Ansible on Clouds 2.x

Red Hat Ansible Automation Platform from AWS Marketplace Guide

Install and configure Red Hat Ansible Automation Platform from AWS Marketplace

Last Updated: 2023-08-29

Ansible on Clouds 2.x Red Hat Ansible Automation Platform from AWS Marketplace Guide

Install and configure Red Hat Ansible Automation Platform from AWS Marketplace

Legal Notice

Abstract

Ansible Automation Platform helps teams manage complex multi-tier deployments by adding control, knowledge, and delegation to Ansible-powered environments. This guide helps you to understand the installation and use of Ansible Automation Platform from AWS Marketplace. This document has been updated to include information for the latest release of Ansible Automation Platform on AWS Marketplace.

Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright's message.

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your feedback on our technical content and encourage you to tell us what you think. If you'd like to add comments, provide insights, correct a typo, or even ask a question, you can do so directly in the documentation.

NOTE

You must have a Red Hat account and be logged in to the customer portal.

To submit documentation feedback from the customer portal, do the following:

1. Select the **Multi-page HTML** format.
2. Click the **Feedback** button at the top-right of the document.
3. Highlight the section of text where you want to provide feedback.
4. Click the **Add Feedback** dialog next to your highlighted text.
5. Enter your feedback in the text box on the right of the page and then click **Submit**.

We automatically create a tracking issue each time you submit feedback. Open the link that is displayed after you click **Submit** and start watching the issue or add more comments.

IMPORTANT

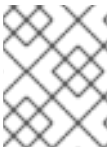
Disclaimer: Links contained in this document to external website(s) are provided for convenience only. Red Hat has not reviewed the links and is not responsible for the content or its availability. The inclusion of any link to an external website does not imply endorsement by Red Hat of the website or their entities, products or services. You agree that Red Hat is not responsible or liable for any loss or expenses that may result due to your use of (or reliance on) the external site or content.

CHAPTER 1. INTRODUCTION

Ansible Automation Platform from AWS Marketplace is an offering that you can deploy from the AWS Marketplace portal. Ansible Automation Platform from AWS Marketplace provides access to a library of Ansible content collections, and it is integrated with key AWS services, so you can start automating the deployment, configuration, and management of infrastructure and applications quickly.

The following Red Hat Ansible Automation Platform components are available on Ansible Automation Platform from AWS Marketplace:

- Automation controller
- Ansible automation hub
- Private automation hub
- Ansible Content Collections
- Automation execution environments
- Ansible content tools, including access to Red Hat Insights for Red Hat Ansible Automation Platform

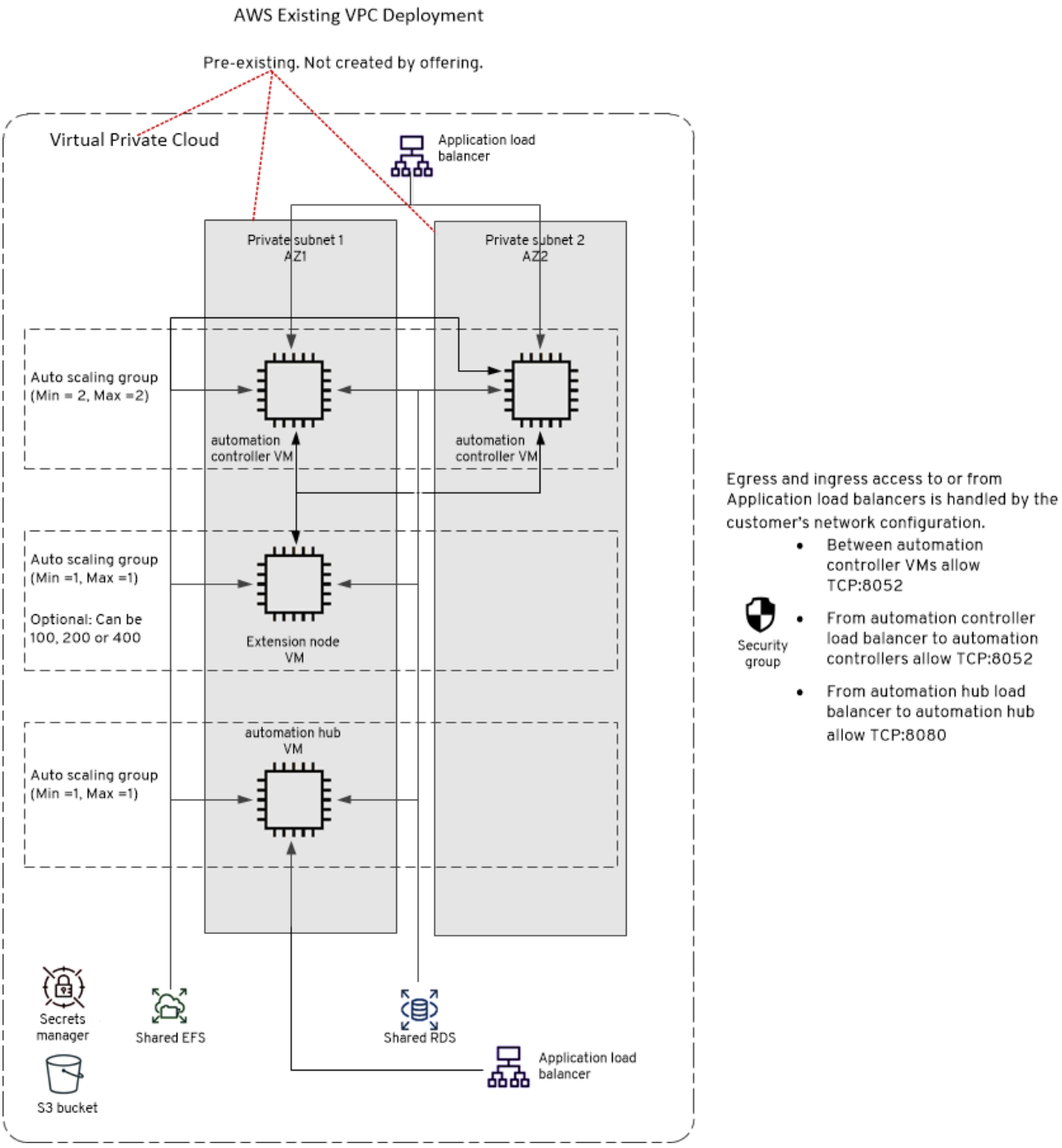


NOTE

Automation mesh is not available on Ansible Automation Platform from AWS Marketplace at this time.

1.1. APPLICATION ARCHITECTURE

Ansible Automation Platform from AWS Marketplace is installed into infrastructure resources running within your AWS account.



Ansible Automation Platform from AWS Marketplace is designed to be private, with no public access allowed by default.

This requires customers to expose the deployed internal *Application Load Balancers* (ALBs) themselves pursuant to their own network requirements and security practices. Some potential ways to expose the ALBs include VPC Peering, Transit Gateway, VPN, External Load Balancers, amongst others.

All cloud infrastructure components are deployed in a Virtual Private Cloud (VPC).

Customers can choose between deploying into an existing VPC, or have the product deploy a new VPC for them. All VM instances and Cloud infrastructure have private IP addresses (allocation determined by the VPC and subnetworks specified at deployment time) by default.

All internal traffic is encrypted using self-signed certificates generated at deployment time. External traffic can also be encrypted by deploying your own certificate on the Application Load Balancers deployed by the product.

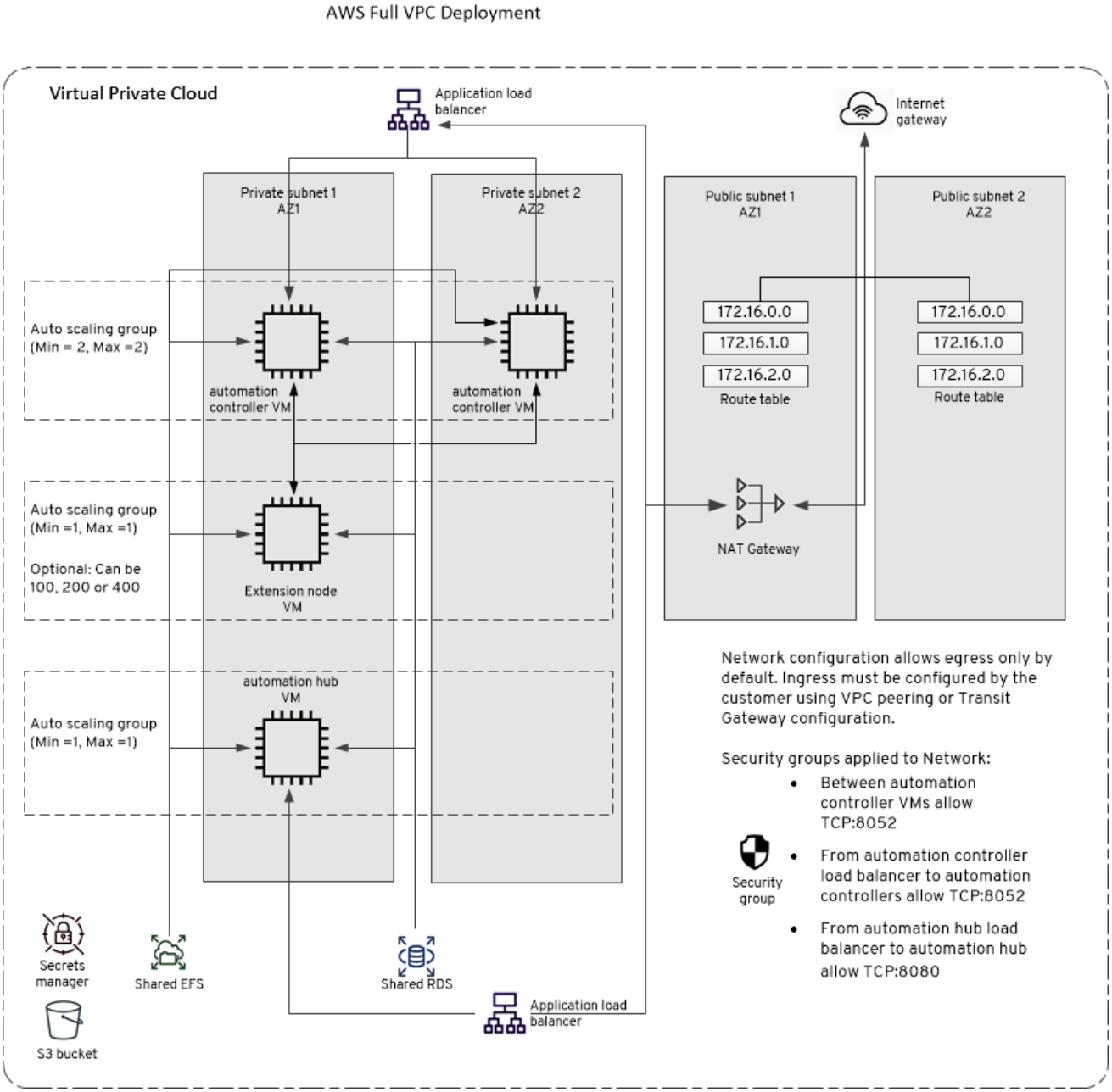
The Ansible Automation Platform software runs as containers on the deployed VM instances.

Autoscaling Groups (ASG) manage VM instances and monitor the health of each service running on the VM instances. ASGs will automatically cycle the VM instances down and replace them with new VM instances if the health check fails to respond ensuring that the Ansible Automation Platform services stay up and available to process requests.

The VM instances run a customized *RedHat Enterprise Linux* (RHEL) *Amazon Machine Image* (AMI) as their base image. This AMI is preloaded with all the required container images and packages to run the Ansible Automation Platform (automation hub, automation controller, and Execution Node components).

A shared EFS (Elastic File Store) volume is mounted into each VM instance provisioned by the product and is used for shared access to common files and resources.

A Relational Database Service (RDS) is provisioned by the product at deployment time and contains databases for both the automation controller and automation hub.



The Foundation product includes two Execution Nodes running on the same VM instances as the

automation controller components (this is called a Hybrid Node configuration in Ansible Automation Platform). Additional Execution Node offerings can be purchased to increase the scale (total number of managed nodes) the Ansible Automation Platform deployment is licensed to automate. When deploying the Execution Node offerings into an existing Ansible Automation Platform Foundation deployment, additional Execution Node VM instances can be deployed and automatically connected to the automation controller of the Foundation deployment, where they immediately begin processing automation tasks.

The Ansible Automation Platform components are run as containers using the Podman container runtime on the VM instances. The Podman runtime configuration is managed as a system service using **systemd** to ensure uptime and availability, and restarting any failed containers automatically.

SELinux is enabled on the VM instances and is supported down to the container level.

Additional operational automations are provided by the offering, available as a separate docker container for download from registry.redhat.io. These additional operational automations include backup, restore, and upgrade.

Any *Common Vulnerabilities and Exposures* (CVEs) found in the RHEL OS base image, the Ansible Automation Platform containers, or any included packages are addressed during upgrade of the Ansible Automation Platform offering by swapping out the base RHEL AMI with a newer version including all required software, packages, and containers.

This is done automatically for you through the use of the included upgrade automation.

Customers can take advantage of these operational automations to simplify the operational readiness of Ansible Automation Platform within their own corporate standards freeing themselves up to focus on developing Ansible Automation to manage their own infrastructure and applications rather than spending time developing automations to manage Ansible Automation Platform.

1.2. SERVICE DESCRIPTIONS

Service Name	Description
Elastic Compute Cloud (EC2)	AWS VM compute platform
Relational Database Service (RDS)	AWS database service
Systems Manager	AWS operations and application management service.
Cloud Watch	AWS logging service
Virtual Private Cloud (VPC)	AWS networking service
NAT Gateway	A NAT gateway is a *Network Address Translation* (NAT) service
Elastic Block Storage (EBS)	AWS block storage service
Elastic File Storage (EFS)	AWS file storage service with support for NFS

CHAPTER 2. INSTALLATION

2.1. PREREQUISITES

Before you can install and register Ansible Automation Platform, you must be familiar with AWS including how services operate, how data is stored, and any privacy implications that may exist by using these services. You must also set up an account with Amazon Web Services.

You must also have an SSH key pair, or *Amazon Elastic Compute Cloud* (EC2) pair to setup Ansible Automation Platform from AWS Marketplace. For more information, read Creating an EC2 pair.

You must have a working knowledge of the following aspects of Amazon Web Services:

- Deploying solutions from the AWS Marketplace
- *Elastic Compute Cloud* (EC2) instances
- *Elastic Block Store* (EBS) volumes
- *Elastic File Storage* (EFS)
- AWS *Virtual Private Clouds* (VPCs)
  - Subnets
  - Route Tables
  - Security Groups
  - Load Balancers
- Network Design
- Hub-and-spoke networking designs
- VPC Peering
- *Class Inter-Domain Routing* (CIDR) blocks
- Transit routing
- AWS CloudWatch
- SSH
- RDS
- AWS SecretsManager

For more information about Amazon Web Services and terminology, see the AWS product documentation.

2.1.1. Policies and permissions

Your AWS account must have the following *Identity and Access Management* (IAM) permissions to create and manage Ansible Automation Platform deployments as well as the resources described in Application architecture.

Your AWS account must also be licensed to deploy Ansible Automation Platform from AWS Marketplace.

The application can fail to deploy if your IAM policies restrict deployment and management of these resources.

