -e nginx_disable_https=true
```

- 在从备份文件中恢复时指定非默认路径：

```
./setup.sh -e 'restore_backup_file=/path/to/nondefault/location' -r
```

3.4.6. 运行 Red Hat Ansible Automation Platform 安装程序设置脚本

在完成使用安装私有 Automation Hub 所需的参数更新**清**单文件后，您可以运行设置脚本。

流程

1. 运行 **setup.sh** 脚本

```
$ ./setup.sh
```

安装将开始。

3.4.7. 验证自动化控制器安装

安装完成后，您可以通过使用插入到**清**单文件中的 admin 凭据登录来验证自动化控制器是否已成功安装。

流程

1. 进入**清**单文件中为自动化控制器节点指定的 IP 地址。
2. 使用您在**清**单文件中设置的 Admin 凭据登录。

注意

自动化控制器服务器可从端口 80（https://<TOWER_SERVER_NAME>/）访问，但会被重定向到端口 443，因此端口 443 需要可以被使用。

重要

如果安装失败，且您购买了 Red Hat Ansible Automation Platform 的有效许可证，请通过红帽客户门户网站 https://access.redhat.com/ 联络 Ansible。

当可以成功登录到自动化控制器后，则代表您的 Red Hat Ansible Automation Platform 2.2 安装已完成。

3.4.7.1. 其他自动化中心配置和资源

请参阅以下资源来探索额外的自动化中心配置。

表 3.12. 配置自动化控制器的资源

Link	描述
在私有自动化 hub 中管理用户访问权限	为自动化 hub 配置用户访问
在自动 Hub 中管理红帽认证的集合和 Ansible Galaxy 集合	在自动化 hub 中添加内容
在自动化 hub 中发布专有内容集合	在自动化 hub 中发布内部开发的集合

3.4.8. Ansible Automation Platform 2.2 的下一步是什么

无论您是希望开始使用自动化的新 Ansible Automation Platform 用户，还是希望将旧的 Ansible 内容迁移到最新版本的 Red Hat Ansible Automation Platform，充分利用 Ansible Automation Platform 2.2 的新功能：

3.4.8.1. 将数据迁移到 Ansible Automation Platform 2.2

对于希望完成升级到 Ansible Automation Platform 2.2 的平台管理员，可能需要额外步骤将数据迁移到新实例：

3.4.8.1.1. 从旧的虚拟环境 (venvs) 迁移到自动化执行环境

通过 Ansible Automation Platform 2.2，您可以从自定义 Python 虚拟环境 (venvs) 转换为使用自动化执行环境 - 它是容器化镜像，包括了执行和扩展 Ansible 自动化所需的组件。这包括 Ansible Core、Ansible 内容集合、Python 依赖项、Red Hat Enterprise Linux UBI 8 以及任何其他软件包依赖项。

如果您希望将 venvs 迁移到执行环境，则需要(1)使用 **awx-manage** 命令列出并从原始实例中导出 venvs 列表，然后（2）使用 **ansible-builder** 创建执行环境。如需更多信息，请参阅 升级到自动化执行环境指南 和 创建和恢复执行环境。

3.4.8.1.2. 使用 Ansible Builder 迁移至 Ansible Engine 2.9 镜像

要迁移 Ansible Engine 2.9 镜像以用于 Ansible Automation Platform 2.2，**ansible-builder** 工具会自动重建镜像（包括其自定义插件和依赖项）的过程，供自动化执行环境使用。有关使用 Ansible Builder 构建执行环境的更多信息，请参阅 创建和恢复执行环境。

3.4.8.1.3. 迁移到 Ansible Core 2.13

升级到 Ansible Core 2.13 时，您需要更新您的 playbook、插件或其他 Ansible 基础架构的一部分，以使用最新版本的 Ansible Core 支持。有关更新 Ansible Core 2.13 兼容性的步骤，请参阅 Ansible 内核 2.13 端口指南。

3.4.8.2. 使用自动化网格扩展自动化

Red Hat Ansible Automation Platform 的自动化网格组件简化了在多站点部署之间分布自动化的过程。对于具有多个隔离的 IT 环境的企业，自动化网格提供了一个一致且可靠的方法，使用对等对网格通信网络在执行节点上部署和扩展自动化。

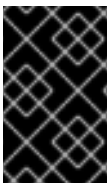
当从版本 1.x 升级到最新版本的 Ansible Automation Platform 时，您需要将旧隔离节点中的数据迁移到自动化网格所需的执行节点。您可以通过规划混合和控制节点网络来实施自动化中心，然后编辑 Ansible Automation Platform 安装程序中找到的清单文件，为每个执行节点分配与网格相关的值。

有关如何从隔离节点迁移到执行节点的步骤，请参阅**升级和迁移指南**。

有关自动化网格以及为您的环境设计自动化网格的各种方法的信息，请参阅 Red Hat Ansible Automation Platform Automation mesh。

第 4 章 多机器集群安装

您可以使用外部管理的数据库使用自动化中心将 Ansible Automation Platform 安装为集群自动化控制器。在这个模式中，安装并激活多个自动化控制器节点。任何节点都可以接收 HTTP 请求，所有节点都可以执行作业。这会在集群中安装 Ansible Automation Platform 服务器，并将它配置为与 PostgreSQL 的远程实例通信作为其数据库。此远程 PostgreSQL 可以是您管理的服务器，也可以由 Amazon RDS 等云服务提供。



重要

Ansible Automation Platform 安装程序允许您为每个清单**仅部署一个**自动化中心。您可以将 Ansible Automation Platform 安装程序用于独立的自动化中心实例，并多次使用任何不同的清单运行安装程序来部署多个自动化中心节点。

4.1. 使用外部管理数据库安装多节点 RED HAT ANSIBLE AUTOMATION PLATFORM

您可以按照以下说明将 Red Hat Ansible Automation Platform 安装为多个自动化控制器节点和带有外部管理数据库的自动化中心。

4.1.1. 先决条件

- 您已经从 Red Hat Ansible Automation Platform 产品软件 中选择并获取了平台安装程序。
- 您需要在满足基本系统要求的机器上安装。
- 您已将所有软件包更新至 RHEL 节点的当前版本。

警告

如果您在 Ansible Automation Platform 安装前没有完全升级 RHEL 节点，您可能会遇到错误。

- 您已创建了 Red Hat Registry Service Account，按照创建 Registry 服务账户指南 中的说明进行操作。
- 您必须在所有节点上配置 NTP 客户端。如需更多信息，请参阅使用 Chrony 配置 NTP 服务器。

4.1.2. 编辑 Red Hat Ansible Automation Platform 安装程序清单文件

您可以使用 Red Hat Ansible Automation Platform 安装程序清单文件指定您的安装场景。

流程

1. 进入安装程序
   a. [Bundled installer]

```
$ cd ansible-automation-platform-setup-bundle-<latest-version>
```

b. [Online installer]

```
$ cd ansible-automation-platform-setup-<latest-version>
```

2. 使用文本编辑器打开**清单**文件。

3. 编辑**清单**文件参数，以指定您的安装场景。请按照以下示例操作。

4.1.3. Red Hat Ansible Automation Platform 多节点清单文件示例

本例描述了如何为自动化控制器的多节点集群安装填充清单文件。

重要

- 您不能在同一节点上安装自动化控制器和自动化 hub。
- 为 **[automationhub]** 主机提供一个可访问的 IP 地址，以确保用户可以从不同节点的 Private Automation Hub 同步内容。
- 对于 **forpg_password**，不要使用特殊字符。可能会导致设置失败。
- 在 **registry_username** 和 **registry_password** 中输入 Red Hat Registry Service account 凭证以链接到 Red Hat 容器 registry。

```
[automationcontroller]
host1
host11
host12


[automationhub]
host2

[database]
1

[all:vars]
ansible_become=true

admin_password='<password>'

pg_host='dbnode.example.com'
pg_port='5432'

pg_database='tower'
pg_username='tower'
pg_password='<password>'

registry_url='registry.redhat.io'
registry_username='<registry username>'
registry_password='<registry password>'
```

1 字段应为空。

4.1.4. 设置脚本标志和额外变量

在运行设置脚本以安装自动化控制器时，您还可以传递标志和额外变量：

表 4.1. 标记

参数	描述
-h	显示帮助信息并退出
-i INVENTORY_FILE	到 Ansible 清单文件的路径（默认：**inventory**）
-e EXTRA_VARS	将额外的 Ansible 变量设置为 key=value 或 YAML/JSON
-b	在安装时执行数据库备份
-r	在安装时执行数据库恢复
-k	生成 **并分发** SECRET_KEY。 当使用此参数执行 setup.sh 脚本时，默认会为自动化控制器生成并分发一个新的 secret 密钥，但不用于自动化服务目录。 要为自动化控制器和自动服务目录生成并分发新的 secret 密钥，请指定变量 **rekey_catalog: true**。

使用 **--** 分隔符添加您希望应用的任何 Ansible 参数。例如：**./setup.sh -i my_awesome_inventory.yml -e matburt_is_country_gold=True — -K**。

注意

- 在进行数据库恢复时使用了 **-r**，则会使用默认的恢复目录，除非使用 EXTRA_VARS 提供了一个非默认路径。请参阅以下通过 EXTRA_VAR 指定恢复路径的示例：

```
./setup.sh -e 'restore_backup_file=/path/to/nondefault/location' -r
```

- 您可以通过使用 **-e bundle_install=false** 来强制在线安装：

```
$ ./setup.sh -e bundle_install=false
```

表 4.2. 额外变量

变量	Description	default
upgrade_ansible_with_tower	安装自动化控制器时，请确保 Ansible 也为最新版本	**False**

变量	Description	default
create_preload_data	安装 Tower 时还创建 Demo Org 、项目 、凭证 、作业模板等。	**True**
bundle_install_folder	在进行捆绑安装时获得捆绑软件仓库的目录	**var/lib/tower-bundle**
nginx_disable_https	禁用使用 nginx 的 HTTPS 数据，这在将 HTTPS 数据卸载到负载均衡器时非常有用	**False**
nginx_disable_hsts	禁用 HSTS web-security policy 机制	**False**
nginx_http_port	nginx 为 HTTP 侦听的端口	**80**
nginx_https_port	nginx 为 HTTPS 侦听的端口	**443**
backup_dir	备份时要使用的临时位置	**/var/backups/tower/**
restore_backup_file	指定要从中恢复的备用文件	None
required_ram	安装 Tower 所需的最小 RAM（应只在测试安装时修改）	**3750**
min_open_fds	最小打开文件描述（应只在测试安装时修改）	None
ignore_preflight_errors	忽略 preflight 检查，在安装到一个模板或其他非系统镜像时很有用（覆盖 **required_ram** 和 **min_open_fds**）	**False**
rekey_catalog	当使用此变量设置为 **true** 来执行 **setup.sh** 脚本时，默认会生成一个新的 secret 密钥，并为自动化控制器和自动服务目录发布。	**False**

例子

- 要升级内核：

```
./setup.sh -e upgrade_ansible_with_tower=1
```

- 在 nginx 中禁用 https 处理：

```
./setup.sh -e nginx_disable_https=true
```

- 在从备份文件中恢复时指定非默认路径：

```
./setup.sh -e 'restore_backup_file=/path/to/nondefault/location' -r
```

4.1.5. 运行 Red Hat Ansible Automation Platform 安装程序设置脚本

在完成使用安装私有 Automation Hub 所需的参数更新**清**单文件后，您可以运行设置脚本。

流程

1. 运行 **setup.sh** 脚本

```
$ ./setup.sh
```

安装将开始。

4.1.6. 验证自动化控制器安装

安装完成后，您可以通过使用插入到**清**单文件中的 admin 凭据登录来验证自动化控制器是否已成功安装。

流程

1. 进入**清**单文件中为自动化控制器节点指定的 IP 地址。
2. 使用您在**清**单文件中设置的 Admin 凭据登录。

注意

自动化控制器服务器可从端口 80（**https://<TOWER_SERVER_NAME>/**）访问，但会被重定向到端口 443，因此端口 443 需要可以被使用。

重要

如果安装失败，且您购买了 Red Hat Ansible Automation Platform 的有效许可证，请通过红帽客户门户网站 https://access.redhat.com/ 联络 Ansible。

当可以成功登录到自动化控制器后，则代表您的 Red Hat Ansible Automation Platform 2.2 安装已完成。

4.1.6.1. 其他自动化控制器配置和资源

请参阅以下资源来探索其他自动化控制器配置。

表 4.3. 配置自动化控制器的资源

Link	描述
自动化控制器快速设置指南	设置自动化控制器并运行第一个 playbook
自动化控制器管理指南	通过客户脚本、管理作业等配置自动化控制器管理。

Link	描述
配置 Red Hat Ansible Automation Platform 的代理支持	使用代理服务器设置自动化控制器
从自动化控制器管理可用性分析和数据收集	管理您与红帽共享的自动化控制器信息
自动化控制器用户指南	更详细地查看自动化控制器功能

4.1.7. 验证自动化中心安装

安装完成后，您可以通过使用插入到**清**单文件中的 admin 凭据登录来验证自动化中心是否已成功安装。

流程

1. 进入**清**单文件中为自动化中心节点指定的 IP 地址。
2. 使用您在**清**单文件中设置的 Admin 凭据登录。

重要

如果安装失败，且您购买了 Red Hat Ansible Automation Platform 的有效许可证，请通过红帽客户门户网站 https://access.redhat.com/ 联络 Ansible。

当可以成功登录到自动化 hub 后，则代表您的 Red Hat Ansible Automation Platform 2.2 安装已完成。

4.1.7.1. 其他自动化中心配置和资源

请参阅以下资源来探索额外的自动化中心配置。

表 4.4. 配置自动化控制器的资源

Link	描述
在私有自动化 hub 中管理用户访问权限	为自动化 hub 配置用户访问
在自动 Hub 中管理红帽认证的集合和 Ansible Galaxy 集合	在自动化 hub 中添加内容
在自动化 hub 中发布专有内容集合	在自动化 hub 中发布内部开发的集合

4.1.8. Ansible Automation Platform 2.2 的下一步是什么

无论您是希望开始使用自动化的新 Ansible Automation Platform 用户，还是希望将旧的 Ansible 内容迁移到最新版本的 Red Hat Ansible Automation Platform，充分利用 Ansible Automation Platform 2.2 的新功能：

4.1.8.1. 将数据迁移到 Ansible Automation Platform 2.2

对于希望完成升级到 Ansible Automation Platform 2.2 的平台管理员，可能需要额外步骤将数据迁移到新实例：

4.1.8.1.1. 从旧的虚拟环境 (venvs) 迁移到自动化执行环境

通过 Ansible Automation Platform 2.2，您可以从自定义 Python 虚拟环境 (venvs) 转换为使用自动化执行环境 - 它是容器化镜像，包括了执行和扩展 Ansible 自动化所需的组件。这包括 Ansible Core、Ansible 内容集合、Python 依赖项、Red Hat Enterprise Linux UBI 8 以及任何其他软件包依赖项。

如果您希望将 venvs 迁移到执行环境，则需要(1)使用 **awx-manage** 命令列出并从原始实例中导出 venvs 列表，然后（2）使用 **ansible-builder** 创建执行环境。如需更多信息，请参阅 升级到自动化执行环境指南 和 创建和恢复执行环境。

4.1.8.1.2. 使用 Ansible Builder 迁移至 Ansible Engine 2.9 镜像

要迁移 Ansible Engine 2.9 镜像以用于 Ansible Automation Platform 2.2，**ansible-builder** 工具会自动重建镜像（包括其自定义插件和依赖项）的过程，供自动化执行环境使用。有关使用 Ansible Builder 构建执行环境的更多信息，请参阅 创建和恢复执行环境。

4.1.8.1.3. 迁移到 Ansible Core 2.13

升级到 Ansible Core 2.13 时，您需要更新您的 playbook、插件或其他 Ansible 基础架构的一部分，以便由最新版本的 Ansible Core 支持。有关更新 Ansible Core 2.13 兼容性的步骤，请参阅 Ansible 内核 2.13 端口指南。

4.1.8.2. 使用自动化网格扩展自动化

Red Hat Ansible Automation Platform 的自动化网格组件简化了在多站点部署之间分布自动化的过程。对于具有多个隔离的 IT 环境的企业，自动化网格提供了一个一致且可靠的方法，使用对等对网格通信网络在执行节点上部署和扩展自动化。

当从版本 1.x 升级到最新版本的 Ansible Automation Platform 时，您需要将旧隔离节点中的数据迁移到自动化网格所需的执行节点。您可以通过规划混合和控制节点网络来实施自动化中心，然后编辑 Ansible Automation Platform 安装程序中找到的清单文件，为每个执行节点分配与网格相关的值。

有关如何从隔离节点迁移到执行节点的步骤，请参阅**升级和迁移指南**。

有关自动化网格以及为您的环境设计自动化网格的各种方法的信息，请参阅 Red Hat Ansible Automation Platform Automation mesh。

第 5 章 配置 RED HAT ANSIBLE AUTOMATION PLATFORM 的代理支持

您可以配置 Red Hat Ansible Automation Platform，以使用代理与流量通信。代理服务器充当来自其他服务器的客户机用于请求查找资源的请求的中介。客户端连接到代理服务器，请求不同的服务器提供某些服务或可用资源，代理服务器会评估请求，以简化和控制其复杂性。以下小节描述了支持的代理配置以及如何设置它们。

5.1. 启用代理支持

为了提供代理服务器支持，自动化控制器通过自动化控制器设置中的 **REMOTE_HOST_HEADERS** 列表变量处理代理请求（如 ALB、NLB、HAProxy、Squid、Nginx 和 tinyproxy）。默认情况下，**REMOTE_HOST_HEADERS** 设置为 **["REMOTE_ADDR", "REMOTE_HOST"]**。

要启用代理服务器支持，请在自动化控制器的设置页面中编辑 **REMOTE_HOST_HEADERS** 字段：

流程

1. 在自动化控制器中，进入到 **Settings → Miscellaneous System**。
2. 在 **REMOTE_HOST_HEADERS** 字段中输入以下值：