The application has two deployment options:

1. Deployment with new VPC
2. Deployment with existing VPC

The following table contains a list of necessary IAM policies:

For deployment with a new VPC	For deployment with an existing VPC

• Managed Policies
  ◦ AWSMarketplaceFullAccess
	• Managed Policies
	  ◦ AWSMarketplaceFullAccess

• CloudFormation inline IAM policies
  ◦ cloudformation:DeleteStack
  ◦ cloudformation:CreateUploadBucket
  ◦ cloudformation:CreateStack
  ◦ cloudformation:UpdateStack
  ◦ cloudformation:GetTemplateSummary
  ◦ cloudformation:ListStacks
  ◦ cloudformation:GetStackPolicy
  ◦ cloudformation:DescribeStacks
  ◦ cloudformation:ListStackResources
  ◦ cloudformation:DescribeStackEvents
	• CloudFormation inline IAM policies
	  ◦ cloudformation:DeleteStack
	  ◦ cloudformation:CreateUploadBucket
	  ◦ cloudformation:CreateStack
	  ◦ cloudformation:UpdateStack
	  ◦ cloudformation:GetTemplateSummary
	  ◦ cloudformation:ListStacks
	  ◦ cloudformation:GetStackPolicy
	  ◦ cloudformation:DescribeStacks
	  ◦ cloudformation:ListStackResources
	  ◦ cloudformation:DescribeStackEvents

For deployment with a new VPC	For deployment with an existing VPC

• S3 inline IAM policies
  ◦ s3:CreateBucket
  ◦ s3:PutObject
  ◦ s3:GetObject
	• S3 inline IAM policies
	  ◦ s3:CreateBucket
	  ◦ s3:PutObject
	  ◦ s3:GetObject

• IAM inline IAM policies
  ◦ iam:DetachRolePolicy
  ◦ iam:RemoveRoleFromInstanceProfile
  ◦ iam:DeleteInstanceProfile
  ◦ iam:DeleteRolePolicy
  ◦ iam:CreateRole
  ◦ iam:PutRolePolicy
  ◦ iam:DeleteRole
  ◦ iam:AttachRolePolicy
  ◦ iam:CreateInstanceProfile
  ◦ iam:AddRoleToInstanceProfile
  ◦ iam:PassRole
  ◦ iam:ListRoles
  ◦ iam:GetRolePolicy
  ◦ iam:TagRole
	• IAM inline IAM policies
	  ◦ iam:DetachRolePolicy
	  ◦ iam:RemoveRoleFromInstanceProfile
	  ◦ iam:DeleteInstanceProfile
	  ◦ iam:DeleteRolePolicy
	  ◦ iam:CreateRole
	  ◦ iam:PutRolePolicy
	  ◦ iam:DeleteRole
	  ◦ iam:AttachRolePolicy
	  ◦ iam:CreateInstanceProfile
	  ◦ iam:AddRoleToInstanceProfile
	  ◦ iam:PassRole
	  ◦ iam:ListRoles
	  ◦ iam:GetRolePolicy
	  ◦ iam:TagRole

• SecretsManager inline IAM policies
  ◦ secretsmanager:DeleteSecret
  ◦ secretsmanager:GetSecretValue
  ◦ secretsmanager:GetRandomPassword
  ◦ secretsmanager:CreateSecret
  ◦ secretsmanager:TagResource
  ◦ secretsmanager:PutSecretValue
	• SecretsManager inline IAM policies
	  ◦ secretsmanager:DeleteSecret
	  ◦ secretsmanager:GetSecretValue
	  ◦ secretsmanager:GetRandomPassword
	  ◦ secretsmanager:CreateSecret
	  ◦ secretsmanager:TagResource
	  ◦ secretsmanager:PutSecretValue

For deployment with a new VPC	For deployment with an existing VPC

• RDS inline IAM policies
  ◦ rds:DeleteDBSubnetGroup
  ◦ rds:DeleteDBInstance
  ◦ rds:CreateDBSubnetGroup
  ◦ rds:AddTagsToResource
  ◦ rds:CreateDBInstance
  ◦ rds:DescribeDBSubnetGroups
  ◦ rds:DescribeDBInstances
	• RDS inline IAM policies
	  ◦ rds:DeleteDBSubnetGroup
	  ◦ rds:DeleteDBInstance
	  ◦ rds:CreateDBSubnetGroup
	  ◦ rds:AddTagsToResource
	  ◦ rds:CreateDBInstance
	  ◦ rds:DescribeDBSubnetGroups
	  ◦ rds:DescribeDBInstances

• Elastic File System inline IAM policies
  ◦ elasticfilesystem:DeleteFileSystem
  ◦ elasticfilesystem:DeleteMountTarget
  ◦ elasticfilesystem:DeleteAccessPoint
  ◦ elasticfilesystem:CreateFileSystem
  ◦ elasticfilesystem:CreateAccessPoint
  ◦ elasticfilesystem:CreateMountTarget
  ◦ elasticfilesystem:DescribeFileSystems
  ◦ elasticfilesystem:DescribeFileSystemPolicy
  ◦ elasticfilesystem:DescribeBackupPolicy
  ◦ elasticfilesystem:DescribeLifecycleConfiguration
  ◦ elasticfilesystem:DescribeAccessPoints
  ◦ elasticfilesystem:DescribeMountTargets
	• Elastic File System inline IAM policies
	  ◦ elasticfilesystem:DeleteFileSystem
	  ◦ elasticfilesystem:DeleteMountTarget
	  ◦ elasticfilesystem:DeleteAccessPoint
	  ◦ elasticfilesystem:CreateFileSystem
	  ◦ elasticfilesystem:CreateAccessPoint
	  ◦ elasticfilesystem:CreateMountTarget
	  ◦ elasticfilesystem:DescribeFileSystems
	  ◦ elasticfilesystem:DescribeFileSystemPolicy
	  ◦ elasticfilesystem:DescribeBackupPolicy
	  ◦ elasticfilesystem:DescribeLifecycleConfiguration
	  ◦ elasticfilesystem:DescribeAccessPoints
	  ◦ elasticfilesystem:DescribeMountTargets

• EC2 inline IAM policies	• EC2 inline IAM policies

For deployment with a new VPC	For deployment with an existing VPC
ec2:RevokeSecurityGroupEgre	ec2:RevokeSecurityGroupEgress
ec2:RevokeSecurityGroupIngress	ec2:RevokeSecurityGroupIngress
ec2:DescribeKeyPairs	ec2:DescribeKeyPairs
ec2:CreateSecurityGroup	ec2:CreateSecurityGroup
ec2:DescribeSecurityGroups	ec2:DescribeSecurityGroups
ec2:DeleteSecurityGroup	ec2:DeleteSecurityGroup
ec2:CreateTags	ec2:CreateTags
ec2:AuthorizeSecurityGroupEgress	ec2:AuthorizeSecurityGroupEgress
ec2:AuthorizeSecurityGroupIngress	ec2:AuthorizeSecurityGroupIngress
ec2:DescribeInstances	ec2:DescribeInstances
ec2:CreateVpc	
ec2:DescribeVpcs	
ec2:DeleteVpc	
ec2:CreateSubnet	
ec2:DeleteSubnet	
ec2:DescribeSubnets	
ec2:DeleteSubnetCidrReservation	
ec2:AssociateSubnetCidrBlock	
ec2:DisassociateSubnetCidrBlock	
ec2:CreateSubnetCidrReservation	
ec2:GetSubnetCidrReservations	
ec2:DescribeAvailabilityZones	
ec2:CreateRouteTable	
ec2:DeleteRouteTable	
ec2:CreateRoute	
ec2:DeleteRoute	
ec2:CreateInternetGateway	
ec2:DeleteInternetGateway	
ec2:DescribeInternetGateways	

For deployment with a new VPC	For deployment with an existing VPC
ec2:AttachInternetGateway ec2:DetachInternetGateway ec2:AssociateRouteTable ec2:DescribeRouteTables ec2:DisassociateRouteTable ec2:ModifyVpcAttribute ec2:DescribeAccountAttributes ec2:DescribeAddresses ec2:AssociateAddress ec2:DisassociateAddress ec2:DescribeAddressesAttribute ec2:ModifyAddressAttribute ec2:AssociateNatGatewayAddress ec2:DisassociateNatGatewayAddress ec2:CreateNatGateway ec2:DeleteNatGateway ec2:DescribeNatGateways ec2:AllocateAddress ec2:ReleaseAddress	

For deployment with a new VPC	For deployment with an existing VPC

• AutoScaling inline IAM policies
  ◦ autoscaling:CreateLaunchConfiguration
  ◦ autoscaling:CreateAutoScalingGroup
  ◦ autoscaling:DeleteLaunchConfiguration
  ◦ autoscaling:UpdateAutoScalingGroup
  ◦ autoscaling:DeleteAutoScalingGroup
  ◦ autoscaling:DescribeAutoScalingGroups
  ◦ autoscaling:DescribeLaunchConfigurations
  ◦ autoscaling:DescribeScalingActivities
  ◦ autoscaling:DescribeAutoScalingInstances
	• AutoScaling inline IAM policies
	  ◦ autoscaling:CreateLaunchConfiguration
	  ◦ autoscaling:CreateAutoScalingGroup
	  ◦ autoscaling:DeleteLaunchConfiguration
	  ◦ autoscaling:UpdateAutoScalingGroup
	  ◦ autoscaling:DeleteAutoScalingGroup
	  ◦ autoscaling:DescribeAutoScalingGroups
	  ◦ autoscaling:DescribeLaunchConfigurations
	  ◦ autoscaling:DescribeScalingActivities
	  ◦ autoscaling:DescribeAutoScalingInstances

For deployment with a new VPC	For deployment with an existing VPC

• ElasticLoadBalancing inline IAM policies
  ◦ elasticloadbalancing:CreateTargetGroup
  ◦ elasticloadbalancing:ModifyTargetGroupAttributes
  ◦ elasticloadbalancing:DeleteTargetGroup
  ◦ elasticloadbalancing:AddTags
  ◦ elasticloadbalancing:CreateLoadBalancer
  ◦ elasticloadbalancing:ModifyLoadBalancerAttributes
  ◦ elasticloadbalancing:DescribeTargetGroups
  ◦ elasticloadbalancing:DescribeListeners
  ◦ elasticloadbalancing:CreateListener
  ◦ elasticloadbalancing:DeleteListener
  ◦ elasticloadbalancingv2:DeleteLoadBalancer
  ◦ elasticloadbalancingv2:DescribeLoadBalancers
	• ElasticLoadBalancing inline IAM policies
	  ◦ elasticloadbalancing:CreateTargetGroup
	  ◦ elasticloadbalancing:ModifyTargetGroupAttributes
	  ◦ elasticloadbalancing:DeleteTargetGroup
	  ◦ elasticloadbalancing:AddTags
	  ◦ elasticloadbalancing:CreateLoadBalancer
	  ◦ elasticloadbalancing:ModifyLoadBalancerAttributes
	  ◦ elasticloadbalancing:DescribeTargetGroups
	  ◦ elasticloadbalancing:DescribeListeners
	  ◦ elasticloadbalancing:CreateListener
	  ◦ elasticloadbalancing:DeleteListener
	  ◦ elasticloadbalancingv2:DeleteLoadBalancer
	  ◦ elasticloadbalancingv2:DescribeLoadBalancers

• SNS inline IAM policies
  ◦ sns:ListTopics
	• SNS inline IAM policies
	  ◦ sns:ListTopics

2.1.2. Creating an EC2 pair

An SSH key pair is required to set up Ansible Automation Platform from AWS Marketplace. You can use an existing key pair or create a new one. If you have an existing key pair, then you can skip this step.

Procedure

1. In the AWS Console, navigate to **EC2**.
2. In the **Network and Security** section, click **Key Pairs**.
3. Click **Create key pair**.
4. Fill out the fields in the input form.

- Use ED25519 as the key pair type.
- Use PEM as the key file format.

5. Click **Create key pair**.
6. The private key will download automatically to your /**downloads** folder. Apply appropriate local file permissions to secure the key file.

2.2. AWS MARKETPLACE

Public Offer

Ansible Automation Platform from AWS Marketplace can be obtained directly from the AWS Marketplace. Follow these steps to purchase and deploy the public offer.

Procedure

1. In the AWS Console, navigate to **AWS Marketplace Subscriptions**.
2. In the navigation bar, click **Discover Products**.
3. Click the listing for **Red Hat Ansible Automation Platform 2 - Up to 100 Managed Nodes**
4. Click **Continue to Subscribe**.
5. Click **Continue to Configuration**.
6. Select the appropriate fulfillment options.
   - Note that some selectors can have only one option.
7. In the **Fulfillment option** field, ensure **Ansible Platform CloudFormations Topology** is selected.
8. In the **Software version** field, select the latest available version from the list.
9. In the **Region** field, select the region with your EC2 key pair. The CloudFormation stack deploys in the same region.
10. Click **Continue to Launch**.
11. In the **Choose Action** field, select **Launch CloudFormation**.
12. Click **Launch**.
    - This opens the CloudFormation create stack page, which is already preconfigured.
13. Next, follow the instructions in Application deployment for detailed guidance about how to configure the deployment using this form.

Private Offer

If you have worked with Red Hat Sales to enable a private offer of Ansible Automation Platform from AWS Marketplace, follow these steps to accept your offer and deploy the solution.

Procedure

1. Visit your Private Offer with the URL link provided by your Red Hat Sales representative.
2. Click **Accept Terms** to subscribe to the AMI Private Offer named Ansible Automation Platform from AWS Marketplace.
3. After accepting the offer terms, click **Continue to Configuration**.
4. Select the appropriate fulfillment options.
    - Note that some selectors can have only one option.
5. In the **Fulfillment option** field, ensure **Ansible Platform CloudFormations Topology** is selected.
6. In the **Software version** field, select the latest available version from the list.
7. In the **Region** field, select the region with your EC2 key pair. The CloudFormation stack deploys in the same region.
8. Click **Continue to Launch**.
9. In the **Choose Action** field, select **Launch CloudFormation**.
10. Click **Launch**.
    - This opens the CloudFormation create stack page, which is already preconfigured.
11. Next, follow the instructions in Application deployment for detailed guidance about how to configure the deployment using this form.

NOTE

If you are accepting terms for the Ansible Automation Platform foundation offer, then accept the discount rate for three virtual machines. All Ansible Automation Platform foundation deployments use three VMs.

If you are accepting terms for Ansible Automation Platform extension nodes, then assign the number of virtual machines arranged with your Red Hat Sales representative. The number of VMs will directly correlate to the number of licensed managed active nodes that you have purchased.

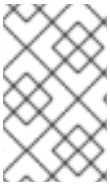
After subscribing, the offer is also listed in your AWS Marketplace Subscriptions.

2.3. APPLICATION DEPLOYMENT

Now that you have subscribed to an offer, you can begin configuration before launching the CloudFormation Stack.

There are two ways to deploy the application.

2.3.1. Deploying an application with a new VPC



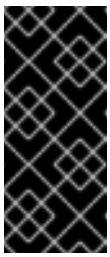
NOTE

The following procedure assumes that you have a marketplace offer. The instructions are a continuation from AWS Marketplace. Complete the procedure in that section before continuing with this section.

This procedure creates a new VPC network and deploys the application in the created VPC.

Procedure

1. For Step 1, on the **Create stack** page, click **Next**.
2. For Step 2, on the **Specify stack details** page
   a. In the **Stack name** field, enter a unique stack name.
   b. In the **EC2 KeyPair** field, select your previously created EC2 keypair.
   c. In the **Select VPC** field, select **New**.
   d. In the **New Network Configuration** section, fill the following fields
      - In the **VPC CIDR Range**, enter the CIDR range to use for the new VPC. Example: 192.168.0.0/16
      - In the **Public Subnet 1 CIDR Range**, enter the CIDR range to use for the first public subnet, for example, 192.168.0.0/24
      - In the **Public Subnet 2 CIDR Range**, enter the CIDR range to use for the second public subnet, for example, 192.168.1.0/24
      - In the **Private Subnet 1 CIDR Range**, enter the CIDR range to use for the first private subnet, for example, 192.168.2.0/24
      - In the **Private Subnet 2 CIDR Range**, enter the CIDR range to use for the second private subnet, for example, 192.168.3.0/24



IMPORTANT

It is crucial to ensure the CIDR ranges provided for your subnets are within your VPC CIDR range and do not overlap with each other to avoid CIDR collisions.

You can look at the default subnet CIDR ranges as a point of reference.

   e. Ignore the fields under the **Existing network configuration** section.
3. Click **Next** to move to Step 3.
4. For Step 3, on the **Configure stack options** page, no changes are necessary.
   - All configurations are optional or have the correct default values.
5. Click **Next** to move to Step 4.
6. For Step 4, on the **Review** page, scroll to the bottom.

- In the **Capabilities** section, check to acknowledge that CloudFormation can create IAM resources.

7. Click **Submit**.
8. The application begins provisioning.
    - It can take some time for the infrastructure and application to fully provision.

If you want to modify your CIDR ranges post deployment, your current deployment must be deleted, then follow the instructions in Deploying an application with an existing VPC .

2.3.2. Deploying an application with an existing VPC

NOTE

The following procedure assumes that you have a marketplace offer. The instructions are a continuation from AWS Marketplace. Complete the procedure in that section before continuing with this section.

The following procedure uses an existing VPC network to deploy an application.

Procedure

1. For Step 1, on the **Create stack** page, click **Next**.
2. For Step 2, on the **Specify stack details** page
    a. In the **Stack name** field, enter a unique stack name.
    b. In the **EC2 KeyPair** field, select your previously created EC2 keypair.
    c. In the **Select VPC** field, select **Existing**.
    d. Ignore the fields under the **New network configuration** section.
    e. In the **Existing network configuration** section, fill the following fields
        - In the **Which VPC ID should this be deployed to?**field, enter the VpcId of your existing Virtual Private Cloud (VPC), for example, vpc-01234567890abcdef
        - In the **Existing VPC CIDR Range**, enter the CIDR Range of your existing Virtual Private Cloud (VPC), for example, 192.168.0.0/16
        - In the **Existing Private Subnet 1 ID**, enter the ID of your first existing private subnet, for example, subnet-077dd9969c32371f7
        - In the **Existing Private Subnet 2 ID**, enter the ID of your second existing private subnet, for example, subnet-077dd9969c32371f7
          Note that copying and pasting these from the AWS console introduces hidden characters, and can cause the entries to fail.
3. Click **Next** to move to Step 3.
4. For Step 3, on the **Configure stack options** page, no changes are necessary.

- All configurations are optional or have the correct default values.

5. Click **Next** to move to Step 4.
6. For Step 4, on the **Review** page, scroll to the bottom.
   - In the **Capabilities** section, check to acknowledge that CloudFormation can create IAM resources.
7. Click **Submit**.
8. The application begins provisioning.
   - It can take some time for the infrastructure and application to fully provision.

2.4. DEPLOYMENT INFORMATION

After deploying Ansible Automation Platform, use the following procedures to retrieve information about your deployment.

2.4.1. Retrieving the administration password

Use the following procedure to retrieve the administration password.

Procedure

1. In the AWS console, navigate to **AWS Secrets Manager**.
   - Ensure you are in the correct region
2. Filter with the name of the deployment. The administration secret is named **<DeploymentName>-aap-admin-secret**.
   - Replace **<DeploymentName>** with the name of your CloudFormation stack
3. Click on the administration secret name
4. In the **Secret value** section, click **Retrieve secret value**.
5. The administration username and password are displayed.

2.4.2. Retrieving the load balancer addresses

Use the following procedure to retrieve the controller and hub load balancer details.

Procedure

1. In the AWS UI, navigate to the CloudFormation UI.
   - Ensure you are in the correct region.
2. Select the deployment name under the **Stack name** column.
3. In the pane that opens, select the **Resources** tab.
   - Ensure **Flat view** is selected.

4. In the **Resources** tab search bar, search for **LoadBalancer**.

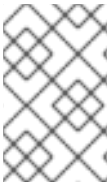
5. Click the **Physical ID** of the load balancers to open the load balancers page, with the proper load balancer preselected.

6. Click on the preselected load balancer to get further detailed information about the load balancer.

2.5. CREATE CREDENTIALS FOR THE COMMAND GENERATOR

You must have an AWS credentials file for all day-2 operations, such as adding and removing Extension Nodes.

The file must follow the format of the AWS credentials file for "Long-term credentials", which includes the **aws_access_key_id** and **aws_secret_access_key** variables. For further information, read AWS Command Line Interface User Guide in the AWS documentation.