```
[
  "HTTP_X_FORWARDED_FOR",
  "REMOTE_ADDR",
  "REMOTE_HOST"
]
```

自动化控制器通过搜索 **REMOTE_HOST_HEADERS** 中的标头列表来确定远程主机的 IP 地址，直到第一个 IP 地址所在的 IP 地址为止。

5.2. 已知的代理

当自动化控制器配置有 **REMOTE_HOST_HEADERS = ['HTTP_X_FORWARDED_FOR', 'REMOTE_ADDR', 'REMOTE_HOST']** 时，它假定 **X-Forwarded-For** 的值源自 Tower 前面的代理/负载均衡器。如果自动化控制器可以在不使用代理/负载均衡器的情况下访问，或者代理没有验证标头，那么 **X-Forwarded-For** 的值可以被断断为原始 IP 地址。在 **REMOTE_HOST_HEADERS** 设置中使用 **HTTP_X_FORWARDED_FOR** 可能会存在安全漏洞。

要避免这种情况，您可以在自动化控制器上的设置菜单中的 PROXY_IP_ALLOWED_LIST 字段中配置允许使用 **PROXY_IP_ALLOWED_LIST** 字段的已知代理列表。不在已知代理列表上的负载均衡器和主机将导致请求被拒绝。

5.2.1. 配置已知的代理

要为自动化控制器配置已知代理列表，请将代理 IP 地址添加到自动化控制器设置页面中的 **PROXY_IP_ALLOWED_LIST** 字段中。

流程

1. 在自动化控制器中，进入到 **Settings → Miscellaneous System**。

2. 在 **PROXY_IP_ALLOWED_LIST** 字段中，输入允许连接到您的自动化控制器的 IP 地址，如下例所示：

PROXY_IP_ALLOWED_LIST 条目示例

```
[
  "example1.proxy.com:8080",
  "example2.proxy.com:8080"
]
```

重要

- **PROXY_IP_ALLOWED_LIST** 需要列表中的代理正确清理标头输入，并正确将 **X-Forwarded-For** 的值设置为客户端的实际源 IP。自动化控制器可以依赖 **PROXY_IP_ALLOWED_LIST** 中的 IP 地址和主机名为 **X-Forwarded-For** 字段提供非欺骗的值。
- 不要将 **HTTP_X_FORWARDED_FOR** 配置为 'REMOTE_HOST_HEADERS' 中的项目，除非以下**所有**条件都满足：
  - 您使用带有 ssl 终止的代理环境；
  - 代理提供 **X-Forwarded-For** 标头的清理或验证处理，以防止客户端欺骗；
  - **/etc/tower/conf.d/remote_host_headers.py** 定义 **PROXY_IP_ALLOWED_LIST**，它只包含可信代理或负载均衡器的原始 IP 地址。

5.3. 配置一个反向代理

您可以通过将 **HTTP_X_FORWARDED_FOR** 添加到自动化控制器设置中的 **REMOTE_HOST_HEADERS** 字段来支持反向代理服务器配置。**X-Forwarded-For** (XFF) HTTP 标头字段标识通过 HTTP 代理或负载平衡器连接到 Web 服务器的客户端的原始 IP 地址。

流程

1. 在自动化控制器中，进入到 **Settings → Miscellaneous System**。
2. 在 **REMOTE_HOST_HEADERS** 字段中输入以下值：

```
[
  "HTTP_X_FORWARDED_FOR",
  "REMOTE_ADDR",
  "REMOTE_HOST"
]
```

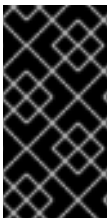
第 6 章 配置自动化控制器 WEBSOCKET 连接

您可以配置自动化控制器，使 websocket 配置与 nginx 或负载均衡器配置保持一致。

6.1. 用于自动化控制器的 WEBSOCKET 配置

自动化控制器节点通过 websocket 连接到所有其他自动化控制器节点。此互连用于将所有 websocket 发送的消息分发到所有其他自动化控制器节点。这是必要的，因为任何浏览器客户端 websocket 都可以订阅可能在任何自动化控制器节点上运行的任何作业。websocket 客户端不路由到特定的自动化控制器节点。任何自动化控制器节点都可以处理任何 websocket 请求，每个自动化控制器节点必须了解所有目标于所有客户端的 websocket 消息。

自动化控制器将通过数据库中的实例记录自动处理其他自动化控制器节点的发现。



重要

- 它旨在使您的节点通过私有、可信任的子网（而不是开放的互联网）对 websocket 流量进行广播。因此，如果您为 websocket 广播关闭 HTTPS，websocket 流量（大部分是 Ansible playbook stdout）会在自动化控制器节点之间进行发送，未加密。

6.1.1. 配置其他自动化控制器节点的自动发现

您可以配置 websocket 连接，使自动化控制器能够通过数据库中的实例记录自动处理其他自动化控制器节点的发现。

- 编辑端口、协议以及是否在建立 websocket 连接时验证证书的自动化控制器 websocket 信息。

```
BROADCAST_WEBSOCKET_PROTOCOL = 'http'
BROADCAST_WEBSOCKET_PORT = 80
BROADCAST_WEBSOCKET_VERIFY_CERT = False
```

第 7 章 从自动化控制器管理可用性分析和数据收集

您可以通过在自动化控制器用户界面中不使用或更改您的设置，改变来自自动化控制器控制器的可用性分析和数据收集。

7.1. 可用性分析和数据收集

自动化控制器包括可用性数据收集，以便更好地了解自动化控制器用户如何与自动化控制器进行交互，以帮助增强未来版本，并持续简化您的用户体验。

只有安装自动化控制器试用或全新安装自动化控制器的用户才会选择使用此数据收集。

其他资源

- 如需更多信息，请参阅红帽隐私政策。

7.1.1. 控制自动化控制器的数据收集

您可以通过在设置菜单中的**用户界面**选项卡中设置参与级别来控制自动化控制器如何收集数据。

流程

1. 登录到您的自动化控制器
2. 进入 **Settings → User Interface**
3. 从 User Tracking State 下拉列表中选择所需的数据收集级别：
   a. **Off**：防止数据收集。
   b. **Anonymous**：启用数据收集功能，而不包括特定于您的用户数据.
   c. **Detailed**：启用数据收集功能，包括特定于您的用户数据。
4. 点 **Save** 应用设置，或点 **Cancel** 来取消更改。

第 8 章 续订和更改 SSL 证书

如果您的当前 SSL 证书已过期或很快过期，您可以续订或替换 Ansible Automation Platform 使用的 SSL 证书。

如果需要使用新主机等新证书重新生成 SSL 证书，则必须续订 SSL 证书。

如果要使用由内部证书颁发机构签名的 SSL 证书，则必须替换 SSL 证书。

8.1. 续订自签名 SSL 证书

以下步骤为自动化控制器和自动化中心重新生成新的 SSL 证书。

流程

1. 将 **aap_service_regen_cert=true** 添加到 **[all:vars]** 部分中的 inventory 文件中：

```
[all:vars]
aap_service_regen_cert=true
```

2. 运行安装程序。

8.2. 更改 SSL 证书

要更改 SSL 证书，您可以编辑清单文件并运行安装程序。安装程序验证所有 Ansible Automation Platform 组件是否正常工作。安装程序可能需要很长时间才能运行。

或者，您可以手动更改 SSL 证书。这速度更快，但没有自动验证。

8.2.1. 先决条件

- 如果存在中间证书颁发机构，您必须将它附加到服务器证书中。
- 自动化控制器和自动化中心都使用 NGINX，因此服务器证书必须采用 PEM 格式。
- 为证书使用正确的顺序：服务器证书首先，后跟中间证书颁发机构。

如需更多信息，请参阅 NGINX 文档中的 ssl 证书部分。

8.2.2. 使用安装程序更改 SSL 证书和密钥

以下流程描述了如何更改清单文件中的 SSL 证书和密钥。

流程

1. 将新的 SSL 证书和密钥复制到相对于 Ansible Automation Platform 安装程序的路径。
2. 将 SSL 证书和密钥的绝对路径添加到清单文件。有关设置这些变量的指导，请参阅 Red Hat Ansible Automation Platform 安装指南中的 Automation controller 变量和 Automation hub 变量部分。
   - 自动化控制器： **web_server_ssl_cert,web_server_ssl_key,custom_ca_cert**
   - 自动化中心：**automationhub_ssl_cert,automationhub_ssl_key,custom_ca_cert**

注意

custom_ca_cert 必须是签署中间证书颁发机构的根证书颁发机构。此文件安装在 **/etc/pki/ca-trust/source/anchors** 中。

3. 运行安装程序。

8.2.3. 手动更改 SSL 证书

8.2.3.1. 在自动化控制器中手动更改 SSL 证书和密钥

以下流程描述了如何在 Automation Controller 上手动更改 SSL 证书和密钥。

流程

1. 备份当前的 SSL 证书：

```
cp /etc/tower/tower.cert /etc/tower/tower.cert-$(date +%F)
```

2. 备份当前密钥文件：

```
cp /etc/tower/tower.key /etc/tower/tower.key-$(date +%F)+
```

3. 将新 SSL 证书复制到 **/etc/tower/tower.cert**。
4. 将新密钥复制到 **/etc/tower/tower.key**。
5. 恢复 SELinux 上下文：

```
restorecon -v /etc/tower/tower.cert /etc/tower/tower.key
```

6. 为证书和密钥文件设置适当的权限：

```
chown root:awx /etc/tower/tower.cert /etc/tower/tower.key
chmod 0600 /etc/tower/tower.cert /etc/tower/tower.key
```

7. 测试 NGINX 配置：

```
nginx -t
```

8. 重新载入 NGINX:

```
systemctl reload nginx.service
```

9. 验证是否安装了新的 SSL 证书和密钥：

```
true | openssl s_client -showcerts -connect ${CONTROLLER_FQDN}:443
```

8.2.3.2. 在 OpenShift Container Platform 上更改自动化控制器的 SSL 证书和密钥

以下流程描述了如何为 OpenShift Container Platform 上运行的自动化控制器更改 SSL 证书和密钥。

流程

1. 将签名的 SSL 证书和密钥复制到安全位置。
2. 在 OpenShift 中创建 TLS secret：

```
oc create secret tls ${CONTROLLER_INSTANCE}-certs-$(date +%F) --cert=/path/to/ssl.crt --key=/path/to/ssl.key
```

3. 修改自动化控制器自定义资源，将 **route_tls_secret** 和新 secret 的名称添加到 spec 部分。

```
oc edit automationcontroller/${CONTROLLER_INSTANCE}
```

```
...
spec:
  route_tls_secret: automation-controller-certs-2023-04-06
...
```

TLS secret 的名称是任意的。在本例中，它的时间戳为创建 secret 的日期，以便将其与应用到自动化控制器实例的其他 TLS secret 进行区分。

1. 等待几分钟，以便应用更改。
2. 验证是否安装了新的 SSL 证书和密钥：

```
true | openssl s_client -showcerts -connect ${CONTROLLER_FQDN}:443
```

8.2.3.3. 在自动化中心中手动更改 SSL 证书和密钥

以下流程描述了如何在自动化中心中手动更改 SSL 证书和密钥。

流程

1. 备份当前的 SSL 证书：

```
cp /etc/pulp/certs/pulp_webserver.crt /etc/pulp/certs/pulp_webserver.crt-$(date +%F)
```

2. 备份当前密钥文件：

```
cp /etc/pulp/certs/pulp_webserver.key /etc/pulp/certs/pulp_webserver.key-$(date +%F)
```

3. 将新 SSL 证书复制到 **/etc/pulp/certs/pulp_webserver.crt**。
4. 将新密钥复制到 **/etc/pulp/certs/pulp_webserver.key**。
5. 恢复 SELinux 上下文：

```
restorecon -v /etc/pulp/certs/pulp_webserver.crt /etc/pulp/certs/pulp_webserver.key
```

6. 为证书和密钥文件设置适当的权限：

```
chown root:pulp /etc/pulp/certs/pulp_webserver.crt /etc/pulp/certs/pulp_webserver.key
```

```
chmod 0600 /etc/pulp/certs/pulp_webserver.crt /etc/pulp/certs/pulp_webserver.key
```

7. 测试 NGINX 配置：

```
nginx -t
```

8. 重新载入 NGINX:

```
systemctl reload nginx.service
```

9. 验证是否安装了新的 SSL 证书和密钥：

```
true | openssl s_client -showcerts -connect ${CONTROLLER_FQDN}:443
```

第 9 章 支持的清单插件模板

升级时，现有配置将迁移到新格式，生成向后兼容的清单输出。使用以下模板来帮助将清单迁移到新样式的清单插件输出。

9.1. AMAZON WEB SERVICES EC2

```
compose:
  ansible_host: public_ip_address
  ec2_account_id: owner_id
  ec2_ami_launch_index: ami_launch_index | string
  ec2_architecture: architecture
  ec2_block_devices: dict(block_device_mappings | map(attribute='device_name') | list |
zip(block_device_mappings | map(attribute='ebs.volume_id') | list))
  ec2_client_token: client_token
  ec2_dns_name: public_dns_name
  ec2_ebs_optimized: ebs_optimized
  ec2_eventsSet: events | default("")
  ec2_group_name: placement.group_name
  ec2_hypervisor: hypervisor
  ec2_id: instance_id
  ec2_image_id: image_id
  ec2_instance_profile: iam_instance_profile | default("")
  ec2_instance_type: instance_type
  ec2_ip_address: public_ip_address
  ec2_kernel: kernel_id | default("")
  ec2_key_name: key_name
  ec2_launch_time: launch_time | regex_replace(" ", "T") | regex_replace("(\+)(\d\d):(\d)(\d)$",
".\g<2>\g<3>Z")
  ec2_monitored: monitoring.state in ['enabled', 'pending']
  ec2_monitoring_state: monitoring.state
  ec2_persistent: persistent | default(false)
  ec2_placement: placement.availability_zone
  ec2_platform: platform | default("")
  ec2_private_dns_name: private_dns_name
  ec2_private_ip_address: private_ip_address
  ec2_public_dns_name: public_dns_name
  ec2_ramdisk: ramdisk_id | default("")
  ec2_reason: state_transition_reason
  ec2_region: placement.region
  ec2_requester_id: requester_id | default("")
  ec2_root_device_name: root_device_name
  ec2_root_device_type: root_device_type
  ec2_security_group_ids: security_groups | map(attribute='group_id') | list |  join(',')
  ec2_security_group_names: security_groups | map(attribute='group_name') | list |  join(',')
  ec2_sourceDestCheck: source_dest_check | default(false) | lower | string
  ec2_spot_instance_request_id: spot_instance_request_id | default("")
  ec2_state: state.name
  ec2_state_code: state.code
  ec2_state_reason: state_reason.message if state_reason is defined else ""
  ec2_subnet_id: subnet_id | default("")
  ec2_tag_Name: tags.Name
  ec2_virtualization_type: virtualization_type
  ec2_vpc_id: vpc_id | default("")
filters:
```

```
  instance-state-name:
  - running
groups:
  ec2: true
hostnames:
  - network-interface.addresses.association.public-ip
  - dns-name
  - private-dns-name
keyed_groups:
  - key: image_id | regex_replace("[^A-Za-z0-9\_]", "_")
    parent_group: images
    prefix: ''
    separator: ''
  - key: placement.availability_zone
    parent_group: zones
    prefix: ''
    separator: ''
  - key: ec2_account_id | regex_replace("[^A-Za-z0-9\_]", "_")
    parent_group: accounts
    prefix: ''
    separator: ''
  - key: ec2_state | regex_replace("[^A-Za-z0-9\_]", "_")
    parent_group: instance_states
    prefix: instance_state
  - key: platform | default("undefined") | regex_replace("[^A-Za-z0-9\_]", "_")
    parent_group: platforms
    prefix: platform
  - key: instance_type | regex_replace("[^A-Za-z0-9\_]", "_")
    parent_group: types
    prefix: type
  - key: key_name | regex_replace("[^A-Za-z0-9\_]", "_")
    parent_group: keys
    prefix: key
  - key: placement.region
    parent_group: regions
    prefix: ''
    separator: ''
  - key: security_groups | map(attribute="group_name") | map("regex_replace", "[^A-Za-z0-9\_]", "_") |
list
    parent_group: security_groups
    prefix: security_group
  - key: dict(tags.keys() | map("regex_replace", "[^A-Za-z0-9\_]", "_") | list | zip(tags.values()
      | map("regex_replace", "[^A-Za-z0-9\_]", "_") | list))
    parent_group: tags
    prefix: tag
  - key: tags.keys() | map("regex_replace", "[^A-Za-z0-9\_]", "_") | list
    parent_group: tags
    prefix: tag
  - key: vpc_id | regex_replace("[^A-Za-z0-9\_]", "_")
    parent_group: vpcs
    prefix: vpc_id
  - key: placement.availability_zone
    parent_group: '{{ placement.region }}'
    prefix: ''
```