NOTE

You must use the **default** profile name in the credentials file. The Command generator does not recognize any other name. For more information, read the Command generator - AWS Credentials File technical note.

An example of the credentials file format is:

```
[default]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

After the credentials file is created, you can use it in the command generator by passing the file path to the **-c** option, for example:

- **/home/user/project/extra_vars.yaml**

```
[default]
aws_remove_extension_nodes:
  cloud_credentials_path: ./my_credentials
  deployment_name:
  extra_vars:
    aws_autoscaling_group_name:
    aws_launch_template_name:
    aws_offer_type:
    aws_region:
    aws_ssm_bucket_name:
```

- **/home/user/project/my_credentials**

```
[default]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

Command:

```
docker run --rm \
        -v /home/user/project:/data:ro \
        $IMAGE \
        command_generator \
        --data-file /data/extra_vars.yml
```

CHAPTER 3. DEPLOYING EXTENSION NODES

The foundation stack is deployed with two automation controller nodes and one automation hub node by default. You can add execution nodes to scale out Ansible Automation Platform from AWS Marketplace. Extension node offers can be mixed and matched together to scale up and out the Ansible Automation Platform deployment.

You must create a backup of the Ansible Automation Platform deployment before deploying new extension nodes.

The procedure contains the following steps:

1. Decide the offer type for the extension nodes.
2. Pull the **ansible-on-clouds-ops** container image.
3. Ensure the minimum permissions are met to manage extension nodes.
4. Generate data files by running the **ansible-on-clouds-ops** container.
5. Populate the data file.
6. Run the **ansible-on-clouds-ops** container to deploy extension nodes.

Prerequisites

- Linux or macOS system (where the **ansible-on-clouds-ops** container image will run).
- Docker

3.1. DECIDING THE OFFER TYPE

The following table lists the offer types and the corresponding AWS instance types. Depending on the workload needs, you can choose a more suitable offer type for the extension nodes.

Offer Type	AWS Instance Type
100	m5.large
200	m5.xlarge
400	m5.2xlarge

3.2. GETTING AN EXTENSION NODE SUBSCRIPTION

Public Offer

Ansible Automation Platform from AWS Marketplace can be obtained directly from the AWS Marketplace. Follow these steps to purchase an extension node subscription.

Procedure

1. In the AWS Console, navigate to **AWS Marketplace Subscriptions**.
2. In the navigation bar, click **Discover Products**.
3. Click one of the listings for **Red Hat Ansible Automation Platform 2 Extension - Up to 100|200|400 Managed Nodes**. Whichever is suitable for your workload needs.
4. Click **Continue to Subscribe**.
5. Read terms, if you agree to them, click **Accept Terms**.
6. You should shortly be subscribed to an extnesion node offer.

Private Offer

If you have worked with Red Hat Sales to enable a private offer of Ansible Automation Platform from AWS Marketplace, follow these steps to accept your offer.

Procedure

1. Visit your Private Offer with the URL link provided by your Red Hat Sales representative.
2. Click **Accept Terms** to subscribe to the AMI Private Offer named Ansible Automation Platform from AWS Marketplace.

NOTE

If you are accepting terms for Ansible Automation Platform extension nodes, then assign the number of virtual machines arranged with your Red Hat Sales representative. The number of VMs directly correlates to the number of licensed managed active nodes that you have purchased.

After subscribing, the offer is also listed in your AWS Marketplace Subscriptions.

3.3. PULLING THE ANSIBLE-ON-CLOUDS-OPS CONTAINER IMAGE

Pull the Docker image for the Ansible on Clouds operational container with the same tag as the version you are deploying to.

NOTE

Before pulling the docker image, ensure you are logged in to registry.redhat.io using docker. Use the following command to login to registry.redhat.io.

```
$ docker login registry.redhat.io
```

For more information about registry login, see Registry Authentication

For example, if your foundation deployment version is 2.4.20230630-00, you must pull the operational image with tag 2.4.20230630 to deploy extension nodes to the foundation stack.

Use the following commands:

```
$ export IMAGE=registry.redhat.io/ansible-on-clouds/ansible-on-clouds-ops-rhel9:2.4.20230630
$ docker pull $IMAGE --platform=linux/amd64
```

3.4. IAM MINIMUM PERMISSIONS

You must have the following policies to manage both adding and removing the extension nodes.

```
required-roles:
  ec2:
    actions:
      - ec2:DeleteTags
      - ec2:DescribeAvailabilityZones
      - ec2:DescribeAccountAttributes
      - ec2:DeleteLaunchTemplate
      - ec2:DescribeLaunchTemplates
      - ec2:DescribeTags
      - ec2:CreateTags
      - ec2:DescribeLaunchTemplateVersions
      - ec2:RunInstances
      - ec2:CreateLaunchTemplateVersion
      - ec2:CreateLaunchTemplate
      - ec2:DescribeVpcs
      - ec2:DescribeInstanceTypes
      - ec2:DescribeSubnets
      - ec2:DescribeInstances
      - ec2:DescribeRegions
    resources:
      - "*"
  iam:
    actions:
      - iam:PassRole
      - iam:GetRole
      - iam:GetInstanceProfile
      - iam:GetRolePolicy
    resources:
      - "*"
  autoscaling:
    actions:
      - autoscaling:DescribeLaunchConfigurations
      - autoscaling:DescribeAutoScalingGroups
      - autoscaling:UpdateAutoScalingGroup
      - autoscaling:DescribeInstanceRefreshes
      - autoscaling:DeleteTags
      - autoscaling:DescribeTags
      - autoscaling:DescribeLifecycleHooks
      - autoscaling:StartInstanceRefresh
      - autoscaling:DisableMetricsCollection
      - autoscaling:CreateOrUpdateTags
      - autoscaling:DeleteAutoScalingGroup
      - autoscaling:CreateAutoScalingGroup
    resources:
      - "*"
  cloudformation:
    actions:
      - cloudformation:DescribeStackEvents
```

```
    - cloudformation:ListStackResources
    - cloudformation:ListStacks
    - cloudformation:DescribeStacks
    - cloudformation:GetTemplate
  resources:
    - "*"
elasticloadbalancing:
  actions:
    - elasticloadbalancing:DescribeLoadBalancers
    - elasticloadbalancing:DescribeTargetGroups
  resources:
    - "*"
s3:
  actions:
    - s3:GetBucketLocation
    - s3:DeleteObject
    - s3:PutObject
  resources:
    - "*"
ssm:
  actions:
    - ssm:StartSession
    - ssm:SendCommand
    - ssm:TerminateSession
  resources:
    - "*"
```

3.5. GENERATING DATA FILES BY RUNNING THE ANSIBLE-ON-CLOUDS-OPS CONTAINER

The following commands generate the required data file. These commands create a directory, and an empty data template that, when populated, is used during the deployment of the extension nodes.

Procedure

1. Create a folder to hold the configuration files.

```
$ mkdir command_generator_data
```

2. Populate the **command_generator_data** folder with the configuration file template.

NOTE

On Linux, any file or directory created by the command generator is owned by **root:root** by default. To change the ownership of the files and directories, you can run the **sudo chmod** command after the files are created. For more information, read Command generator - Linux files owned by root .

```
$ docker run --rm -v $(pwd)/command_generator_data:/data $IMAGE \
command_generator_vars aws_add_extension_nodes \
--output-data-file /data/extra_vars.yml
```

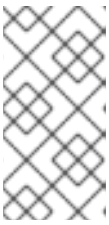
3. When you have run these commands, a **command_generator_data/extra_vars.yml** template file is created. This template file resembles the following:

```
# Create
aws_add_extension_nodes:
  ansible_config_path:
  cloud_credentials_path:
  deployment_name:
  extra_vars:
    aws_autoscaling_group_name:
    aws_extension_node_subscription:
    aws_launch_template_name:
    aws_offer_type:
    aws_region:
    aws_ssm_bucket_name:
    seller_name:
```

3.6. POPULATING THE DATA FILE

You must populate the data file before triggering the operation. The variables listed in the data file are defined below.

- **ansible_config_path** (Optional) is a value that overrides the default Ansible configuration used to run the playbook.
- **cloud_credentials_path** is the path to your AWS credentials file. For example, **~/.aws/credentials**
- **deployment_name** is the name of the foundation deployment. This is the same name you used when you deployed the foundation.
- **aws_autoscaling_group_name** is the name of the AWS AutoScaling Group to create for the extension nodes.
- **aws_extension_node_subscription** is a boolean value either **true** or **false** specifying whether or not you have an extension node subscription
- **aws_launch_template_name** is the name of the AWS EC2 launch template to create.
- **aws_offer_type** is the offer type of the extension nodes. This must be **100**, **200**, or **400**.
- **aws_region** is the region where the foundation deployment is located.
- **aws_ssm_bucket_name** is the name of the S3 bucket where temporary configuration files for the AWS SSM are stored. You can use an existing bucket or create a new one.



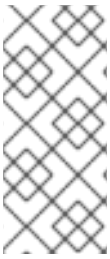
NOTE

The **aws_ssm_bucket_name** parameter is ONLY used to store temporary config files. It does NOT need to be saved for use in other playbooks. Any valid existing bucket can be used. For more information on creating S3 buckets, read AWS Creating A Bucket in the AWS documentation.

- **seller_name** (Optional) is used to designate the AWS Marketplace Seller. The default value is **redhatinc**. If you are based in EMEA and purchased this offering through the **redhatlimited** AWS Marketplace seller, ensure that this value is set to **redhatlimited**.

After populating the data file, it resembles the following:

The values are provided as examples.



NOTE

The optional values **ansible_config_path** and **seller_name** in this data file example have been removed. If you do not wish to use these optional values and want to use the default values for these variables, you 'must' also remove them for your data file like it was done in the example below. If you wish to use these optional variables, they 'must' be included in the data file and be assigned a value.

```
aws_add_extension_nodes:
  cloud_credentials_path: ~/.aws/credentials
  deployment_name: AnsibleAutomationPlatform
  extra_vars:
    aws_autoscaling_group_name: AnsibleAutomationPlatform-ext-asg1-100
    aws_extension_node_subscription: true
    aws_launch_template_name: AnsibleAutomationPlatform-ext-lt1-100
    aws_offer_type: "100"
    aws_region: us-east-1
    aws_ssm_bucket_name: aap-ssm-bucket
```

3.7. DEPLOYING THE EXTENSION NODES

Procedure

1. To deploy the extension nodes, run the command generator to generate the CLI command.

   ```
   $ docker run --rm -v $(pwd)/command_generator_data:/data $IMAGE command_generator --
   data-file /data/extra_vars.yml
   ```

   Provides the following command:

   ```
   ------------------------------------------------
   docker run --rm --env PLATFORM=AWS -v
   ~/.aws/credentials:/home/runner/.aws/credentials:ro \
   --env ANSIBLE_CONFIG=../aws-ansible.cfg --env
   DEPLOYMENT_NAME=AnsibleAutomationPlatform \
   --env GENERATE_INVENTORY=true  $IMAGE
   redhat.ansible_on_clouds.aws_add_extension_nodes \
   -e 'aws_foundation_stack_name=AnsibleAutomationPlatform aws_region=us-east-1 \
   aws_launch_template_name=AnsibleAutomationPlatform-ext-lt1-100 \
   aws_autoscaling_group_name=AnsibleAutomationPlatform-ext-asg1-100 \
   aws_extension_node_subscription=True aws_offer_type=100 \
   aws_ssm_bucket_name=aap-ssm-bucket'
   ================================================
   ```

2. Run the supplied command to add the extension nodes.

```
$ docker run --rm --env PLATFORM=AWS -v
~/.aws/credentials:/home/runner/.aws/credentials:ro \
--env ANSIBLE_CONFIG=../aws-ansible.cfg --env
DEPLOYMENT_NAME=AnsibleAutomationPlatform \
--env GENERATE_INVENTORY=true  $IMAGE
redhat.ansible_on_clouds.aws_add_extension_nodes \
-e 'aws_foundation_stack_name=AnsibleAutomationPlatform aws_region=us-east-1 \
aws_launch_template_name=AnsibleAutomationPlatform-ext-lt1-100 \
aws_autoscaling_group_name=AnsibleAutomationPlatform-ext-asg1-100 \
aws_extension_node_subscription=True aws_offer_type=100 \
aws_ssm_bucket_name=aap-ssm-bucket'
```

CHAPTER 4. REMOVING EXTENSION NODES

You must create a backup of the Ansible Automation Platform deployment before removing extension nodes.

Follow these steps to remove execution nodes from your Ansible Automation Platform from AWS Marketplace environment.

Prerequisites

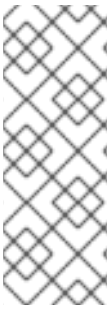
- Linux or macOS system (where the **ansible-on-clouds-ops** container image will run)
- Docker

Steps

1. Pull the **ansible-on-clouds-ops** container image.
2. Generate data files by running the ansible-on-clouds-ops container.
3. Update the data file.
4. Run the **ansible-on-clouds-ops** container to remove the extension nodes.

4.1. PULLING THE ANSIBLE-ON-CLOUDS-OPS CONTAINER IMAGE

Pull the Docker image for the Ansible on Clouds operational container which aligns with the version of your foundation deployment.



NOTE

Before pulling the docker image, ensure you are logged in to registry.redhat.io using docker. Use the following command to login to registry.redhat.io.

```
$ docker login registry.redhat.io
```

For more information about registry login, see Registry Authentication

For example, if your foundation deployment version is 2.4.20230630-00, you must pull the operational image with tag 2.4.20230630.

Use the following commands:

```
$ export IMAGE=registry.redhat.io/ansible-on-clouds/ansible-on-clouds-ops-rhel9:2.4.20230630
$ docker pull $IMAGE --platform=linux/amd64
```

4.2. IAM MINIMUM PERMISSIONS

You must have the following policies to manage both adding and removing the extension nodes.

```
required-roles:
  ec2:
```

```
  actions:
    - ec2:DeleteTags
    - ec2:DescribeAvailabilityZones
    - ec2:DescribeAccountAttributes
    - ec2:DeleteLaunchTemplate
    - ec2:DescribeLaunchTemplates
    - ec2:DescribeTags
    - ec2:CreateTags
    - ec2:DescribeLaunchTemplateVersions
    - ec2:RunInstances
    - ec2:CreateLaunchTemplateVersion
    - ec2:CreateLaunchTemplate
    - ec2:DescribeVpcs
    - ec2:DescribeInstanceTypes
    - ec2:DescribeSubnets
    - ec2:DescribeInstances
    - ec2:DescribeRegions
  resources:
    - "*"
iam:
  actions:
    - iam:PassRole
    - iam:GetRole
    - iam:GetInstanceProfile
    - iam:GetRolePolicy
  resources:
    - "*"
autoscaling:
  actions:
    - autoscaling:DescribeLaunchConfigurations
    - autoscaling:DescribeAutoScalingGroups
    - autoscaling:UpdateAutoScalingGroup
    - autoscaling:DescribeInstanceRefreshes
    - autoscaling:DeleteTags
    - autoscaling:DescribeTags
    - autoscaling:DescribeLifecycleHooks
    - autoscaling:StartInstanceRefresh
    - autoscaling:DisableMetricsCollection
    - autoscaling:CreateOrUpdateTags
    - autoscaling:DeleteAutoScalingGroup
    - autoscaling:CreateAutoScalingGroup
  resources:
    - "*"
cloudformation:
  actions:
    - cloudformation:DescribeStackEvents
    - cloudformation:ListStackResources
    - cloudformation:ListStacks
    - cloudformation:DescribeStacks
    - cloudformation:GetTemplate
  resources:
    - "*"
elasticloadbalancing:
  actions:
    - elasticloadbalancing:DescribeLoadBalancers
    - elasticloadbalancing:DescribeTargetGroups
```

```
  resources:
    - "*"
s3:
  actions:
    - s3:GetBucketLocation
    - s3:DeleteObject
    - s3:PutObject
  resources:
    - "*"
ssm:
  actions:
    - ssm:StartSession
    - ssm:SendCommand
    - ssm:TerminateSession
  resources:
    - "*"
```

4.3. GENERATING DATA FILES BY RUNNING THE ANSIBLE-ON-CLOUDS-OPS CONTAINER

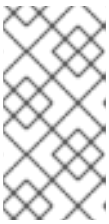
The following commands generate the required data file. These commands create a directory, and an empty data template that, when populated, is used during the upgrade.

Procedure

1. Create a folder to hold the configuration files.

```
$ mkdir command_generator_data
```

2. Populate the **$(pwd)/command_generator_data** folder with the configuration file template.



NOTE

On Linux, any file or directory created by the command generator is owned by **root:root** by default. To change the ownership of the files and directories, you can run the **sudo chmod** command after the files are created. For more information, read Command generator - Linux files owned by root .

```
$ docker run --rm -v $(pwd)/command_generator_data:/data $IMAGE \
command_generator_vars aws_remove_extension_nodes \
--output-data-file /data/extra_vars.yml
```

3. When you have run these commands, a **command_generator_data/extra_vars.yml** template file is created. This template file resembles the following:

```
aws_remove_extension_nodes:
  ansible_config_path:
  cloud_credentials_path:
  deployment_name:
  extra_vars:
    aws_autoscaling_group_name:
```

```
aws_launch_template_name:
aws_region:
aws_ssm_bucket_name:
```

4.4. UPDATE THE DATA FILE

You must populate the data file before triggering the operation. The variables listed in the data file are defined below.

- **ansible_config_path** (Optional) is a value that overrides the default Ansible configuration used to run the playbook.
- **cloud_credentials_path** is the path to your AWS credentials file. For example, **~/.aws/credentials**
- **deployment_name** is the name of the foundation deployment. This is the same name you used when you deployed the foundation.
- **aws_autoscaling_group_name** is the name of the AWS AutoScaling Group to create for the extension nodes.
- **aws_launch_template_name** is the name of the AWS EC2 launch template to create.
- **aws_region** is the region where the foundation deployment is located.
- **aws_ssm_bucket_name** is the name of the S3 bucket where temporary configuration files for the AWS SSM are stored. You can use an existing bucket or create a new one.