```
    separator: ''
plugin: amazon.aws.aws_ec2
use_contrib_script_compatible_sanitization: true
```

9.2. GOOGLE COMPUTE ENGINE

```
auth_kind: serviceaccount
compose:
  ansible_ssh_host: networkInterfaces[0].accessConfigs[0].natIP |
default(networkInterfaces[0].networkIP)
  gce_description: description if description else None
  gce_id: id
  gce_image: image
  gce_machine_type: machineType
  gce_metadata: metadata.get("items", []) | items2dict(key_name="key", value_name="value")
  gce_name: name
  gce_network: networkInterfaces[0].network.name
  gce_private_ip: networkInterfaces[0].networkIP
  gce_public_ip: networkInterfaces[0].accessConfigs[0].natIP | default(None)
  gce_status: status
  gce_subnetwork: networkInterfaces[0].subnetwork.name
  gce_tags: tags.get("items", [])
  gce_zone: zone
hostnames:
- name
- public_ip
- private_ip
keyed_groups:
- key: gce_subnetwork
  prefix: network
- key: gce_private_ip
  prefix: ''
  separator: ''
- key: gce_public_ip
  prefix: ''
  separator: ''
- key: machineType
  prefix: ''
  separator: ''
- key: zone
  prefix: ''
  separator: ''
- key: gce_tags
  prefix: tag
- key: status | lower
  prefix: status
- key: image
  prefix: ''
  separator: ''
plugin: google.cloud.gcp_compute
retrieve_image_info: true
use_contrib_script_compatible_sanitization: true
```

9.3. MICROSOFT AZURE RESOURCE MANAGER

```
conditional_groups:
  azure: true
default_host_filters: []
fail_on_template_errors: false
hostvar_expressions:
  computer_name: name
  private_ip: private_ipv4_addresses[0] if private_ipv4_addresses else None
  provisioning_state: provisioning_state | title
  public_ip: public_ipv4_addresses[0] if public_ipv4_addresses else None
  public_ip_id: public_ip_id if public_ip_id is defined else None
  public_ip_name: public_ip_name if public_ip_name is defined else None
  tags: tags if tags else None
  type: resource_type
keyed_groups:
- key: location
  prefix: ''
  separator: ''
- key: tags.keys() | list if tags else []
  prefix: ''
  separator: ''
- key: security_group
  prefix: ''
  separator: ''
- key: resource_group
  prefix: ''
  separator: ''
- key: os_disk.operating_system_type
  prefix: ''
  separator: ''
- key: dict(tags.keys() | map("regex_replace", "^(.*)$", "\1_") | list | zip(tags.values() | list)) if tags else
[]
  prefix: ''
  separator: ''
plain_host_names: true
plugin: azure.azcollection.azure_rm
use_contrib_script_compatible_sanitization: true
```

9.4. VMWARE VCENTER

```
compose:
  ansible_host: guest.ipAddress
  ansible_ssh_host: guest.ipAddress
  ansible_uuid: 99999999 | random | to_uuid
  availablefield: availableField
  configissue: configIssue
  configstatus: configStatus
  customvalue: customValue
  effectiverole: effectiveRole
  guestheartbeatstatus: guestHeartbeatStatus
  layoutex: layoutEx
  overallstatus: overallStatus
  parentvapp: parentVApp
  recenttask: recentTask
  resourcepool: resourcePool
  rootsnapshot: rootSnapshot
```

```
  triggeredalarmstate: triggeredAlarmState
filters:
- runtime.powerState == "poweredOn"
keyed_groups:
- key: config.guestId
  prefix: ''
  separator: ''
- key: '"templates" if config.template else "guests"'
  prefix: ''
  separator: ''
plugin: community.vmware.vmware_vm_inventory
properties:
- availableField
- configIssue
- configStatus
- customValue
- datastore
- effectiveRole
- guestHeartbeatStatus
- layout
- layoutEx
- name
- network
- overallStatus
- parentVApp
- permission
- recentTask
- resourcePool
- rootSnapshot
- snapshot
- triggeredAlarmState
- value
- capability
- config
- guest
- runtime
- storage
- summary
strict: false
with_nested_properties: true
```

9.5. RED HAT SATELLITE 6

```
group_prefix: foreman_
keyed_groups:
- key: foreman['environment_name'] | lower | regex_replace(' ', '') | regex_replace('[^A-Za-z0-9_]', '_') |
regex_replace('none', '')
  prefix: foreman_environment_
  separator: ''
- key: foreman['location_name'] | lower | regex_replace(' ', '') | regex_replace('[^A-Za-z0-9_]', '_')
  prefix: foreman_location_
  separator: ''
- key: foreman['organization_name'] | lower | regex_replace(' ', '') | regex_replace('[^A-Za-z0-9_]', '_')
  prefix: foreman_organization_
  separator: ''
```

```
- key: foreman['content_facet_attributes']['lifecycle_environment_name'] | lower | regex_replace(' ', '') |
regex_replace('[^A-Za-z0-9_]', '_')
  prefix: foreman_lifecycle_environment_
  separator: ''
- key: foreman['content_facet_attributes']['content_view_name'] | lower | regex_replace(' ', '') |
regex_replace('[^A-Za-z0-9_]', '_')
  prefix: foreman_content_view_
  separator: ''
legacy_hostvars: true
plugin: theforeman.foreman.foreman
validate_certs: false
want_facts: true
want_hostcollections: false
want_params: true
```

9.6. OPENSTACK

```
expand_hostvars: true
fail_on_errors: true
inventory_hostname: uuid
plugin: openstack.cloud.openstack
```

9.7. RED HAT VIRTUALIZATION

```
compose:
  ansible_host: (devices.values() | list)[0][0] if devices else None
keyed_groups:
- key: cluster
  prefix: cluster
  separator: _
- key: status
  prefix: status
  separator: _
- key: tags
  prefix: tag
  separator: _
ovirt_hostname_preference:
- name
- fqdn
ovirt_insecure: false
plugin: ovirt.ovirt.ovirt
```

9.8. 自动化控制器

```
include_metadata: true
inventory_id: <inventory_id or url_quoted_named_url>
plugin: awx.awx.tower
validate_certs: <true or false>
```

第 10 章 自定义通知支持的属性

本节介绍支持的作业属性列表，以及构建通知消息文本的正确语法。支持的作业属性有：

- **allow_simultaneous** - （布尔值）表示多个作业是否可以从与此作业关联的 JT 同时运行
- **controller_node** - （字符串）管理隔离执行环境的实例
- **created** - （日期时间）这个作业创建时的时间戳
- **custom_virtualenv** - （字符串）用于执行作业的自定义虚拟环境
- **description** - （字符串）该作业的可选描述
- **diff_mode** - （布尔值）如果启用，标准输出中会显示对主机上任何模板文件进行的文本更改
- **elapsed** - （十进制）作业运行所经过的时间（以秒为单位）
- **execution_node** - （字符串）执行作业的节点
- **failed** - （布尔值）如果作业失败，则为 true
- **finished** - （日期时间）作业完成执行的日期和时间
- **force_handlers** - （布尔值）当处理程序被强制运行时，即使在该主机上的作业失败，它们也会在收到通知时运行（请注意，在一些情况下，如不可访问的主机，仍然可以阻止处理程序运行）
- **fork** - （整数）作业请求的 fork 数量
- **id** - （整数）此作业的数据库 id
- **job_explanation** - （字符串）状态字段，指示作业在无法运行和捕获 stdout 时的状态
- **job_slice_count** - （整数）如果作为分片作业的一部分运行，分片的总数（如果为 1，则作业不是分片作业的一部分）
- **job_slice_number** - （整数）如果作为分片作业的一部分运行，则为所操作的清单分片的 ID（如果不是分片作业的一部分，不使用这个属性）
- **job_tags** - （字符串）仅执行具有指定标签的任务
- **job_type** - (选择) run、check 或 scan
- **launch_type** - （选择）manual、relaunch、callback、scheduled、dependency、workflow、sync 或 scm
- **limit** - （字符串）如果指定，则 playbook 执行仅限于这组主机
- **modified** - （日期时间）最后一次修改作业的时间戳
- **name** - （字符串）此作业的名称
- **playbook** - （字符串）执行的 playbook
- **scm_revision** - （字符串）用于此作业的项目中的 scm 修订（如果可用）
- **skip_tags** - （字符串）如果指定，playbook 执行将跳过此组标签

- **start_at_task** - （字符串）如果指定，则 playbook 执行从与此名称匹配的任务开始
- **started** - （日期时间）作业加入启动队列的日期和时间
- **status** - （选择）new、pending、waiting、running、successful、failed、error、canceled
- **timeout** - （整数）取消任务前运行的时间（以秒为单位）
- **type** - （选择）此作业的数据类型
- **url** - （字符串）此作业的 URL
- **use_fact_cache** - （布尔值）如果已为作业启用，Tower 将充当 Ansible 事实缓存插件；在 playbook 运行结束后将事实保留到数据库，并缓存事实以供 Ansible 使用
- **verbosity** - （选择）0 到 5（与 Normal 到 WinRM Debug 级别相对应）
- **host_status_counts** （分配给每个状态的唯一主机数量）
  - **skipped** （整数）
  - **ok** （整数）
  - **changed** （整数）
  - **failures** （整数）
  - **dark** （整数）
  - **processed** （整数）
  - **rescued** （整数）
  - **ignored** （整数）
  - **failed** （布尔值）
- **summary_fields:**
  - **清单（inventory）**
    - **id** - （整数）清单的数据库 ID
    - **name** - （字符串）清单的名称
    - **description** - （字符串）清单的可选描述
    - **has_active_failures** - （布尔值）（已弃用）指明此清单中是否有主机失败的标记
    - **total_hosts** - （已弃用）（整数）此清单中的主机总数。
    - **hosts_with_active_failures** - （已弃用）（整数）此清单中有活跃故障的主机数
    - **total_groups** - （已弃用）（整数）此清单中的组总数
    - **groups_with_active_failures** - （已弃用）（整数）此清单中有活跃故障的主机数
    - **has_inventory_sources** - （已弃用）（布尔值）指明此清单是否具有外部清单源的标记

- **total_inventory_sources** - （整数）在此清单中配置的外部清单源总数
- **inventory_sources_with_failures** - （整数）此清单中有故障的外部清单源数量
- **organization_id** - (id) 包含此清单的机构
- **kind** -（选择）（空字符串）（代表主机与清单有直接链接）或 'smart'

- **project**
  - **id** - （整数）项目的数据库 ID
  - **name** - （字符串）项目的名称
  - **description** - （字符串）项目的可选描述
  - **status** - （选择）new、pending、waiting、running、successful、failed、error、canceled、never updated、ok 或 missing 之一
  - **scm_type** - 其中一个（空字符串）、git、hg、svn、insights
- **job_template**
  - **id** - （整数）作业模板的数据库 ID
  - **name** - （字符串）作业模板的名称
  - **description** - （字符串）作业模板的可选描述信息
- **unified_job_template**
  - **id** - （整数）统一的作业模板的数据库 ID
  - **name** - （字符串）统一的作业模板的名称
  - **description** - （字符串）统一的作业模板的可选描述
  - **unified_job_type** - （选择）统一的作业类型（job、workflow_job、project_update 等）
- **instance_group**
  - **id** - （整数）实例组的数据库 ID
  - **name** - （字符串）实例组的名称
- **created_by**
  - **id** -（整数）启动操作的用户的数据库 ID
  - **username** - （字符串）启动了操作的用户名
  - **first_name** - （字符串）名
  - **last_name** - （字符串）姓
- **labels**
  - **count** - （整数）标签数

- **results** - 表示标签的字典列表（例如 {"id": 5, "name": "database jobs"}）

可以在自定义通知消息中使用分组大括号 {{ }} 来引用关于作业的信息。使用点表示法访问特定作业属性，如 {{ job.summary_fields.inventory.name }}。在大括号或周围使用的任何字符，或纯文本，均可添加以进行说明，如 '#' 用于作业 ID，单引号用于表示某些描述符。自定义消息可在整个消息中包含多个变量：

```
{{ job_friendly_name }} {{ job.id }} ran on {{ job.execution_node }} in {{ job.elapsed }} seconds.
```

除了作业属性外，其他一些变量还可添加到模板中：

approval_node_name - （字符串）批准节点名称

approval_status - （选择）批准、被拒绝和 timed_out 之一

url - （字符串）发出通知的作业 URL（这适用于启动、成功、失败和批准通知）

workflow_url - （字符串）指向相关批准节点的 URL。这可让通知接收者进入相关的工作流作业页面来查看具体情况（例如，此节点可在以下位置查看：{{ workflow_url }}）。在与批准相关的通知中，url 和 workflow_url 都相同。

job_friendly_name - （字符串）作业的友好名称

job_metadata - （字符串）作业元数据以 JSON 字符串表示，例如：