NOTE

The **aws_ssm_bucket_name** parameter is ONLY used to store temporary config files. It does NOT need to be saved for use in other playbooks. Any valid existing bucket can be used. For more information on creating S3 buckets, read AWS Creating A Bucket in the AWS documentation.

After populating the data file, it should resemble the following:

The values below are provided as examples.

NOTE

The optional value **ansible_config_path** in this data file example has been removed. If you do not wish to use this optional value and want to use the default value for these variable, you 'must' also remove it for your data file like it was done in the example below. If you wish to use this optional variable, then it 'must' be included in the data file and be assigned a value.

```
aws_remove_extension_nodes:
  cloud_credentials_path: ~/.aws/credentials
  deployment_name: AnsibleAutomationPlatform
  extra_vars:
    aws_autoscaling_group_name: AnsibleAutomationPlatform-ext-asg1-100
```

```
aws_launch_template_name: AnsibleAutomationPlatform-ext-lt1-100
aws_region: us-east-1
aws_ssm_bucket_name: aap-ssm-bucket
```

4.5. RUNNING THE ANSIBLE-ON-CLOUDS-OPS CONTAINER TO REMOVE THE EXTENSION NODES

Procedure

1. To remove a set of extension nodes, run the command generator to generate the CLI command.

```
$ docker run --rm -v $(pwd)/command_generator_data:/data $IMAGE command_generator --data-file /data/extra_vars.yml
```

The command generator output provides the following command:

```
docker run --rm --env PLATFORM=AWS -v ~/.aws/credentials:/home/runner/.aws/credentials:ro --env ANSIBLE_CONFIG=../aws-ansible.cfg \
--env DEPLOYMENT_NAME=AnsibleAutomationPlatform --env GENERATE_INVENTORY=true  $IMAGE redhat.ansible_on_clouds.aws_remove_extension_nodes \
-e 'aws_foundation_stack_name=AnsibleAutomationPlatform aws_region=us-east-1 aws_launch_template_name=AnsibleAutomationPlatform-ext-lt1-100 \
aws_autoscaling_group_name=AnsibleAutomationPlatform-ext-asg1-100 \
aws_ssm_bucket_name=aap-ssm-bucket'
```

2. Run the supplied upgrade command to remove a set of extension nodes.

```
$ docker run --rm --env PLATFORM=AWS -v ~/.aws/credentials:/home/runner/.aws/credentials:ro --env ANSIBLE_CONFIG=../aws-ansible.cfg \
--env DEPLOYMENT_NAME=AnsibleAutomationPlatform --env GENERATE_INVENTORY=true  $IMAGE redhat.ansible_on_clouds.aws_remove_extension_nodes \
-e 'aws_foundation_stack_name=AnsibleAutomationPlatform aws_region=us-east-1 aws_launch_template_name=AnsibleAutomationPlatform-ext-lt1-100 \
aws_autoscaling_group_name=AnsibleAutomationPlatform-ext-asg1-100 \
aws_ssm_bucket_name=aap-ssm-bucket'
```

CHAPTER 5. NETWORKING AND APPLICATION ACCESS

5.1. ACCESSING THE APPLICATION

When Ansible Automation Platform from AWS Marketplace is deployed, it is deployed into an isolated VPC and cannot be accessed. The following sections describe how to connect the VPC used by Ansible Automation Platform from AWS Marketplace to your existing AWS network.

When connected, you must determine how your users connect to Ansible Automation Platform.

There are many ways to enable this connectivity such as VPNs, Direct Connect, or bastion servers for private network access. You can also expose the platform with public internet access using AWS services such as a Load Balancer. How your organization configures application access on AWS is outside the scope of Red Hat's guidelines and support for Ansible Automation Platform from AWS Marketplace. Red Hat recommends contacting Amazon Web Services for guidelines on these products and topics.

5.2. NETWORK PEERING OPTIONS

Many networking configurations are possible, but the following configurations have been validated to work with Ansible Automation Platform from AWS Marketplace by Red Hat.

Private deployments omit access from the public internet. This is the default deployment model based on the AWS architecture used by Ansible Automation Platform from AWS Marketplace.

If you use this approach you must configure VPC peering, a transit network gateway, or similar AWS networking connectivity to access Ansible Automation Platform from AWS Marketplace.

When VPC peering and routing are configured, you can access Ansible Automation Platform through a VM on a connected VPC subnet, or directly if your organization has a transit routing setup between AWS and your local networks.

Network peering is required for Ansible Automation Platform to access resources that reside on private VPCs or where transit routing between AWS and your on-premises networks exists.

Ansible Automation Platform requires peering and routing for outbound communication to perform automations against devices on your AWS networks and connected on-premise or multi-cloud networks.

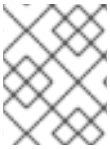
NOTE

While every effort has been made to align with Amazon Web Services's documentation for this content, there may be a drift in accuracy over time. Use Amazon Web Services documentation as the source of information regarding networking topics for AWS.

AWS offers different ways to peer private networks together. This document focuses on basic network peering and routing models:

Peering type	Description
VPC Peering	VPCs are individually connected to one another with no other routing hops between them. This is a simple peering model and is useful when connecting a few networks. Complex peering can be configured, but routing can become more complex over time.

Peering type	Description
Transit Gateway	A transit gateway is a network transit hub that you can use to interconnect your virtual private clouds (VPCs) and on-premises networks. As your cloud infrastructure expands globally, inter-Region peering connects transit gateways together using the AWS Global Infrastructure. Your data is automatically encrypted and never travels over the public internet.

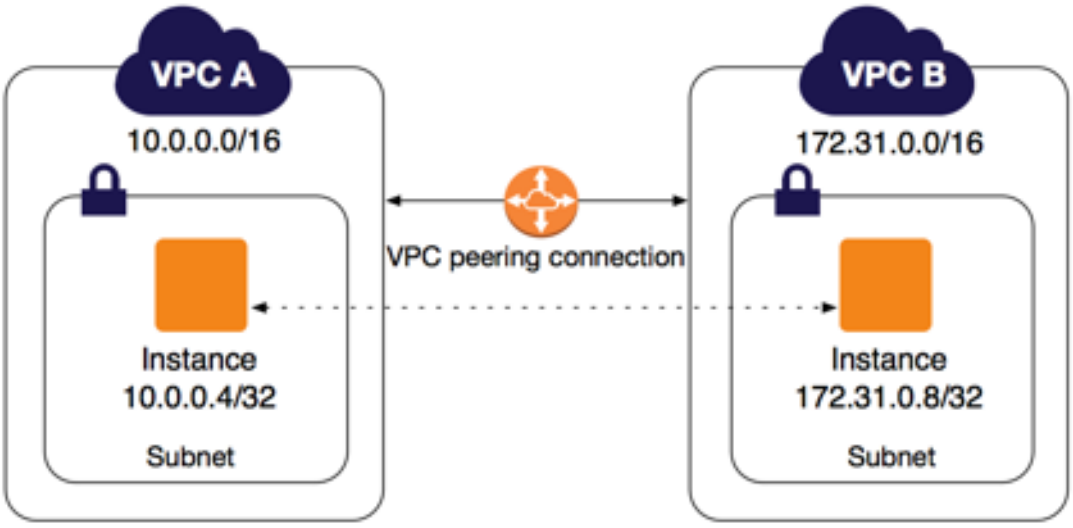


NOTE

There are also ways that AWS directs organizations not to attempt to peer networks in their Invalid network peering document.

5.3. VPC PEERING

VPC Peering offers the ability to directly connect different networks within your AWS infrastructure. When two or more networks are peered, the peering only establishes the connection between the networks. You must also update routing tables to direct the networks to send traffic over the appropriate networking peer.



Prerequisites

Before using VPC peering to connect any VPC, you must ensure that there is no network address space overlap between the networks that you intend to route traffic between your VPCs, and Ansible Automation Platform from AWS Marketplace's VPC address space.

The following procedure lets you configure VPC peering with Ansible Automation Platform.

Procedure

1. In the AWS Portal, navigate to **VPC**.
2. In the **VPC menu**, click **Peering Connections**.
3. Click **Create peering connection**.

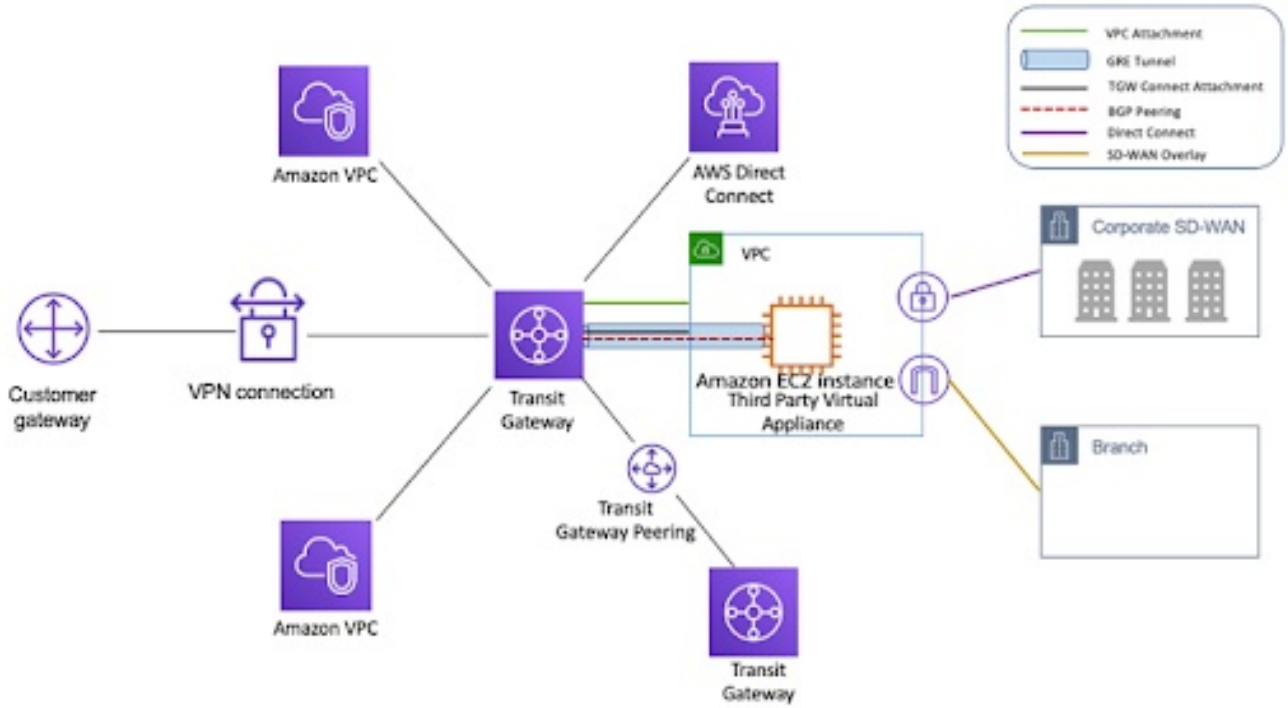
4. Fill out the following fields:
    a. In the **Name** field, enter the name of the peering connection as you need.
    b. In the **Select a local VPC to peer with** field, enter the Virtual Private Cloud. This can be your existing VPC.
    c. In the **Select another VPC to peer with** field, enter the name of the Virtual Private Cloud. This can be the Ansible Automation Platform from AWS Marketplace VPC
    d. Select **My Account** as these two networks should exist in the same account.
    e. Choose the appropriate region configuration based on your VPC layout.
5. Click **Create peering connection**.
6. On the next screen, the summary page for the peering, select the **Actions** menu in the top right and select **Accept request**.

When you have configured the direct network peering you must configure the route tables, security groups, and Network ACLs for both VPCs for traffic to route properly.

For additional information on route tables, security groups and other networking components regarding VPC Peering, see What is VPC peering? in the AWS documentation.

5.4. TRANSIT GATEWAYS

Transit Gateways provide a more advanced hub-and-spoke peering model than direct VPC peering provides. When Ansible Automation Platform from AWS Marketplace is peered to this type of model, the hub has an additional peering with the CIDR range deployed for the Ansible Automation Platform from AWS Marketplace VPC. The Ansible Automation Platform from AWS Marketplace VPC can be peered to a transit gateway to participate in the peering model that this AWS service provides. Follow Instructions from AWS for configuring a transit gateway.



CHAPTER 6. ADDITIONAL CONFIGURATIONS

If you want to use automation hub to host execution environments, you must first configure the automation hub's Application Load Balancer, which also involves trusting the self-signed certificate, **update-ca-trust** on the EC2 Instance.

6.1. CHANGING THE DEFAULT ANSIBLE AUTOMATION PLATFORM ADMINISTRATOR PASSWORD

The default Administrator password for Ansible Automation Platform is generated randomly when Ansible Automation Platform from AWS Marketplace is deployed. Follow these steps to change the Administrator password for both automation controller and automation hub:

Procedure

1. Navigate to the AWS Secrets Manager console.
    a. Locate and open the secret for the CloudFormation Stack with the name **<stack_name>-aap-admin-secret**.
    b. Select **Retrieve secret value** to view the secret value.
    c. Select **Edit** and provide an updated password value.
    d. Click **Save**.
2. Change the running Ansible Automation Platform EC2 instances to use the new Administrator password.
    a. Navigate to the AWS EC2 Instances console.
    b. Identify and Terminate one automation controller instance and one automation hub instance for the CloudFormation Stack.
    c. Wait for the automation controller and automation hub Auto Scaling Groups to create new EC2 instances.
3. The new Administrator can be used when the new automation controller and automation hub EC2 instances reach a *Running* instance state.

6.2. CHANGING THE DEFAULT RDS DATABASE PASSWORD

The default RDS database password for Ansible Automation Platform is generated randomly when Ansible Automation Platform from AWS Marketplace is deployed. Follow these steps to change the RDS password:

Procedure

1. Navigate to the AWS Secrets Manager console.
    a. Locate and open the secret for the CloudFormation Stack with the name **<stack_name>-aap-rds-secret**.
    b. Selecting **Retrieve secret value** to view the secret value.

c. Select **Edit** and provide an updated password value.

d. Click **Save**.

2. Update the automation hub EC2 instance with the new RDS password.

a. Navigate to the AWS EC2 Instances console

b. Identify and connect to the automation hub instance for the CloudFormation Stack.

c. Install the **postgresql** package using

```
# dnf install postgresql
```

d. View the database settings from the following file - **/etc/pulp/settings.py**. Variables shown here are used in following steps.

e. Run

```
$ psql -h <DB_HOST> -U awx
```

i. When prompted for password: enter <DB_PASSWORD>

f. Run

```
$ ALTER USER awx WITH PASSWORD <UPDATED_PASSWORD>;
```

3. Change the running Ansible Automation Platform EC2 instances to use the new RDS password.

a. Navigate to the AWS EC2 Instances console.

b. Terminate all EC2 instances. The autoscale group restarts them.

c. Wait for the automation controller and automation hub Auto Scaling Groups to create new EC2 instances.

d. The new RDS can be used when the new automation controller and automation hub EC2 instances reach a *Running* Instance State.

4. To validate the RDS password has been changed:

a. Identify and connect to the automation hub instance for the CloudFormation Stack.

b. View the updated database settings from the following file - **/etc/pulp/settings.py**.

c. Attempt to connect using the following sample connection string:

```
psql postgresql://<DB_USER>:<DB_PASSWORD>@<DB_HOST>:
<DB_PORT>/<DB_NAME>
```

Variable definitions can be found in the settings file.

6.3. REPLACING AUTOMATION CONTROLLER AND AUTOMATION HUB EC2 INSTANCES SSL/TLS CERTIFICATE AND KEY

By default, EC2 instances are secured with a self-signed SSL/TLS certificate with a validity period of ten years. When the certificate expires or you want EC2 instances to use your own certificate, you are required to replace the SSL/TLS certificate and key.

Procedure

1. Navigate to the AWS Secrets Manager console.
    a. Locate and open the secret for the CloudFormation Stack with the name **<stack_name>-pulp_cert**.
    b. Select **Retrieve secret value** to view the secret value.
    c. Select **Edit** and provide new SSL/TLS certificate value.
    d. Click **Save**.
2. Navigate to the AWS Secrets Manager console.
    a. Locate and open the secret for the CloudFormation Stack with the name **<stack_name>-pulp_key**.
    b. Select **Retrieve secret value** to view the secret value.
    c. Select **Edit** and provide new SSL/TLS key value.
    d. Click **Save**.
3. Change the running Ansible Automation Platform EC2 instances to use the new SSL/TLS certificate and key.
    a. Navigate to the AWS EC2 Instances console.
    b. Identify and terminate all automation controller and automation hub instances for the CloudFormation Stack.
    c. Wait for the automation controller and automation hub Auto Scaling Groups to create new EC2 instances.
4. The new certificate is in use when the new automation controller and automation hub EC2 instances reach a *Running* instance state.

6.4. SECURING THE CONNECTION BETWEEN AUTOMATION HUB AND AUTOMATION CONTROLLER

The following procedure describes how to secure the automation hub load balancer

Prerequisites

- Ensure you are using openssl v3 or later.

Procedure

1. Generate the automation hub certificate with the following command:

```
$ openssl req -x509 -nodes -newkey rsa:4096 -keyout hub_key.pem -out hub_cert.pem -
sha256 -days 365 -addext "subjectAltName = DNS:<HUB_DNS_NAME>"
```

where **HUB_DNS_NAME** is the DNS name of the hub loadbalancer. Certificates can also be requested or imported into AWS.

2. In the Amazon Web Services UI, navigate to **Amazon Certificate Manager**. Ensure you are in the correct region.
3. Click **Import** to start the import of the generated certificate.
4. Paste the contents of **hub_cert.pem** into the **Certificate body** field.
5. Paste the contents of **hub_key.pem** into the **Certificate private key** field.
6. Click **Next**
7. Optional: Add any additional tags you require.
8. Click **Next**.
9. Click **Import**.

Updating the automation hub internal Load Balancer Listener

Procedure

1. In the Amazon Web Services UI, navigate to **EC2 Load Balancers**. Ensure you are in the correct region.
2. Select the internal load balancer for automation hub.
3. Click the **Listeners** tab.
4. Click **Add listener**.
5. Select **HTTPS** for the protocol, and ensure the port is **443**.
6. Select **Forward** for the default actions, and ensure the target group is **automation hub** instance group.
7. Select **Secure listener settings > Default SSL/TLS certificate** and ensure that **From ACM** is selected.
8. Select your imported ACM certificate.
9. Click **Add**.

Updating the load balancer security group

Procedure

1. In the Amazon Web Services UI, navigate to **EC2 Security Groups**. Ensure you are in the correct region.
2. Select the automation hub **ALB Security group**.

3. Select the **Inbound rules** tab and click **Edit inbound rules**.
4. Add a rule of Type: **HTTPS**, with Source: **Anywhere-IPv4**.
5. Delete the old HTTP rule and click **Save rules**.

6.5. SECURING THE CONTROLLER LOAD BALANCER

The following procedure describes how to secure the automation controller load balancer.

Prereqisites

- Ensure you are using openssl 3 or later

Procedure

1. Generate the automation controller certificate with the following command:

```
$ openssl req -x509 -nodes -newkey rsa:4096 -keyout controller_key.pem -out
controller_cert.pem -sha256 -days 365 -addext "subjectAltName = DNS:
<CONTROLLER_DNS_NAME>"
```

   where **CONTROLLER_DNS_NAME** is the DNS name of the controller load balancer. Certificates can also be requested or imported into AWS.

2. In the Amazon Web Services UI, navigate to **ACM**. Ensure you are in the correct region.
3. Click **Import** to start the import of the generated certificate.
4. Paste the contents of **controller_cert.pem** into the **Certificate body** field.
5. Paste the contents of **controller_key.pem** into the **Certificate private key** field.
6. Click **Next**
7. Optional: Add any additional tags you require.
8. Click **Next**.
9. Click **Import**.

Updating the automation controller internal load balancer listener

Procedure

1. In the Amazon Web Services UI, navigate to **EC2 Load Balancers**. Ensure you are in the correct region.
2. Select the internal load balancer for automation controller.
3. Click the **Listeners** tab.
4. Click **Add listener**.
5. Select **HTTPS** for the protocol, and ensure the port is **443**.

6. Select **Forward** for the default actions, and ensure the target group is **automation controller** instance group.
7. Select **Secure listener settings > Default SSL/TLS certificate** and ensure that **From ACM** is selected.
8. Select your imported ACM certificate.
9. Click **Add**.

Updating the load balancer security group

Procedure

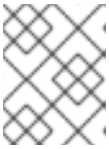
1. In the Amazon Web Services UI, navigate to **EC2 Security Groups**. Ensure you are in the correct region.
2. Select the automation controller **ALB Security group**.
3. Select the **Inbound rules** tab and click **Edit inbound rules**.
4. Add a rule of Type: **HTTPS**, with Source: **Anywhere-IPv4**.
5. Delete the old HTTP rule and click **Save rules**.

CHAPTER 7. THE COMMAND GENERATOR

The Command Generator is used to generate commands for launching operational playbooks provided by the Ansible-on-clouds operational playbook collection.

The process involves five steps:

1. Pull the **ansible-on-clouds-ops** container image.
2. List the available playbooks.
3. Use a command generator to generate a data file and the next command to run. **command_generator_vars** and the command_generator are implemented using a docker container and are run using the docker command line interface.
4. Populate the data file and run the previous generated command. This generates the final command with all parameters.



NOTE

When this step is complete, you can save the generated command and run the playbook when it is required.

5. Run the final command.

Prerequisites

- Docker
- An AWS credentials file
- An internet connection to Amazon Web Services

7.1. PULLING THE ANSIBLE-ON-CLOUDS-OPS CONTAINER IMAGE

Pull the Docker image for the Ansible on Clouds operational container with the same tag version as your deployment.

NOTE

Before Pulling the docker image, ensure you are logged in to registry.redhat.io using docker. Use the following command to login to registry.redhat.io.

```
$ docker login registry.redhat.io
```

For more information about registry login, see Registry Authentication

For example, if your foundation deployment version is 2.4.20230630-00, you must pull the operational image with tag 2.4.20230630.

Use the following commands:

```
$ export IMAGE=registry.redhat.io/ansible-on-clouds/ansible-on-clouds-ops-rhel9:2.4.20230630
$ docker pull $IMAGE --platform=linux/amd64
```

7.2. LISTING THE AVAILABLE PLAYBOOKS

Procedure

1. For the list of available playbooks without details, use the following command.

```
$ docker run --rm $IMAGE command_generator_vars | grep Playbook

The current version of the operational playbooks collection contains the following playbooks:

Playbook: aws_add_extension_nodes
Playbook: aws_add_labels
Playbook: aws_backup_delete
Playbook: aws_backup_stack
Playbook: aws_backups_delete
Playbook: aws_check_aoc_version
Playbook: aws_deployment_inventory
Playbook: aws_get_aoc_version
Playbook: aws_remove_extension_nodes
Playbook: aws_remove_labels
Playbook: aws_restore_stack
Playbook: aws_upgrade
Playbook: gcp_aap_health_check
Playbook: gcp_add_extension_nodes
Playbook: gcp_add_labels
Playbook: gcp_backup_delete
Playbook: gcp_backup_deployment
Playbook: gcp_backup_list
Playbook: gcp_backups_delete
Playbook: gcp_check_aoc_version
Playbook: gcp_deployment_inventory
Playbook: gcp_get_aoc_version
Playbook: gcp_health_check
Playbook: gcp_list_deployments
Playbook: gcp_nodes_health_check
Playbook: gcp_remove_extension_nodes
Playbook: gcp_remove_labels
Playbook: gcp_restore_deployment
Playbook: gcp_setup_logging_monitoring
Playbook: gcp_upgrade
```

2. To provide a list of all available playbooks, and to use the command generator, use the following command.

```
$ docker run --rm $IMAGE command_generator_vars
```

This provides a list of playbooks and a command similar to the following:

```
================================================
Playbook: aws_add_extension_nodes
```

```
Description: Add extension nodes to an existing Ansible Automation Platform from AWS
Marketplace stack
---------------------------------------------
This playbook is used to deploy extension nodes to an existing Ansible Automation Platform
from AWS Marketplace environment.
For more information regarding extension nodes, visit our official documentation -
https://access.redhat.com/documentation/en-
us/ansible_on_clouds/2.x/html/red_hat_ansible_automation_platform_from_aws_marketplace_g
uide/assembly-aap-aws-extension

---------------------------------------------
Command generator template:

docker run --rm $IMAGE command_generator aws_add_extension_nodes [--ansible-config
ansible_config_path>] \
-d <deployment_name> -c <cloud_credentials_path> \
--extra-vars 'aws_region=<aws_region> aws_launch_template_name=
<aws_launch_template_name> aws_autoscaling_group_name=
<aws_autoscaling_group_name> aws_asg_min_size=<aws_asg_min_size>
aws_asg_desired_capacity=<aws_asg_desired_capacity> aws_offer_type=
<aws_offer_type> [seller_name=<seller_name>]'
=============================================
```

7.3. GENERATING THE DATA FILE

Procedure

1. Run the command generator.

```
$ docker run --rm -v <local_directory_data_file>:/data $IMAGE command_generator_vars
<playbook_name> --output-data-file /data/<data-file>.yml
```

The outputs of this command are the command to run and the data file template. The data file is also saved in your **<local_data_file_directory>**. This is the template which you populate with your data.

The following example uses the **aws_backup_stack** playbook.

```
$ docker run --rm -v <local_data_file_directory>:/data $IMAGE command_generator_vars
aws_backup_stack \
--output-data-file /data/backup.yml
```

2. Producing the following output.

```
=============================================
Playbook: aws_backup_stack
Description: This playbook is used to backup the Ansible Automation Platform from AWS
Marketplace environment.
---------------------------------------------
This playbook is used to backup the Ansible Automation Platform from AWS Marketplace
environment.
For more information regarding backup and restore, visit our official documentation -

---------------------------------------------
```

```
Command generator template:

docker run --rm -v /tmp:/data $IMAGE command_generator aws_backup_deployment --data-file /data/backup.yml

Data template:

aws_backup_stack:
  cloud_credentials_path:
  deployment_name:
  extra_vars:
    aws_backup_iam_role:
 aws_backup_vault_name:
    aws_region:
    aws_s3_bucket:

================================================
```

7.4. POPULATING THE DATA FILE

Procedure

- Edit the data-file generated in Generating the data file.
  Any attribute that represents a path must be an absolute path. The **command_generator** automatically mounts a volume for it in the final command.

  For example, in the case of the **aws_backup_stack** playbook, the file becomes:

```
aws_backup_stack:
  cloud_credentials_path: ~/.aws/credentials
  deployment_name: AnsibleAutomationPlatform
  extra_vars:
    aws_backup_iam_role: arn:aws:iam::<Your AWS Account Number>:role/service-role/AWSBackupDefaultServiceRole
 aws_backup_vault_name: Default
    aws_region: us-east-1
    aws_s3_bucket: ansible-automation-platform-bucket
```

7.5. RUNNING THE GENERATED COMMAND

Procedure

1. Ensure that the mounted volume points to the directory where the data file is.
   For the **aws_backup_stack** playbook example, this is:

```
$ docker run --rm -v /tmp:/data $IMAGE command_generator aws_backup_stack --data-file /data/backup.yml
```

   Which generates the following output:

```
Command to run playbook:
```

```
$ docker run --rm --env PLATFORM=AWS -v
~/.aws/credentials:/home/runner/.aws/credentials:ro \
--env ANSIBLE_CONFIG=../aws-ansible.cfg  $IMAGE
redhat.ansible_on_clouds.aws_backup_stack \
-e 'aws_foundation_stack_name=AnsibleAutomationPlatform aws_region=us-east-1 \
aws_backup_vault_name=Default \
aws_backup_iam_role_arn=arn:aws:iam::123456789012:role/service-
role/AWSBackupDefaultServiceRole \ aws_s3_bucket=AnsibleAutomationPlatform-bucket'
```

This new command has the required parameters, environment variables and mounted volumes to run the playbook.

2. Run the generated command. You can save this command to rerun it later if required.

7.6. USING THE PLAYBOOKS

Some, but not all of the playbooks are described in this document. Here, those which are used either to retrieve information from the Ansible on Clouds deployment or to check it are described. These playbooks do not modify the deployment.

aws_add_labels

This playbook adds labels to the deployment.

```
$ docker run --rm $IMAGE command_generator_vars aws_add_labels
```

Which generates the following output:

```
================================================
Playbook: aws_add_labels
Description: This playbook adds labels to the deployment.
------------------------------------------------
Add labels to the Ansible Automation Platform from AWS Marketplace deployment.

------------------------------------------------
Command generator template:

docker run --rm $IMAGE command_generator aws_add_labels -d <deployment_name> -c
<cloud_credentials_path> --extra-vars 'aws_region=<aws_region> aws_labels=<aws_labels>'
================================================
```

The parameter **aws_labels** is a comma separated list of **key=value** pairs to add or update. For example: key1=value1,key2=value2

Launching this command by replacing the parameters generates a new command to launch and outputs:

```
...
PLAY RECAP *********************************************************************
localhost                  : ok=22   changed=2    unreachable=0    failed=0    skipped=1    rescued=0
ignored=0
```

aws_check_aoc_version

This playbook checks whether the Ansible on Cloud version is the same as the command generator container. The check is done each time a playbook is called.

```
$ docker run --rm $IMAGE command_generator_vars aws_check_aoc_version
```

Which generates the following output:

```
================================================
Playbook: aws_check_aoc_version
Description: Check the operational container version matches the Ansible on Clouds version.
------------------------------------------------
Check the operational container version matches the Ansible on Clouds version.

------------------------------------------------
Command generator template:

docker run --rm $IMAGE command_generator aws_check_aoc_version [--ansible-config
ansible_config_path>] -d <deployment_name> -c <cloud_credentials_path> --extra-vars
'aws_region=<aws_region> aws_ssm_bucket_name=<aws_ssm_bucket_name>'
================================================
```

Launching this command by replacing the parameters generates a new command to launch and outputs:

```
...
TASK [redhat.ansible_on_clouds.standalone_check_aoc_version : Verify operational playbook and
Ansible on Clouds deployment versions] ***
fatal: [localhost]: FAILED! => {
    "assertion": "ops_version == aoc_version",
    "changed": false,
    "evaluated_to": false,
    "msg": "This operation playbook version 2.4.20230606-00 is not valid for the Ansible on Clouds
deployment version 2.4.20230531-00"
}

PLAY RECAP *********************************************************************
localhost                  : ok=7    changed=0    unreachable=0    failed=1    skipped=0    rescued=0
ignored=0
```

A *failed* not equal zero indicates that the Ansible on Clouds deployment version does not match the **command_generator** container and a different version is required for the command generator to manage that deployment.

aws_get_aoc_version

This playbook retrieves the version of the Ansible on Clouds deployment.

```
$ docker run --rm $IMAGE command_generator_vars aws_get_aoc_version
```

Which generates the following output:

```
================================================
Playbook: aws_get_aoc_version
Description: Get the current Ansible on Clouds version.
------------------------------------------------
Get the current Ansible on Clouds version.

------------------------------------------------
Command generator template:
```

```
docker run --rm $IMAGE command_generator aws_get_aoc_version [--ansible-config
ansible_config_path>] -d <deployment_name> -c <cloud_credentials_path> --extra-vars
'aws_region=<aws_region> aws_ssm_bucket_name=<aws_ssm_bucket_name>'
==============================================
```

Launching this command by replacing the parameters generates a new command to launch and outputs:

```
...
TASK [Print version] ************************************************************
ok: [localhost] => {
    "msg": "The AOC version is 2.4.20230531-00"
}

PLAY RECAP **********************************************************************
localhost                  : ok=5    changed=0    unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
```

aws_remove_labels

This playbook removes labels from the deployment.

```
$ docker run --rm $IMAGE command_generator_vars aws_remove_labels
```

Which generates the following output:

```
==============================================
Playbook: aws_remove_labels
Description: This playbook removes labels from the deployment.
----------------------------------------------
Remove labels from the Ansible Automation Platform from AWS Marketplace deployment.

----------------------------------------------
Command generator template:

docker run --rm $IMAGE command_generator aws_remove_labels -d <deployment_name> -c
<cloud_credentials_path> --extra-vars 'aws_region=<aws_region> aws_labels=<aws_labels>'
==============================================
```

The parameter **aws_labels** is a single key to remove. For example, **key1**. To remove multiple keys, run the command multiple times.

Launching this command by replacing the parameters generates a new command to launch and outputs:

```
...
PLAY RECAP **********************************************************************
localhost                  : ok=22   changed=2    unreachable=0    failed=0    skipped=1    rescued=0
ignored=0
```

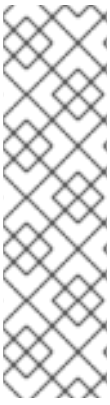
CHAPTER 8. AUTOMATION WORKLOADS

The default Ansible Automation Platform from AWS Marketplace is designed and licensed to automate 100 managed nodes.

8.1. AUTOMATION PERFORMANCE

Automating against the managed node allotment in the offer is provided with the following operational expectations. Automating outside of the boundaries of these criteria is not supported, but can still function depending on your automation.

Metric	Threshold
Concurrent Jobs	10
Forks per job	10



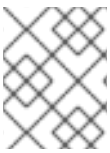
NOTE

Ansible Automation Platform from AWS Marketplace is driven using three m5.large instances, two of which run Ansible Automation Platform and automation controller and one which runs automation hub. The automation controller instances collectively support a standard workload of 100 managed active nodes. This operating criteria also supports up to 10 forks for each node. The operating criteria was set and tested using output-intensive, chatty workloads that produce 2 messages 7 seconds apart on both automation controller nodes. Workloads that are more I/O intensive might not work inside the boundaries of these conditions and can require the use of extension nodes to scale the deployment to support such automation.

8.2. DEPLOYMENT SCALING

If you want to scale your deployment beyond the initial number of supported managed nodes, Ansible Automation Platform from AWS Marketplace can be manually scaled using separately sold extension nodes. Extension nodes are additional compute instances that can be deployed to scale-up or scale-out depending on the immediate scaling requirements. If your requirements are for higher parallel automation operations you can select compute shapes that scale-up, while if you have a requirement to automate more nodes over time you can select compute shapes that scale out.

Extension nodes are the supported way of extending the capabilities of Ansible Automation Platform from AWS Marketplace.



NOTE

Red Hat does not support environments that are extended through customer design and implementation.

CHAPTER 9. BACKUP AND RESTORE

To backup and restore your Ansible Automation Platform deployment, it is vital to ensure automatic backups have been set for EFS and that these backups are accessible for restoration.

When you create an Ansible Automation Platform deployment, automatic Amazon *Elastic File System* (EFS) backups are set by default. However, it is prudent to check that those backups can be restored before a disaster scenario.

In addition, it is also extremely important that regular manual snapshots of the Amazon Relational Database Service (RDS) are taken to ensure a deployment can be restored as close as possible to its previous working state.

9.1. BACKING UP THE ANSIBLE AUTOMATION PLATFORM DEPLOYMENT

Use the following procedures to ensure a smooth backup process.

9.1.1. AWS permissions

Before starting the backup process, the AWS account used to execute a backup must have the following permissions granted.