```
{'url': 'https://towerhost/$/jobs/playbook/13',
 'traceback': '',
 'status': 'running',
 'started': '2019-08-07T21:46:38.362630+00:00',
 'project': 'Stub project',
 'playbook': 'ping.yml',
 'name': 'Stub Job Template',
 'limit': '',
 'inventory': 'Stub Inventory',
 'id': 42,
 'hosts': {},
 'friendly_name': 'Job',
 'finished': False,
 'credential': 'Stub credential',
 'created_by': 'admin'}
```

附录 A. 清单文件变量

下表包含 Ansible 安装清单文件中使用的预定义变量的信息。

并非所有这些变量都是必需的。

第 11 章 常规变量

变量	描述
enable_insights_collection	如果节点使用 Subscription Manager 注册，则默认安装会将节点注册到 Red Hat Ansible Automation Platform Service 中。设置为 **False** 以禁用。 默认为 **true**
registry_password	用于访问 **registry_url** 的密码凭证。 用于 **[automationcontroller]** 和 **[automationhub]** 组。 在 **registry_username** 和 **registry_password** 中输入 Red Hat Registry Service account 凭证以链接到 Red Hat 容器 registry。 当 **registry_url** 为 **registry.redhat.io** 时，如果没有使用捆绑安装程序，则需要用户名和密码。
registry_url	用于 **[automation controller]** 和 **[automation hub]** 组。 默认为 **registry.redhat.io**。
registry_username	用于访问 **registry_url** 的用户权限。 用于 **[automationcontroller]** 和 **[automationhub]** 组，但只有 **registry_url** 的值为 **registry.redhat.io** 时才使用。 在 **registry_username** 和 **registry_password** 中输入 Red Hat Registry Service account 凭证以链接到 Red Hat 容器 registry。

第 12 章 ANSIBLE AUTOMATION HUB 变量

变量	描述
automationhub_admin_password	必填
automationhub_api_token	如果从 Ansible Automation Platform 2.0 或更早版本升级，您必须： ● 提供一个现有的 Ansible Automation hub 令牌作为 **automationhub_api_token** 或 ● 将 **generate_automationhub_token** 设置为 **true** 来生成新令牌 生成新令牌会导致现有令牌无效。
automationhub_authentication_backend	默认不设置此变量。将它设置为 **ldap** 以使用 LDAP 身份验证。 当它被设置为 **ldap** 时，还必须设置以下变量： ● **automationhub_ldap_server_uri** ● **automationhub_ldap_bind_dn** ● **automationhub_ldap_bind_password** ● **automationhub_ldap_user_search_base_dn** ● **automationhub_ldap_group_search_base_dn**
automationhub_auto_sign_collections	如果启用了集合签名服务，则集合默认不会自动签名。 将此参数设置为 **true** 默认禁用它们。 默认为 **false**。
automationhub_backup_collections	*可选* Ansible Automation hub 在 **/var/lib/pulp** 中提供工件。自动化控制器默认自动备份工件。 您还可以设置 **automationhub_backup_collections = false**，备份/恢复 进程不会备份或恢复 **/var/lib/pulp**。 默认为 **true**

变量	描述
automationhub_collection_signing_service_key	如果启用了集合签名服务，您必须提供此变量，以确保可以正确签名集合。 **/absolute/path/to/key/to/sign**
automationhub_collection_signing_service_script	如果启用了集合签名服务，您必须提供此变量，以确保可以正确签名集合。 **/absolute/path/to/script/that/signs**
automationhub_create_default_collection_signing_service	默认安装不会创建签名服务。如果设置为 **true**，则创建签名服务。 默认为 **false**
automationhub_disable_hsts	默认安装部署一个启用了 TLS 的 Ansible Automation hub。如果 Automation hub 部署了启用了 *HTTP Strict Transport Security* (HSTS) web-security 策略的自动化中心。除非另有指定，否则启用 HSTS web-security 策略机制。如果需要，此设置允许您禁用此设置。 默认为 **false**
automationhub_disable_https	*可选* 如果 Ansible Automation hub 部署启用了 HTTPS。 默认为 **false**。
automationhub_enable_api_access_log	当设置为 **true** 时，在 **/var/log/galaxy_api_access.log** 中创建一个日志文件，该文件会记录提供给平台的所有用户操作，包括其用户名和 IP 地址 默认为 **false**。
automationhub_importer_settings	*可选* 传递给 galaxy-importer 的设置字典。 在导入时，集合可以通过一系列检查。 行为由 **galaxy-importer.cfg** 配置驱动。 例如 **ansible-doc**、**ansible-lint** 和 **flake8**。 这个参数可让您驱动此配置。

要使 Ansible Automation hub 直接连接到 LDAP，必须配置以下变量。可以使用 **ldap_extra_settings** 变量传递的其他 LDAP 相关变量（未由以下 **automationhub_ldap_xxx** 变量涵盖）的列表，可在此处找到： https://django-auth-ldap.readthedocs.io/en/latest/reference.html#settings

变量	描述
automationhub_ldap_bind_dn	使用 **automationhub_ldap_bind_password** 绑定到 LDAP 服务器时使用的名称。
automationhub_ldap_bind_password	*必需* 与 **automationhub_ldap_bind_dn** 搭配使用的密码。
automationhub_ldap_group_search_base_dn	一个 LDAPSearch 对象，用于查找用户可能属于的所有 LDAP 组。如果您的配置对 LDAP 组有任何引用，则必须设置此 和 **automationhub_ldap_group_type**。 默认为 **None**
automatiohub_ldap_group_search_filter	*可选* 用于查找组成员资格的搜索过滤器。 默认为 **(objectClass=Group)**
automationhub_ldap_group_search_scope	*可选* 默认为 **SUBTREE**
automationhub_ldap_group_type_class	*可选* Default =**django_auth_ldap.config:GroupOfNamesType**
automationhub_ldap_server_uri	LDAP 服务器的 URI。这可以是底层 LDAP 库支持的任何 URI。
automationhub_ldap_user_search_base_dn	在目录中查找用户的 LDAPSearch 对象。filter 参数应包含用户名的占位符 %(user)。它必须完全返回一个结果才能成功进行身份验证。
automationhub_main_url	在使用 Single Sign-On 时，指定客户端要连接到的主自动化中心 URL，例如 **https://<hubaddress.example.com>**，它会导致在 **/etc/pulp/settings.py** 中输入外部地址。 如果没有指定，则使用 **[automationhub]** 组中的第一个节点。
automationhub_pg_database	*必需* 数据库名称。 默认为 **automationhub**

变量	描述
automationhub_pg_host	如果不使用内部数据库，则需要此项。
automationhub_pg_password	Automation hub PostgreSQL 数据库的密码。 不要对 **automationhub_pg_password** 使用特殊字符。它们可能会导致密码失败。
automationhub_pg_port	如果不使用内部数据库，则需要此项。 默认 = 5432
automationhub_pg_sslmode	必需。 默认为 **prefer**
automationhub_pg_username	必填 默认为 **automationhub**
automationhub_require_content_approval	*可选* 如果自动化中心在集合可用前强制实施批准机制。 默认情况下，当您将集合上传到自动化中心时，管理员必须批准它，然后供用户使用。 如果要禁用内容批准流，请将变量设置为 **false**。 默认为 **true**
automationhub_ssl_cert	*可选* **/path/to/automationhub.cert** 与 **web_server_ssl_cert** 相同，但用于自动化中心 UI 和 API
automationhub_ssl_key	*可选* **/path/to/automationhub.key** 与 **web_server_ssl_key** 相同，但用于自动化 hub UI 和 API
automationhub_ssl_validate_certs	对于 Red Hat Ansible Automation Platform 2.2 及更新的版本，不再使用这个值。 如果自动化中心因为默认设置，在请求自身时需要验证证书，则 Ansible Automation Platform 会使用自签名证书部署。 默认为 **false**。

变量	描述
generate_automationhub_token	如果从 Red Hat Ansible Automation Platform 2.0 或更早版本进行升级，您必须： ● 提供一个现有的 Ansible Automation hub 令牌作为 **automationhub_api_token** 或 ● 将 **generate_automationhub_token** 设置为 **true** 来生成新令牌。生成新令牌会导致现有令牌无效。
pulp_db_fields_key	您要导入的 Fernet 对加密密钥的相对或绝对路径。该路径位于 Ansible 管理节点上。它用于加密数据库中的某些字段（如凭证）。 如果未指定，将生成一个新密钥。

第 13 章 RED HAT SINGLE SIGN-ON 变量

* 将这些变量用于 **automationhub** 或 **Automationcatalog**。

变量	描述
sso_automation_platform_login_theme	*可选* 用于 Ansible Automation Platform 管理的和外部管理的 Red Hat Single Sign-On。 主题文件所在的目录的路径。如果更改此变量，您必须提供自己的主题文件。 默认 = **ansible-automation-platform**
sso_automation_platform_realm	*可选* 用于 Ansible Automation Platform 管理的和外部管理的 Red Hat Single Sign-On。 SSO 中域的名称。 默认 = **ansible-automation-platform**
sso_automation_platform_realm_displayname	*可选* 用于 Ansible Automation Platform 管理的和外部管理的 Red Hat Single Sign-On。 显示域的名称。 默认 = **Ansible Automation Platform**
sso_console_admin_username	*可选* 用于 Ansible Automation Platform 管理的和外部管理的 Red Hat Single Sign-On。 SSO 管理用户名。 默认 = **admin**
sso_console_admin_password	*必需* 用于 Ansible Automation Platform 管理的和外部管理的 Red Hat Single Sign-On。 SSO 管理密码。

变量	描述
sso_custom_keystore_file	*可选* 仅用于 Ansible Automation Platform 管理的 Red Hat Single Sign-On。 为 SSO 提供客户提供的密钥存储。
sso_host	*必需* 仅用于外部管理的 Red Hat Single Sign-On。 自动化中心和 Automation 服务目录需要 SSO 和 SSO 管理凭证进行身份验证。 还需要 SSO 管理凭据，以设置应用所需的自动化服务目录特定角色。 如果清单中未提供用于配置的 SSO，则必须使用此变量来定义 SSO 主机。
sso_keystore_file_remote	*可选* 仅用于 Ansible Automation Platform 管理的 Red Hat Single Sign-On。 如果客户提供的密钥存储位于远程节点上，则设置为 **true**。 默认为 **false**