Service/Permission	Resources
backup • backup:DescribeBackupJob • backup:ListRecoveryPointsByBackupVault • backup:StartBackupJob	
cloudformation • cloudformation:DescribeStacks • cloudformation:DescribeStackResources	
elasticfilesystem • elasticfilesystem:DescribeFileSystems	
iam • iam:PassRole	arn:aws:iam::*:`role/service-role/AWSBackupDefaultServiceRole`

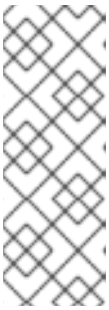
Service/Permission	Resources
rds • rds:CreateDBSnapshot • rds:DescribeDBSnapshots	
secretsmanager • secretsmanager:GetSecretValue	
s3 • s3:CreateBucket • s3:GetObject • s3:PutObject	

9.1.2. Setting the ansible-on-clouds-ops container image

The **ansible-on-clouds-ops** image tag should match the version of your foundation deployment. For example, if your foundation deployment version is 2.4.20230630-00, pull the **ansible-on-clouds-ops** image with tag 2.4.20230630.

Procedure

1. Pull the **ansible-on-clouds-ops** container image with the same tag version as the foundation deployment.



NOTE

Before pulling the docker image, ensure you are logged in to registry.redhat.io using docker. Use the following command to login to registry.redhat.io.

```
$ docker login registry.redhat.io
```

For more information about registry login, see Registry Authentication

```
$ export IMAGE=registry.redhat.io/ansible-on-clouds/ansible-on-clouds-ops-rhel9:2.4.20230630
$ docker pull $IMAGE --platform=linux/amd64
```

9.1.3. Generating the backup data file

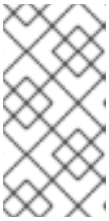
The following commands create a directory, and populate it with an empty data template that, when completed, will be used during the backup.

Procedure

1. Create a folder to hold the configuration

```
$ mkdir command_generator_data
```

2. Populate the **command_generator_data** folder with the configuration file template. This creates the **backup.yml** file within the **command-generator_data** directory.



NOTE

On Linux, any file or directory created by the command generator is owned by **root:root** by default. To change the ownership of the files and directories, you can run the **sudo chmod** command after the files are created. For more information, read Command generator - Linux files owned by root .

```
$ docker run --rm -v $(pwd)/command_generator_data:/data $IMAGE \
  command_generator_vars aws_backup_stack \
  --output-data-file /data/backup.yml
```

3. After running these commands, a **$(pwd)/command_generator_data/backup.yml** template file is created. This template file resembles the following:

```
aws_backup_stack:
  ansible_config_path:
  cloud_credentials_path:
  deployment_name:
  extra_vars:
    aws_backup_iam_role_arn:
    aws_backup_vault_name: Default
    aws_region:
    aws_s3_bucket:
    aws_ssm_bucket_name:
    backup_prefix: aoc-backup
```

9.1.4. Updating the backup data file

You must populate the data file before triggering the backup. The following variables are parameters listed in the data file.

- **ansible_config_path** (Optional) Only use if overriding with a customer **ansible_config**.
- **cloud_credentials_path** is the path to your AWS credentials file.
- **deployment_name** is the name of the foundation deployment. This is the same name you used when you deployed the foundation.
- **aws_backup_iam_role_arn** is the *Amazon Resource Name* (ARN) of the AWS IAM Role that has permissions to perform backup operations.

NOTE

You can use the AWS Backup Default Service Role for this which has the format **arn:aws:iam::<Your AWS Account Number>:role/service-role/AWSBackupDefaultServiceRole**.

- **aws_backup_vault_name** is the name of the backup vault that will hold the EFS recovery points.

NOTE

Please ensure the referenced backup vault exists and the AWS Backup IAM role provided above has the required permissions to create an EFS recovery point inside the vault.

- **aws_region** is the region where the foundation deployment is deployed.
- **aws_s3_bucket** is the name of the S3 bucket where the backup files are stored. This creates a new bucket if one does not already exist. Every backup is stored in the bucket.
  For guidance on bucket naming, see Bucket naming rules.
  For guidance on creating S3 buckets, see AWS Creating a bucket .
- **aws_ssm_bucket_name** is the name of the S3 bucket where temporary configuration files for the AWS SSM are stored. You can use an existing bucket or create a new one.

NOTE

The **aws_ssm_bucket_name** parameter is ONLY used to store temporary config files. It does NOT need to be saved for use in other playbooks. Any valid existing bucket can be used. For more information on creating S3 buckets, read AWS Creating A Bucket in the AWS documentation.

- **backup_prefix** is a prefix you would like to add to the backup name (default: aoc-backup)

When you have populated the data file, it should resemble the following. The values in this file are provided as examples:

NOTE

The optional value **ansible_config_path** has been removed. If you do not wish to use this optional value and want to use the default value for this variables, you 'must' also remove it for your data file like it was done in the example below. If you wish to use this optional variable, it 'must' be included in the data file and be assigned a value.

```
aws_backup_stack:
  cloud_credentials_path: ~/.aws/credentials
  deployment_name: AnsibleAutomationPlatform
  extra_vars:
    aws_backup_iam_role_arn: arn:aws:iam::<Your AWS Account Number>:role/service-
role/AWSBackupDefaultServiceRole
    aws_backup_vault_name: Default
    aws_region: us-east-1
```

```
aws_s3_bucket: ansible-automation-platform-bucket
aws_ssm_bucket_name: aap-ssm-bucket
backup_prefix: aoc-backup
```

9.1.5. Running the backup playbook

The following procedure runs the backup playbook as a container.

Procedure

1. To run the backup, run the command generator.

```
$ docker run --rm -v $(pwd)/command_generator_data:/data $IMAGE command_generator
aws_backup_stack --data-file /data/backup.yml
```

This generates the backup CLI command

```
-----------------------------------------------
docker run --rm --env PLATFORM=AWS -v
~/.aws/credentials:/home/runner/.aws/credentials:ro \
--env ANSIBLE_CONFIG=../aws-ansible.cfg --env
DEPLOYMENT_NAME=AnsibleAutomationPlatform --env GENERATE_INVENTORY=true  \
$IMAGE redhat.ansible_on_clouds.aws_backup_stack \
-e 'aws_foundation_stack_name=AnsibleAutomationPlatform aws_region=us-east-1
aws_backup_vault_name=Default \
aws_backup_iam_role_arn=arn:aws:iam::<Your AWS Account Number>:role/service-
role/AWSBackupDefaultServiceRole \
aws_s3_bucket=ansible-automation-platform-bucket aws_ssm_bucket_name=aap-ssm-
bucket backup_prefix=aoc-backup'
===============================================
```

2. Run the generated command to trigger the backup.

```
$ docker run --rm --env PLATFORM=AWS -v
~/.aws/credentials:/home/runner/.aws/credentials:ro \
--env ANSIBLE_CONFIG=../aws-ansible.cfg --env
DEPLOYMENT_NAME=AnsibleAutomationPlatform --env GENERATE_INVENTORY=true  \
$IMAGE redhat.ansible_on_clouds.aws_backup_stack \
-e 'aws_foundation_stack_name=AnsibleAutomationPlatform aws_region=us-east-1
aws_backup_vault_name=Default \
aws_backup_iam_role_arn=arn:aws:iam::<Your AWS Account Number>:role/service-
role/AWSBackupDefaultServiceRole \
aws_s3_bucket=ansible-automation-platform-bucket aws_ssm_bucket_name=aap-ssm-
bucket backup_prefix=aoc-backup'
```

3. The backup can take several minutes to complete, depending on the database size. A successful backup returns a log similar to the following:

```
{
    "msg": [
        "Successfully backed up AnsibleAutomationPlatform!",
        "Please note below the bucket name, region and backup name which are required for
restore process.",
        "aws_s3_bucket: ansible-automation-platform-bucket ",
```

```
        "aws_region: us-east-1",
        "aws_backup_name: ansible-automation-platform-bucket-20230706T163309",
        "Your backup files can be found at:",
        "https://s3.console.aws.amazon.com/s3/buckets/ansible-automation-platform-bucket?
region=us-east-1&prefix=aoc-backup-AnsibleAutomationPlatform-
20230706T163309/&showversions=false"
    ]
}
```

4. Your Ansible Automation Platform from AWS Marketplace deployment is now successfully backed up. As the log shows, the playbook successfully creates a backup folder in the S3 bucket specified above.

9.1.6. Deleting backups

There are two playbooks to delete backups:

- Use the **aws_backup_delete** playbook which deletes a single backup.
- Use the **aws_backups_delete** playbook which deletes multiple backups at once.

aws_backups_delete takes a array of strings ["backup1","backup2",...], while **aws_backup_delete** takes only one string, that being the name of a specific backup, "backup1".

The use of **aws_backups_delete** is desscribed in this section.

Procedure

1. Populate the **command_generator_data** directory with the configuration file template.

NOTE

On Linux, any file or directory created by the command generator is owned by **root:root** by default. To change the ownership of the files and directories, you can run the **sudo chmod** command after the files are created. For more information, read Command generator - Linux files owned by root .

```
docker run --rm -v $(pwd)/command_generator_data/:/data $IMAGE
command_generator_vars aws_backups_delete --output-data-file /data/backups_delete.yml
```

Produces the following output:

```
================================================
Playbook: aws_backups_delete
Description: This playbook delete a specified backup.
------------------------------------------------
This playbook delete a specified backup

------------------------------------------------
Command generator template:

docker run --rm -v <local_data_file_directory>:/data $IMAGE command_generator
aws_backups_delete --data-file /data/backups_delete.yml
```

2. After running the command, a **$(pwd)/command_generator_data/backups_delete.yml** template file is created. This template file resembles the following:

```
aws_backups_delete:
  cloud_credentials_path:
  extra_vars:
    aws_backup_names:
    aws_region:
    aws_s3_bucket:
    delete:
```

The **aws_backup_names** parameter must specify an array of strings, for example, **["backup1","backup2"]**. The **delete** parameter must be set to **true** to successfully delete.

1. To delete the backups, run the command generator to generate the **aws_backups_delete** command.

```
docker run --rm -v $(pwd)/command_generator_data:/data $IMAGE command_generator
aws_backups_delete --data-file /data/backups_delete.yml
```

Resulting in the following ouput:

```
Command to run playbook:

docker run --rm --env PLATFORM=AWS -v
~/.aws/credentials:/home/runner/.aws/credentials:ro \
--env ANSIBLE_CONFIG=../aws-ansible.cfg  $IMAGE
redhat.ansible_on_clouds.aws_backups_delete \
-e 'aws_region=<region> aws_s3_bucket=<bucket> aws_backup_names=
["backup1","backup2"] delete=True'
==============================================
```

2. Run the supplied backup command to delete the backups.

3. When the playbook has finished running, the output resembles the following:

```
TASK [redhat.ansible_on_clouds.standalone_aws_backup_delete : [delete_backup] Dry-run
message] ***
skipping: [localhost]

PLAY RECAP *********************************************************************
localhost                  : ok=21   changed=2    unreachable=0    failed=0    skipped=2
rescued=0    ignored=0
```

9.1.6.1. Failing to delete a backup

If the deletion of backup fails, take the following steps.

Procedure

1. Navigate to the bucket containing the backup.

2. Locate the directory that has the name of the backup.

3. Open the backup directory.
4. Open the file names **restore-vars.json** and note last part of the **Recovery_Point_ARN** and the **Backup_Vault_Name**.
5. Navigate to **AWS backup**.
6. Select **Backup Vault**
7. Select the vault with the name you noted from **Backup_Vault_Name**.
8. Search for the **Recovery_Point_ARN**.
9. Delete the **Recovery_Point_ARN** if not already deleted.
10. Navigate to the bucket containing the backup.
11. Delete the directory having the name of the backup.

9.2. RESTORING THE ANSIBLE AUTOMATION PLATFORM DEPLOYMENT

Use the following procedures to ensure a smooth restore process.

NOTE

A restored deployment contains the same VPC networking setup as the original backed up deployment. If a backed up deployment was deployed within an existing VPC, its restored deployment is also deployed into that VPC. If a backed up deployment was deployed with a new VPC, its restored deployment is also deployed with a new VPC.

9.2.1. AWS permissions

You must have the following permissions granted to restore the deployment you are planning to restore.

The following table contains a list of necessary IAM policies

For deployment with a new VPC	For deployment with an existing VPC
autoscaling • autoscaling:CreateAutoScalingGroup • autoscaling:CreateLaunchConfiguration • autoscaling:DeleteAutoScalingGroup • autoscaling:DeleteLaunchConfiguration • autoscaling:DescribeAutoScalingGroups • autoscaling:DescribeAutoScalingInstances • autoscaling:DescribeLaunchConfigurations • autoscaling:DescribeScalingActivities • autoscaling:UpdateAutoScalingGroup	autoscaling • autoscaling:CreateAutoScalingGroup • autoscaling:CreateLaunchConfiguration • autoscaling:DeleteAutoScalingGroup • autoscaling:DeleteLaunchConfiguration • autoscaling:DescribeAutoScalingGroups • autoscaling:DescribeAutoScalingInstances • autoscaling:DescribeLaunchConfigurations • autoscaling:DescribeScalingActivities • autoscaling:UpdateAutoScalingGroup
backup • backup:DescribeRestoreJob • backup:StartRestoreJob	backup • backup:DescribeRestoreJob • backup:StartRestoreJob

For deployment with a new VPC	For deployment with an existing VPC
cloudformation • cloudformation:CreateChangeSet • cloudformation:CreateStack • cloudformation:CreateUploadBucket • cloudformation:DeleteStack • cloudformation:DescribeChangeSet • cloudformation:DescribeStackEvents • cloudformation:DescribeStacks • cloudformation:ExecuteChangeSet • cloudformation:GetStackPolicy • cloudformation:GetTemplateSummary • cloudformation:ListChangeSets • cloudformation:ListStackResources • cloudformation:ListStacks • cloudformation:TagResource • cloudformation:UpdateStack • cloudformation:ValidateTemplate	cloudformation • cloudformation:CreateChangeSet • cloudformation:CreateStack • cloudformation:CreateUploadBucket • cloudformation:DeleteStack • cloudformation:DescribeChangeSet • cloudformation:DescribeStackEvents • cloudformation:DescribeStacks • cloudformation:ExecuteChangeSet • cloudformation:GetStackPolicy • cloudformation:GetTemplateSummary • cloudformation:ListChangeSets • cloudformation:ListStackResources • cloudformation:ListStacks • cloudformation:TagResource • cloudformation:UpdateStack • cloudformation:ValidateTemplate
ec2 • ec2:AllocateAddress • ec2:AssociateAddress • ec2:AssociateNatGatewayAddress • ec2:AssociateRouteTable • ec2:AssociateSubnetCidrBlock • ec2:AttachInternetGateway • ec2:AuthorizeSecurityGroupEgress • ec2:AuthorizeSecurityGroupIngress • ec2:CreateInternetGateway • ec2:CreateNatGateway • ec2:CreateRoute • ec2:CreateRouteTable	ec2 • ec2:RevokeSecurityGroupEgress • ec2:RevokeSecurityGroupIngress • ec2:DescribeKeyPairs • ec2:CreateSecurityGroup • ec2:DescribeSecurityGroups • ec2:DeleteSecurityGroup • ec2:CreateTags • ec2:AuthorizeSecurityGroupEgress • ec2:AuthorizeSecurityGroupIngress • ec2:DescribeInstances

For deployment with a new VPC	For deployment with an existing VPC
• ec2:CreateSecurityGroup • ec2:CreateSubnet • ec2:CreateSubnetCidrReservation • ec2:CreateTags • ec2:CreateVpc • ec2:DeleteInternetGateway • ec2:DeleteNatGateway • ec2:DeleteRoute • ec2:DeleteRouteTable • ec2:DeleteSecurityGroup • ec2:DeleteSubnet • ec2:DeleteSubnetCidrReservation • ec2:DeleteVpc • ec2:DescribeAccountAttributes • ec2:DescribeAddresses • ec2:DescribeAddressesAttribute • ec2:DescribeAvailabilityZones • ec2:DescribeInstances • ec2:DescribeInternetGateways • ec2:DescribeKeyPairs • ec2:DescribeNatGateways • ec2:DescribeRouteTables • ec2:DescribeSecurityGroups • ec2:DescribeSubnets • ec2:DescribeVpcs • ec2:DetachInternetGateway • ec2:DisassociateAddress • ec2:DisassociateNatGatewayAddress • ec2:DisassociateRouteTable • ec2:DisassociateSubnetCidrBlock • ec2:GetSubnetCidrReservations • ec2:ModifyAddressAttribute • ec2:ModifyVpcAttribute	

For deployment with a new VPC	For deployment with an existing VPC
• ec2:ReleaseAddress • ec2:RevokeSecurityGroupEgress • ec2:RevokeSecurityGroupIngress	
elasticfilesystem • elasticfilesystem:CreateAccessPoint • elasticfilesystem:CreateFileSystem • elasticfilesystem:CreateMountTarget • elasticfilesystem:DeleteAccessPoint • elasticfilesystem:DeleteFileSystem • elasticfilesystem:DeleteMountTarget • elasticfilesystem:DescribeAccessPoints • elasticfilesystem:DescribeBackupPolicy • elasticfilesystem:DescribeFileSystemPolicy • elasticfilesystem:DescribeFileSystems • elasticfilesystem:DescribeLifecycleConfiguration • elasticfilesystem:DescribeMountTargets	elasticfilesystem • elasticfilesystem:CreateAccessPoint • elasticfilesystem:CreateFileSystem • elasticfilesystem:CreateMountTarget • elasticfilesystem:DeleteAccessPoint • elasticfilesystem:DeleteFileSystem • elasticfilesystem:DeleteMountTarget • elasticfilesystem:DescribeAccessPoints • elasticfilesystem:DescribeBackupPolicy • elasticfilesystem:DescribeFileSystemPolicy • elasticfilesystem:DescribeFileSystems • elasticfilesystem:DescribeLifecycleConfiguration • elasticfilesystem:DescribeMountTargets
elasticloadbalancing • elasticloadbalancing:AddTags • elasticloadbalancing:CreateListener • elasticloadbalancing:CreateLoadBalancer • elasticloadbalancing:CreateTargetGroup • elasticloadbalancing:DeleteListener • elasticloadbalancing:DeleteTargetGroup • elasticloadbalancing:DescribeListeners • elasticloadbalancing:DescribeTargetGroups • elasticloadbalancing:ModifyLoadBalancerAttributes • elasticloadbalancing:ModifyTargetGroupAttributes	elasticloadbalancing • elasticloadbalancing:AddTags • elasticloadbalancing:CreateListener • elasticloadbalancing:CreateLoadBalancer • elasticloadbalancing:CreateTargetGroup • elasticloadbalancing:DeleteListener • elasticloadbalancing:DeleteTargetGroup • elasticloadbalancing:DescribeListeners • elasticloadbalancing:DescribeTargetGroups • elasticloadbalancing:ModifyLoadBalancerAttributes • elasticloadbalancing:ModifyTargetGroupAttributes