sso_keystore_name	*可选* 仅用于 Ansible Automation Platform 管理的 Red Hat Single Sign-On。 SSO 的密钥存储名称。 默认 = **ansible-automation-platform**
sso_keystore_password	启用 HTTPS 的 SSO 的密钥存储密码。 在使用 Ansible Automation Platform 管理的 SSO 以及启用 HTTPS 时，需要此项。默认安装使用 **sso_use_https=true** 部署 SSO。

变量	描述
sso_redirect_host	*可选* 用于 Ansible Automation Platform 管理的和外部管理的 Red Hat Single Sign-On。 如果设置了 **sso_redirect_host**，应用程序会使用它来连接到 SSO 以进行身份验证。 这必须可从客户端机器访问。
sso_ssl_validate_certs	*可选* 用于 Ansible Automation Platform 管理的和外部管理的 Red Hat Single Sign-On。 如果要在连接过程中验证证书，设置为 **true**。 默认为 **true**
sso_use_https	*可选* 用于 Ansible Automation Platform 管理的和外部管理的 Red Hat Single Sign-On。 如果单点登录使用 https。 默认为 **true**

第 14 章 自动化服务目录变量

变量	描述
automationcatalog_controller_password	用于从控制器主机生成令牌。 还需要定义 **automation_controller_main_url**。
automationcatalog_controller_token	用于为自动化控制器创建预创建的 OAuth 令牌。此令牌将被使用，而不是生成令牌。
automationcatalog_controller_username	用于从控制器主机生成令牌。还需要定义 **automation_controller_main_url**。
automationcatalog_controller_verify_ssl	用于启用或禁用自动化服务目录到自动化控制器的 SSL 验证。 默认为 **true**。
automationcatalog_disable_hsts	用于启用或禁用自动服务目录的 HSTS web-security 策略。 默认 = 'false.
automationcatalog_disable_https	用于启用或禁用 Services Catalog 的 HSTS web-security 策略。 默认为 **false**。
automationcatalog_enable_analytics_collection	用于控制自动服务目录的分析集合激活
automationcatalog_main_url	如果需要在 SSO 和自动服务目录主机之间进行使用的替代主机名，则由红帽单点登录主机配置使用。
automationcatalog_pg_database	自动服务目录的 postgres 数据库 URL。
automationcatalog_pg_host	用于自动化服务目录的 PostgreSQL 主机（数据库节点）
automationcatalog_pg_password	自动服务目录的 PostgreSQL 数据库的密码。 对于 **automationcatalog_pg_password**，不要使用特殊字符。它们可能会导致密码失败。
automationcatalog_pg_port	用于自动化服务目录的 PostgreSQL 端口。 默认 = 5432
automationcatalog_pg_username	自动服务目录的 postgres ID。

变量	描述
automationcatalog_ssl_cert	自定义提供的 SSL 证书文件的路径。需要 **Automationcatalog_ssl_key**。使用内部管理的 CA 进行签名，如果未提供证书并且启用了 https，则创建证书。
automationcatalog_ssl_key	自定义提供的 SSL 证书密钥文件的路径。 需要 **automationcatalog_ssl_cert**。 内部管理的 CA 签署，并在为提供且启用了 https 的情况下创建证书。

第 15 章 自动化控制器变量

变量	描述
admin_password	安装完成时访问 UI 的管理用户的密码。
automationcontroller_main_url	对于 SSO 配置自动化服务目录所需的替代前端 URL，请提供 URL。 自动化服务目录需要控制器安装，或者一个指向活跃和可路由的 Controller 服务器的 URL 则必须提供此变量
automationcontroller_password	自动化控制器实例的密码。
automationcontroller_username	自动化控制器实例的用户名。
node_state	*可选* 节点或一组节点的状态。有效选项为 **active**, **deprovision** 从集群中删除节点，或 **iso_migrate** 将旧的隔离节点迁移到执行节点。 默认为 **active**。
node_type	对于 **[automationcontroller]** 组。 可以为这个组分配两个有效的 **node_types**。 **node_type=control** 意味着节点只运行项目和清单更新，但不运行常规作业。 **node_type=hybrid** 能够运行所有内容。 这个组的默认为 **hybrid**。 对于 **[execution_nodes]** 组 可以为这个组分配两个有效的 node_types。 **node_type=hop** 表示节点将作业转发到执行节点。 **node_type=execution** 表示节点可以运行作业。 此组的默认值为 **execution**。

变量	描述
peers	*可选* 对等关系定义节点到节点的连接。 此变量用于在 **receptor.conf** 文件中添加用于与其他节点建立网络连接的 **tcp-peer** 条目。请参阅 Peering peers 变量可以是清单中以逗号分隔的主机和/或组的列表。这被解析为用来构造 **receptor.conf** 文件的一组主机。
pg_database	postgres 数据库的名称。 默认为 **awx**。
pg_host	postgreSQL 主机，可以是外部管理的数据库。
pg_password	postgreSQL 数据库的密码。 对于 **forpg_password**，不要使用特殊字符。它们可能会导致密码失败。 注意 安装时，您将不再需要在清单文件中提供 **apg_hashed_password**，因为 PostgreSQL 13 现在可以更加安全地存储用户的密码。 当您在清单文件中为安装程序提供 **pg_password** 时，PostgreSQL 使用 SCRAM-SHA-256 哈希在安装过程中保护该密码。
pg_port	要使用的 postgreSQL 端口。 默认 = 5432
pg_ssl_mode	**prefer** 或 **verify-full** 之一。 设置为 **verify-full**，用于客户端强制执行 SSL。 默认为 **prefer**。
pg_username	您的 postgres 数据库用户名。 默认为 **awx**。
postgres_ssl_cert	postgres ssl 证书的位置。 **/path/to/pgsql_ssl.cert**

变量	描述
`postgres_ssl_key	postgres ssl 键的位置。 **/path/to/pgsql_ssl.key**
postgres_use_cert	postgres 用户证书的位置。 **/path/to/pgsql.crt**
postgres_use_key	postgres 用户密钥的位置。 **/path/to/pgsql.key**
postgres_use_ssl	postgres 是否使用 SSL。
receptor_listener_port	用于 recptor 连接的端口。 默认 = 27199。
web_server_ssl_cert	*可选* **/path/to/webserver.cert** 与 **automationhub_ssl_cert** 相同，但用于 Web 服务器 UI 和 API。
web_server_ssl_key	*可选* **/path/to/webserver.key** 与 **automationhub_server_ssl_key** 相同，但用于 Web 服务器 UI 和 API。

第 16 章 ANSIBLE 变量

以下变量控制 Ansible Automation Platform 与远程主机交互的方式。

有关特定于特定插件的变量的附加信息，请参考 https://docs.ansible.com/ansible-core/devel/collections/ansible/builtin/index.html

全局配置选项列表请参考 https://docs.ansible.com/ansible-core/devel/reference_appendices/config.html

变量	描述
ansible_connection	用于目标主机上任务的连接插件。 这可以是任何 ansible 连接插件的名称。SSH 协议类型是 **smart**、**ssh** 或 **paramiko**。 默认为 **smart**
ansible_host	要使用的目标主机的 ip 或名称，而不是 **inventory_hostname**。
ansible_port	连接端口号（如果没有提供，默认为 22 用于 ssh）。
ansible_user	连接到主机时使用的用户名。
ansible_password	用于向主机进行身份验证的密码。 从不以纯文本形式存储此变量。 始终使用密码库。
ansible_ssh_private_key_file	ssh 使用的私钥文件。在使用多个密钥且您不想使用 SSH 代理时很有用。
ansible_ssh_common_args	此设置始终附加到 **sftp**、**scp** 和 **ssh** 的默认命令行中。对于为特定主机（或组）配置 ProxyCommand 很有用。
ansible_sftp_extra_args	此设置始终附加到默认的 **sftp** 命令行。
ansible_scp_extra_args	此设置始终附加到默认的 **scp** 命令行。
ansible_ssh_extra_args	此设置始终附加到默认的 **ssh** 命令行。
ansible_ssh_pipelining	确定是否使用 SSH pipelining。这可以覆盖 **ansible.cfg** 中的 pipelining 设置。
ansible_ssh_pass	

变量	描述
ansible_ssh_user	此变量为安装程序设置要使用的 SSH 用户，默认为 root。此用户必须在不需要密码的情况下允许基于 SSH 的身份验证。如果使用基于 SSH 密钥的身份验证，则密钥必须由 SSH 代理管理。
ansible_ssh_executable	（在 2.2 版中添加） 此设置覆盖使用系统 ssh 的默认行为。这可以覆盖 **ansible.cfg** 中的 ssh_executable 设置。
ansible_shell_type	目标系统的 shell 类型。除非将 **ansible_shell_executable** 设置为一个非兼容 shell，否则不应使用此设置。默认情况下，使用 sh 样式的语法对命令进行了格式化。把它设置为 **csh** 或 **fish** 会导致在目标系统上执行命令来遵循这些 shell 的语法。
ansible_shell_executable	这会设置 ansible 控制器在目标机器上使用的 shell，并覆盖 **ansible.cfg** 中的可执行文件，默认为 **/bin/sh**。 只有在无法使用 **/bin/sh** 时（也就是说，如果目标机器上没有安装 **/bin/sh** ）或者无法从 sudo 运行，才应更改。

用户不能直接设置以下变量。Ansible 始终覆盖它们，以反映内部状态。

变量	描述
ansible_check_mode	指明我们是否处于检查模式的布尔值
ansible_dependent_role_names	当前导入到当前 play 中的角色名称作为其他 play 的依赖关系
ansible_limit	用于当前 Ansible 的 **--limit** CLI 选项的内容
ansible_loop	使用 **loop_control.extended** 来启用时包含扩展循环信息的字典或映射
ansible_loop_var	为 loop_control.loop_var 提供的值的名称。在 2.8 中添加
ansible_index_var	为 **loop_control.index_var** 提供的值的名称。在 2.9 中添加

变量	描述
ansible_parent_role_names	在当前角色通过 **include_role** 或 **import_role** 操作来执行时，此变量包含所有父角色的列表，以及最新的角色。换句话说，包含或导入此角色的角色是列表中的第一项。当发生多个包含时，此列表中的第一项是最后一个角色（包含此角色的角色）。一个特定角色可以在此列表中存在超过一次。 例如：当角色 A 包括角色 B （在角色 B 中）时，**ansible_parent_role_names** 将等于 ['A']。如果角色 B 随后包含角色 C，则列表将变为 ['B'，'A']。
ansible_parent_role_paths	如果当前角色通过 **include_role** 或 **import_role** 操作来执行，则此变量包含所有父角色路径的列表，其中最新的角色（换句话说，包含/导入此角色）的角色是列表中的第一个元素。如需此列表中的项目顺序，请参阅 **ansible_parent_role_names**。
ansible_play_batch	当前 play 运行中的活动主机列表串行进行，也称为 **batch（批量处理）**。失败或无法访问的主机不能被视为 **active**。
ansible_play_hosts	在当前的 play 运行中的主机列表，没有串行的限制。失败或无法访问的主机会从这个类别中排除。
ansible_play_hosts_all	play 目标的所有主机列表
ansible_play_role_names	当前导入到当前 play 中的角色名称。此列表不包含通过依赖项隐式包含的角色名称。
ansible_play_name	当前执行 play 的名称。在 2.8. 中添加 (play 的 name 属性，不是 playbook 的文件名)。
ansible_search_path	当前搜索操作插件和查找的路径，换句话说，我们在执行模板时搜索相对路径： src=myfile
ansible_version	包含关于当前运行版本的 ansible 的字典或映射，它有以下键：**full**, **major**, **minor**, **revision** 和 **string**。