For deployment with a new VPC	For deployment with an existing VPC
iam • iam:AddRoleToInstanceProfile • iam:AttachRolePolicy • iam:CreateInstanceProfile • iam:CreateRole • iam:DeleteInstanceProfile • iam:DeleteRole • iam:DeleteRolePolicy • iam:DetachRolePolicy • iam:GetRolePolicy • iam:ListRoles • iam:PassRole • iam:PutRolePolicy • iam:RemoveRoleFromInstanceProfile • iam:TagRole	iam • iam:AddRoleToInstanceProfile • iam:AttachRolePolicy • iam:CreateInstanceProfile • iam:CreateRole • iam:DeleteInstanceProfile • iam:DeleteRole • iam:DeleteRolePolicy • iam:DetachRolePolicy • iam:GetRolePolicy • iam:ListRoles • iam:PassRole • iam:PutRolePolicy • iam:RemoveRoleFromInstanceProfile • iam:TagRole
kms • kms:CreateGrant • kms:Decrypt • kms:DescribeKey • kms:GenerateDataKey	kms • kms:CreateGrant • kms:Decrypt • kms:DescribeKey • kms:GenerateDataKey

For deployment with a new VPC	For deployment with an existing VPC
rds • rds:AddTagsToResource • rds:CreateDBInstance • rds:CreateDBSubnetGroup • rds:DeleteDBInstance • rds:DeleteDBSubnetGroup • rds:DescribeDBInstances • rds:DescribeDBSnapshots • rds:DescribeDBSubnetGroups • rds:ModifyDBInstance • rds:RestoreDBInstanceFromDBSnapshot	rds • rds:AddTagsToResource • rds:CreateDBInstance • rds:CreateDBSubnetGroup • rds:DeleteDBInstance • rds:DeleteDBSubnetGroup • rds:DescribeDBInstances • rds:DescribeDBSnapshots • rds:DescribeDBSubnetGroups • rds:ModifyDBInstance • rds:RestoreDBInstanceFromDBSnapshot
s3 • s3:CreateBucket • s3:GetObject • s3:PutObject	s3 • s3:CreateBucket • s3:GetObject • s3:PutObject
secretsmanager • secretsmanager:CreateSecret • secretsmanager:DeleteSecret • secretsmanager:GetRandomPassword • secretsmanager:GetSecretValue • secretsmanager:PutSecretValue • secretsmanager:TagResource	secretsmanager • secretsmanager:CreateSecret • secretsmanager:DeleteSecret • secretsmanager:GetRandomPassword • secretsmanager:GetSecretValue • secretsmanager:PutSecretValue • secretsmanager:TagResource
sns • sns:ListTopics	sns • sns:ListTopics

9.2.2. Setting the ansible-on-clouds-ops container image

The **ansible-on-clouds-ops** image tag should match the version of your foundation deployment. For example, if your foundation deployment version is 2.4.20230630-00, pull the **ansible-on-clouds-ops** image with tag 2.4.20230630.

Procedure

1. Pull the **ansible-on-clouds-ops** container image with the same tag version as the foundation deployment.

NOTE

Before pulling the docker image, ensure you are logged in to registry.redhat.io using docker. Use the following command to login to registry.redhat.io.

```
$ docker login registry.redhat.io
```

For more information about registry login, see Registry Authentication

```
$ export IMAGE=registry.redhat.io/ansible-on-clouds/ansible-on-clouds-ops-rhel9:2.4.20230630
$ docker pull $IMAGE --platform=linux/amd64
```

9.2.3. Generating the restore data file

The following commands create a directory, and populate it with an empty data template that, when completed, is used during the restore.

Procedure

1. Create a folder to hold the configuration

```
$ mkdir command_generator_data
```

2. Populate the **$(pwd)/command_generator_data** folder with the configuration file template.

NOTE

On Linux, any file or directory created by the command generator is owned by **root:root** by default. To change the ownership of the files and directories, you can run the **sudo chmod** command after the files are created. For more information, read Command generator - Linux files owned by root .

```
$ docker run --rm -v $(pwd)/command_generator_data:/data $IMAGE \
  command_generator_vars aws_restore_stack \
  --output-data-file /data/restore.yml
```

3. After running these commands, a **$(pwd)/command_generator_data/restore.yml** template file is created. This template file resembles the following:

```
aws_restore_stack:
  ansible_config_path:
  cloud_credentials_path:
  deployment_name:
  extra_vars:
    aws_backup_iam_role_arn:
    aws_backup_name:
```

```
aws_backup_restore_point_arn:
aws_backup_vault_name:
aws_rds_db_snapshot_arn:
aws_region:
aws_s3_bucket:
aws_ssm_bucket_name:
```

9.2.4. Updating the restore data file

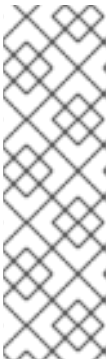
You must populate the data file before triggering the restore. The following variables are parameters listed in the data file.

- **ansible_config_path** (Optional) Only use if overriding with a custom **ansible_config**.
- **cloud_credentials_path** is the path to your AWS credentials file.
- **deployment_name** is the name you want for your restored deployment
- **aws_backup_iam_role_arn** (Optional) is the *Amazon Resource Name* (ARN) of the AWS IAM Role that has permissions to perform backup operations.

NOTE

If provided, the playbook defaults to this value over the value of the IAM role referenced in the backup file on S3. For further information, see Backing up the Ansible Automation Platform deployment

- **aws_backup_name** is the name of the backup folder stored on S3. This value was outputted after running the backup playbook.
- **aws_backup_restore_point_arn** (Optional) is the ARN of the of the EFS recovery point you want to use for restore.



NOTE

If provided, the playbook defaults to this value over the EFS restore point referenced in the backup file on S3. See AWS permissions.

You must also ensure that the provided restore point is from a version of Ansible on Clouds (AoC) that matches the version of the ansible-on-clouds-ops container image running the restore. For example, a restore point taken from a deployment on AoC version 2.4.20230630-00 needs to use the **ansible-on-clouds-ops** container image version **2.4.20230630**.

- **aws_backup_vault_name** (Optional) is the name of the backup vault that holds the EFS recovery points.

NOTE

If provided, the playbook defaults to this value over the backup vault referenced in the backup file on S3 which was generated when running the backup operation. See AWS permissions. You must also ensure that this backup vault matches the one used to backup the deployment.

- **aws_rds_db_snapshot_arn** (Optional) is the ARN of the RDS snapshot you want to use for restore.

NOTE

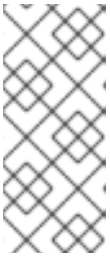
If provided, the playbook defaults to this value over the RDS snapshot referenced in the backup file on S3. You must also ensure that the provided RDS snapshot is from a version of Ansible on Clouds that matches the version of the **ansible-on-clouds-ops** container image running the restore. For example, an RDS snapshot taken from a deployment on AoC version 2.4.20230630-00 needs to use the **ansible-on-clouds-ops** container image version **2.4.20230630**.

- **aws_region** is the region where the foundation deployment is deployed.
- **aws_s3_bucket** is the name of the S3 bucket where the backup files are stored. This must be the same bucket used for backup.
- **aws_ssm_bucket_name** is the name of the S3 bucket where temporary configuration files for the AWS SSM are stored. You can use an existing bucket or create a new one.

NOTE

The **aws_ssm_bucket_name** parameter is ONLY used to store temporary config files. It does NOT need to be saved for use in other playbooks. Any valid existing bucket can be used. For more information on creating S3 buckets, read AWS Creating A Bucket in the AWS documentation.

When you have populated the data file, it should resemble the following. The values in this file are provided as examples.



NOTE

The optional values in this data file example have been removed. If you do not wish to use these optional values and want to use the default values for these variables, you 'must' also remove them for your data file like it was done in the example below. If you wish to use these optional variables, they 'must' be included in the data file and be assigned a value.

```
aws_restore_stack:
  cloud_credentials_path: ~/.aws/credentials
  deployment_name: AnsibleAutomationPlatform
  extra_vars:
    aws_backup_name: ansible-automation-platform-bucket-20230706T163309
    aws_region: us-east-1
    aws_s3_bucket: ansible-automation-platform-bucket
    aws_ssm_bucket_name: aap-ssm-bucket
```

9.2.5. Running the restore playbook

The following procedure runs the restore playbook as a container.

Procedure

1. To run the restore, run the command generator.

```
$ docker run --rm -v $(pwd)/command_generator_data:/data $IMAGE command_generator
aws_restore_stack --data-file /data/restore.yml
```

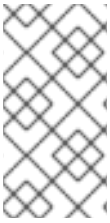
This generates the restore CLI command

```
-----------------------------------------------
docker run --rm --env PLATFORM=AWS -v
~/.aws/credentials:/home/runner/.aws/credentials:ro --env ANSIBLE_CONFIG=../aws-
ansible.cfg \
--env DEPLOYMENT_NAME=AnsibleAutomationPlatform --env
GENERATE_INVENTORY=true --env CHECK_GENERATED_INVENTORY=false  \
$IMAGE redhat.ansible_on_clouds.aws_restore_stack -e
'aws_foundation_stack_name=AnsibleAutomationPlatform \
aws_backup_name=ansible-automation-platform-bucket-20230706T163309 aws_region=us-
east-1 \
aws_s3_bucket=ansible-automation-platform-bucket aws_ssm_bucket_name=aap-ssm-
bucket'
===============================================
```

2. Run the generated command to trigger the restore.

```
$ docker run --rm --env PLATFORM=AWS -v
~/.aws/credentials:/home/runner/.aws/credentials:ro --env ANSIBLE_CONFIG=../aws-
ansible.cfg \
--env DEPLOYMENT_NAME=AnsibleAutomationPlatform --env
GENERATE_INVENTORY=true --env CHECK_GENERATED_INVENTORY=false  \
$IMAGE redhat.ansible_on_clouds.aws_restore_stack -e
'aws_foundation_stack_name=AnsibleAutomationPlatform \
aws_backup_name=ansible-automation-platform-bucket-20230706T163309 aws_region=us-
east-1 \
aws_s3_bucket=ansible-automation-platform-bucket aws_ssm_bucket_name=aap-ssm-
bucket'
```

3. The playbook can take some time to create a restored deployment. When you have run the playbook successfully, a restored deployment is available in AWS CloudFormation matching the name provided for it in the **restore.yml** file.



NOTE

A restored deployment contains the same VPC networking setup as the original backed up deployment. If a backed up deployment was deployed within an existing VPC, its restored deployment is also deployed into that VPC. If a backed up deployment was deployed with a new VPC, its restored deployment is also deployed with a new VPC.

CHAPTER 10. UPGRADING

You can upgrade your existing Ansible Automation Platform deployment to the newer version. The upgrade process covers upgrading the automation hub, automation controller, and extension nodes. The upgrade process takes roughly the same amount of time as a Ansible Automation Platform deployment install. You are required to create a backup before running the upgrade.

NOTE

Ansible Automation Platform from AWS Marketplace supports sequential upgrades. All upgrades should be no more than one major version behind the version you are currently upgrading to. For example, to upgrade Ansible Automation Platform to 2.4.20230630, you must be on version 2.3.20230221.

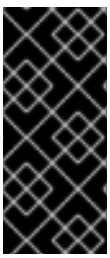
Prerequisites

- Docker must be installed to run the upgrade playbook.
- A Linux or macOS system, where the **ansible-on-clouds-ops** container image will run.
- The upgrade process requires several volumes to be mounted. Prepare a fresh directory to be used for this process.

The following procedures form the upgrade process:

1. Backup your Ansible Automation Platform 2.3 stack
   - Follow the Ansible on Clouds 2.3 backup instructions
2. Upgrade Ansible Automation Platform
   a. Pull the ansible-on-clouds-ops 2.4 container image
   b. Ensure minimum required permissions are met
   c. Generate the data file
   d. Update the data file
   e. Begin upgrading Ansible Automation Platform by running the operational container
3. (Optional) Restore the stack from a backup.
   - Follow the Ansible on Clouds 2.3 restore instructions.

10.1. BACKUP BEFORE UPGRADE



IMPORTANT

Before beginning the upgrade of your Ansible Automation Platform environment, it is first required to take a backup of your environment at its current version.

To backup your Ansible Automation Platform environment at the earlier version, follow the Ansible on Clouds 2.3 backup instructions .

10.2. UPGRADE ANSIBLE AUTOMATION PLATFORM

10.2.1. Pulling the ansible-on-clouds-ops container image

Procedure

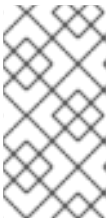
- Pull the Docker image for Ansible on Clouds operational container with the same tag as the version you want to upgrade to.

NOTE

Before pulling the docker image, ensure you are logged in to registry.redhat.io using docker. Use the following command to login to registry.redhat.io.

```
$ docker login registry.redhat.io
```

For more information about registry login, see Registry Authentication



NOTE

The Ansible on Clouds operational image tag must match the version that you want to upgrade to. For example, if your foundation deployment version is 2.3.20230221, pull the operational image with tag 2.4.20230630 to upgrade to version 2.4.20230630.

```
$ export IMAGE=registry.redhat.io/ansible-on-clouds/ansible-on-clouds-ops-
rhel9:2.4.20230630
$ docker pull $IMAGE --platform=linux/amd64
```

10.2.2. Required permissions

You must have the following AWS IAM permissions to upgrade the stack:

```
required-roles:
  ec2:
    actions:
      - ec2:DescribeInstances
      - ec2:GetLaunchTemplateData
      - ec2:DescribeAccountAttributes
      - ec2:DescribeLaunchTemplates
      - ec2:CreateTags
      - ec2:RunInstances
      - ec2:DescribeInstanceTypes
      - ec2:DescribeSubnets
      - ec2:DeleteTags
      - ec2:DescribeRegions
      - ec2:DescribeAvailabilityZones
      - ec2:DeleteLaunchTemplate
      - ec2:DescribeTags
      - ec2:DescribeLaunchTemplateVersions
      - ec2:DescribeSecurityGroups
      - ec2:CreateLaunchTemplateVersion
```

```
    - ec2:CreateLaunchTemplate
    - ec2:DescribeVpcs
    - ec2:ModifyLaunchTemplate
  resources:
    - "*"
iam:
  actions:
    - iam:ListRoleTags
    - iam:AttachRolePolicy
    - iam:PutRolePolicy
    - iam:AddRoleToInstanceProfile
    - iam:ListAttachedRolePolicies
    - iam:GetRole
    - iam:GetRolePolicy
    - iam:CreateInstanceProfile
    - iam:ListInstanceProfilesForRole
    - iam:PassRole
    - iam:GetInstanceProfile
    - iam:CreateServiceLinkedRole
  resources:
    - "*"
secretsmanager:
  actions:
    - secretsmanager:DescribeSecret
    - secretsmanager:ListSecrets
    - secretsmanager:TagResource
    - secretsmanager:UntagResource
    - secretsmanager:CreateSecret
    - secretsmanager:GetSecretValue
    - secretsmanager:UpdateSecret
    - secretsmanager:GetResourcePolicy
  resources:
    - "*"
ssm:
  actions:
    - ssm:StartSession
    - ssm:TerminateSession
  resources:
    - "*"
autoscaling:
  actions:
    - autoscaling:DescribeAutoScalingGroups
    - autoscaling:UpdateAutoScalingGroup
    - autoscaling:DescribeInstanceRefreshes
    - autoscaling:DeleteTags
    - autoscaling:DescribeTags
    - autoscaling:CreateOrUpdateTags
    - autoscaling:CreateAutoScalingGroup
    - autoscaling:DescribeLaunchConfigurations
    - autoscaling:DescribeLifecycleHooks
    - autoscaling:StartInstanceRefresh
    - autoscaling:CreateLaunchConfiguration
    - autoscaling:DisableMetricsCollection
    - autoscaling:DeleteLaunchConfiguration
    - autoscaling:DetachLoadBalancerTargetGroups
    - autoscaling:AttachLoadBalancerTargetGroups
```

```
  resources:
    - "*"
s3:
  actions:
    - s3:DeleteObject
    - s3:PutObject
    - s3:GetObject
    - s3:GetBucketLocation
  resources:
    - "*"
cloudformation:
  actions:
    - cloudformation:ListStackResources
    - cloudformation:DescribeStacks
    - cloudformation:GetTemplate
  resources:
    - "*"
elasticloadbalancing:
  actions:
    - elasticloadbalancing:DescribeTargetGroupAttributes
    - elasticloadbalancing:DescribeTags
    - elasticloadbalancing:DescribeTargetHealth
    - elasticloadbalancing:DescribeTargetGroups
    - elasticloadbalancing:ModifyTargetGroup
    - elasticloadbalancing:DescribeLoadBalancers
    - elasticloadbalancing:DescribeLoadBalancerAttributes
    - elasticloadbalancing:DescribeListeners
    - elasticloadbalancing:CreateListener
    - elasticloadbalancing:DeleteListener
    - elasticloadbalancing:DescribeRules
    - elasticloadbalancing:DeleteTargetGroup
    - elasticloadbalancing:CreateTargetGroup
    - elasticloadbalancing:ModifyTargetGroupAttributes
    - elasticloadbalancing:AddTags
    - elasticloadbalancing:RemoveTags
  resources:
    - "*"
```

10.2.3. Generating the data file

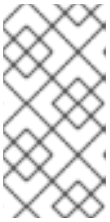
The commands in this section create a directory and populate it with an empty data template that, when populated, is used during the upgrade.

Procedure

1. Run the following commands to generate the required data file.

```
# Create a folder to hold the configuration files
$ mkdir command_generator_data
```

2. Populate the **command_generator_data** folder with the configuration file template.



NOTE

On Linux, any file or directory created by the command generator is owned by **root:root** by default. To change the ownership of the files and directories, you can run the **sudo chmod** command after the files are created. For more information, read Command generator - Linux files owned by root .

```
$ docker run --rm -v $(pwd)/command_generator_data:/data $IMAGE \
command_generator_vars aws_upgrade \
--output-data-file /data/extra_vars.yml
```

3. After running these commands, a **command_generator_data/extra_vars.yml** template file is created. This template file resembles the following:

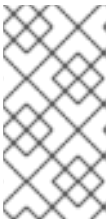
```
aws_upgrade:
  ansible_config_path:
  cloud_credentials_path:
  deployment_name:
  extra_vars:
    aws_backup_taken:
    aws_region:
    aws_ssm_bucket_name:
    seller_name:
```

10.2.4. Updating the data file

You must populate the data file before triggering the upgrade. Each parameter is required unless it is noted as optional. If the parameter is noted as optional, both its key and value can be removed from the data file.

The following variables are parameters listed in the data file:

- **ansible_config_path** (Optional) Only use if overriding with a custom **ansible_config**.
- **cloud_credentials_path** is the path to your AWS credentials file.
- **deployment_name** is the name of the foundation deployment.
  - This is the same name you used when you deployed the foundation.
- **aws_backup_taken** is the verification that a manual backup of the current deployment was recently created prior to running this upgrade.
- **aws_region** is the AWS region where the foundation deployment is located.
- **aws_ssm_bucket_name** is the name of the S3 bucket where temporary configuration files for the AWS SSM are stored. You can use an existing bucket or create a new one.



NOTE

The **aws_ssm_bucket_name** parameter is ONLY used to store temporary config files. It does NOT need to be saved for use in other playbooks. Any valid existing bucket can be used. For more information on creating S3 buckets, read AWS Creating A Bucket in the AWS documentation.

The bucket name must not contain uppper case letters.

- **seller_name** (Optional) is used to designate the AWS Marketplace Seller. The default value is **redhatinc**. If you are based in EMEA and purchased this offering through the **redhatlimited** AWS Marketplace seller, ensure that this value is set to **redhatlimited**.

After populating the data file, it should resemble the following.

The following values are provided as examples:

```
aws_upgrade:
  ansible_config_path:
  cloud_credentials_path: ~/.aws/credentials
  deployment_name: AnsibleAutomationPlatform
  extra_vars:
    aws_backup_taken: true
    aws_region: us-east-1
    aws_ssm_bucket_name: aap-ssm-bucket
    seller_name: redhatinc
```

10.2.5. Running the upgrade playbook

NOTE

Ansible Automation Platform upgrade to 2.4.20230630 updates the listeners on its internal load balancers. If a resource dependancy was added after installation, this must be temporarily removed to allow the upgrade to succeed. See technical note for more detail.

1. To run the upgrade, run the command generator to generate the upgrade CLI command:

```
$ docker run --rm -v $(pwd)/command_generator_data:/data $IMAGE command_generator --
data-file /data/extra_vars.yml
```

This generates the following upgrade command:

```
------------------------------------------------
docker run --rm --env PLATFORM=AWS -v
~/.aws/credentials:/home/runner/.aws/credentials:ro --env ANSIBLE_CONFIG=../aws-
ansible.cfg --env DEPLOYMENT_NAME=AnsibleAutomationPlatform --env
GENERATE_INVENTORY=true  $IMAGE redhat.ansible_on_clouds.aws_upgrade -e
'aws_foundation_stack_name=AnsibleAutomationPlatform aws_region=us-east-1
aws_ssm_bucket_name=aap-ssm-bucket aws_backup_taken=True seller_name=redhatinc'
================================================
```

2. Run the given upgrade command to trigger the upgrade.

```
$ docker run --rm --env PLATFORM=AWS -v
~/.aws/credentials:/home/runner/.aws/credentials:ro --env ANSIBLE_CONFIG=../aws-
ansible.cfg --env DEPLOYMENT_NAME=AnsibleAutomationPlatform --env
GENERATE_INVENTORY=true  $IMAGE redhat.ansible_on_clouds.aws_upgrade -e
'aws_foundation_stack_name=AnsibleAutomationPlatform aws_region=us-east-1
aws_ssm_bucket_name=aap-ssm-bucket aws_backup_taken=True seller_name=redhatinc'
```

3. The upgrade can take some time to complete, but can take longer depending on the number of extension nodes on the system. A successful upgrade is marked by the log below.

```
TASK [redhat.ansible_on_clouds.standalone_aws_upgrade : [upgrade] [LOG] upgrade
version] ***
ok: [localhost] => {
    "msg": "Successfully upgraded from '2.3.20230221-00' -> '2.4.20230630-00'."
}
```

4. Your Ansible Automation Platform from AWS Marketplace deployment is now upgraded to a newer version and you can log in to Red Hat Ansible Automation Platform automation controller and automation hub using your deployment credentials.

10.3. RESTORE FROM BACKUP

If you would like to restore the earlier version of your Ansible Automation Platform environment to the state it was in before the upgrade, follow the Ansible on Clouds 2.3 restore instructions.

CHAPTER 11. UNINSTALLING

This chapter describes how to uninstall Ansible Automation Platform.

11.1. REMOVING EXTENSION NODES

If you have deployed extension nodes, you must take the following steps to remove the extension nodes before uninstalling the stack.

If you have added extension nodes, follow the procedures in Removing extension nodes to remove them.

11.2. UNINSTALLING ANSIBLE AUTOMATION PLATFORM

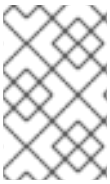
Procedure

1. In the AWS UI, navigate to the cloudformation page. Ensure you are in the correct region.
2. Select the stack for your Ansible Automation Platform deployment.
3. Click **Delete**.
4. Click **Delete** to confirm the deletion.

It can take some time to complete the deletion.

11.3. REMOVING UPGRADE RESOURCES

The following procedures describes how to delete any resources left behind after an upgrade.



NOTE

These actions cannot be reversed.

You must only perform these actions if your deployment has been deleted.

11.3.1. Removing the launch configurations

Use the following procedure to remove the old launch configurations.

Procedure

1. In the AWS UI, navigate to the Launch configurations page. Ensure you are in the same region as your stack.
    a. Click **View launch configurations** in the popup that appears.
2. Identify the **Launch Configuration** corresponding to your Ansible Automation Platform deployment.
    a. Search for your deployment name. These are named **<deployment_name>-hubautoscalegroup-<sufix>** for the hub autoscalegroup launch configuration and **<deployment_name>-controllerautoscalegroup-<sufix>** for the controller autoscalegroup launch configuration.

3. Delete the resources.
    a. Select the checkbox next to the Launch Configurations.
    b. Click **Actions**.
    c. Click **Delete launch configurations**.
    d. Confirm the deletion.

11.3.2. Removing the hub IAM role

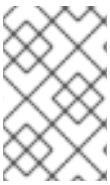
Use the following procedure to remove the hub IAM role. This step is necessary if you are uninstalling your Ansible Automation Platform environment after an upgrade and your CloudFormation stack has a status of **DELETE_FAILED**

Procedure

1. In the AWS UI, navigate to the CloudFormation page. Ensure you are in the same region as your stack.
2. Select the stack for your Ansible Automation Platform deployment.
3. Click **Delete**.
4. If a popup appears listing your hub IAM role as failing to delete, click the link below the hub IAM role open the role in a tab.
5. Delete the hub role.
    a. Click **Delete**.
    b. Confirm the deletion.
    c. Ensure the deletion is successful.
6. Return to your CloudFormation page in the previous tab.
    a. Ensure the checkbox next to the hub IAM role is unchecked.
    b. Click **Delete** to confirm the stack deletion.
    c. Confirm the deletion.

11.4. REMOVING BACKUP RESOURCES

The following procedures describe how to delete any resources left behind when uninstalling your Ansible Automation Platform deployment, after having run a backup on it.



NOTE

These actions cannot be reversed.

You must only perform these actions if your deployment has been deleted.

11.4.1. Removing the S3 bucket

The following procedure removes the S3 bucket used for backup.

Procedure

1. In the AWS UI, navigate to the S3 Bucket page. Ensure you have selected the same region as your stack.
2. Identify the S3 Bucket associated with your backup.
    a. Search for the bucket name. The bucket name must be the same the name used during the backup procedure.
3. Delete the bucket.
    a. Select the radio button next to the bucket name.
    b. Click **Empty**.
    c. Confirm all objects have been deleted.
    d. Select the radio button next to the bucket name.
    e. Click **Delete**.
    f. Confirm the deletion.

11.4.2. Removing AWS backup recovery points

The following procedure deletes AWS backup files left after backup.

Procedure

1. In the AWS UI, navigate to the AWS Backup page. Ensure you have selected the same region as your stack.
2. Navigate to the Backup Vaults in the sidebar.
3. Identify the Backup Vault associated with your backup.
4. Search for the vault name. This must be the same as the name used during the backup procedure.
5. Delete the resource.
    a. Select the checkboxes of all recovery points you want to delete.
    b. Click **Actions**, then click **Delete**.
    c. Confirm the deletion.

CHAPTER 12. TECHNICAL NOTES

Ansible Automation Platform from AWS Marketplace is a self-managed deployment. The following are technical notes regarding the Ansible Automation Platform from AWS Marketplace deployment.

12.1. ADD AND REMOVE TAGS

Tags are custom fields that you assign to an AWS resource to help organize and identify your instances. Ansible Automation Platform from AWS Marketplace supports stack-level tagging during stack deployment time. A stack-level tag, when applied to the stack itself, is also applied to all tag supported components that make up the stack.

Using the add or remove labels playbooks only updates the tags for the individual components of a stack and not the tag applied to the stack itself. Stack-level tags are never modified after deployment.

The **aws_remove_labels** playbook only supports removing a single AWS tag at a time. The parameter **aws_labels** is a single key to remove. To remove multiple keys, run the command multiple times.

12.2. ANSIBLE ON CLOUDS OPS CONTAINER

Having a deployment with the same name in multiple regions will break the functionality of the add extension node, remove extension node and backup stack playbooks found in the Ansible on Clouds Ops Container. To ensure these playbooks function as expected, please ensure that Ansible Automation Platform from AWS Marketplace deployments are uniquely named across all regions.

12.3. BACKUP AND RESTORE

1. The **aws_rds_db_snapshot_arn** optional parameter which allowed customers to specify a specific AWS RDS Snapshot to use for restoring their deployment is no longer supported. The restore process no longer uses AWS RDS Snapshots to restore a deployment and instead relies on the backup files generated by the Ansible on Clouds Ops Container backup playbook. The Ansible on Clouds Ops Container restore playbook does not use any provided RDS snapshot passed as a parameter.

2. A restored Ansible Automation Platform from AWS Marketplace deployment does not maintain all secrets from the original Ansible Automation Platform from AWS Marketplace deployment. If using your own certificates, they must be reset after a restore.

   Secrets that are maintained:

   - **<stack_name>-aap-admin-secret**
   - **<stack_name>-aap-secret-key**

   Secrets that are NOT maintained:

   - **<stack_name>-aap-rds-secret**
   - **<stack_name>-database_fields_symmetric_key**
   - **<stack_name>-container_auth_private_key_pem**
   - **<stack_name>-container_auth_public_key_pem**

- **<stack_name>-pulp_cert**
- **<stack_name>-pulp_key**

3. The Ansible on Clouds Ops Container restore playbook fails when attempting to restore a secret starting with **-**. If the restore operation is failing, on your deployment please ensure the secret **<stack_name>-database_fields_symmetric_key** does not start with a **-**. Follow the steps below to check for this case and update the secret appropriately.

   a. Find the backup you are passing to the restore playbook on S3.

      i. The **aws_backup_name** parameter you passed to the restore playbook is the backup we are trying to locate and it should be a folder inside of an S3 bucket

      ii. The **aws_s3_bucket** parameter you passed to the restore playbook should be the bucket which contains the backup we are trying to locate

   b. After locating the backup, navigate inside of the backup folder and download the **secrets.json** file.

   c. In your downloaded **secrets.json** file, find the value of **<stack_name>-database_fields_symmetric_key**. If **<stack_name>-database_fields_symmetric_key** does not begin with a **-**, you do not need to do anything else; your restore operation is not failing due to this issue. If **<stack_name>-database_fields_symmetric_key** does begin with a **-**, edit the downloaded **secrets.json** file and replace the **-** with any alphanumeric character.

   d. After replacing the **-** with an alphanumeric character, navigate back to backup folder in the S3 console and click **Upload**.

   e. Drag and drop the updated **secrets.json** file into the file upload box on the S3 console webpage and click **Upload**.

   f. You have successfully updated the **secrets.json** file for a backup. Using this backup for a restore operation should now allow you to successfully create a restored deployment.

4. Cross-region backup and restore operations are not supported in this release. This means that a backed up deployment in one region can only be restored in to that same region.

5. If a deployment is created inside of an existing VPC, this VPC must exist for backup and restore operations to function. A backed up deployment cannot be restored if the VPC containing this deployment is deleted.

12.4. ADD AND REMOVE EXTENSION NODES

1. On rare occasions, the Anisble on Clouds Ops Container remove extension node playbook can fail with the message **Waited too long for old instances to terminate**. Run the playbook again to resolve the issue.

2. When removing extension nodes using the Ansible-on-Clouds ops playbook, ensure you have provided the correct autoscaling group name and launch template name. Providing incorrect autoscaling group and launch template names results in some orphaned resources.

12.5. COMMAND GENERATOR - LINUX FILES OWNED BY ROOT

On Linux, any file or directory created by the command generator is owned by **root:root** by default. To change the ownership of the files and directories, you can run the **sudo chmod** command after the files are created:

```
# Change the owner of the command_generator_data directory recursively
$ sudo chown -R $USER:$USER command_generator_data/

# Check the permissions
$ ls -la command_generator_data/
```

The command generator currently expects to use the Docker CLI in the daemon mode. The default Docker installation does not have User namespace mapping for Discretionary access control (DAC). So, for any file created by **root** from the container will also be owned by **root** on the host if the file is located in a shared volume.

You can learn more about the Linux namespaces, including the User namespace at the article The 7 most used Linux namespaces.

12.6. UPGRADE NOTE

Upgrading Ansible Automation Platform to 2.4.20230630 updates the protocol of the listeners attached to the internal load balancers from HTTP to HTTPs. To update the listener protocol, the listener must be deleted and recreated properly. For this operation to succeed there must not have been an added resource dependancy on the listener. If there is a resource dependancy on the listener, this halts the upgrade, and the dependancy must be manually removed for the upgrade to succeed. When the upgrade has suceeded, the dependant resource can be recreated. You must revalidate any additional added networking configurations after upgrade.

12.7. ANSIBLE AUTOMATION PLATFORM CONTROLLER API

API endpoints for Controller must contain a trailing slash in order for requests to go through. Automatic trailing slash redirects are not supported in this current offering of Ansible Automation Platform from AWS Marketplace. For example a request such as **<controller_base_url>/api/v2/metrics** times out while **<controller_base_url>/api/v2/metrics/** goes through.

12.8. COMMAND GENERATOR - AWS CREDENTIALS FILE

The command generator only supports the AWS credentials file with a **default** profile name.

Example of a supported AWS credentials file:

```
[default]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

Example of an unsupported AWS credentials file:

```
[something_else]
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

CHAPTER 13. SUPPORT

Ansible Automation Platform from AWS Marketplace is a self-managed deployment. When the application is deployed into your AWS infrastructure, customers are responsible for the maintenance of the AWS infrastructure, OS patching, and Ansible Automation Platform patching.

Red Hat does not support changes to the infrastructure resources deployed as part of the solution unless there is documentation by Red Hat describing the process, such as in route table configuration for networking or documented upgrade processes. Adding additional compute resources outside of extension nodes voids Red Hat support for Ansible Automation Platform from AWS Marketplace deployment.

Upgrades to Ansible Automation Platform from AWS Marketplace are performed differently from self-installed Ansible Automation Platform. Upgrading individual packages on virtual machines using **dnf** or other means is also not supported. Instructions will be provided in the upgrade section of this documentation with relevant content for each version, as they become available.

13.1. SUPPORTED INFRASTRUCTURE CONFIGURATION CHANGES

Red Hat supports changes to the following options:

- VPC Route Table configuration
- VPC Security Group configuration
- VPC Load Balancer configuration
- EC2 Block Storage expansion
- DNS and **resolv.conf** files

Red Hat responsibilities

Red Hat has the following responsibilities:

- Premium support for Ansible Automation Platform
- Directions and how-to processes for upgrading Ansible Automation Platform from AWS Marketplace
- Red Hat supports the current version of Ansible Automation Platform. Bug fixes and CVE patches require an upgrade to the latest version

Customer responsibilities

You have the following responsibilities:

- AWS infrastructure uptime
- AWS infrastructure changes, for example, increasing block storage sizes
- AWS network peering and configuration
- Application of Ansible Automation Platform upgrades
    - Operating system upgrades are included

- Backing up AWS resources

APPENDIX A. RELEASE NOTES FOR ANSIBLE ON CLOUDS 2.4

This release includes a number of enhancements, additions, and fixes that have been implemented for Ansible Automation Platform from AWS Marketplace.

NOTE

Automation Mesh and Event Driven Automation are not yet supported by the self-managed Ansible on Cloud offerings at this time.

Enhancements

The release version 2.4.20230630 of Ansible Automation Platform from AWS Marketplace includes the following enhancements:

- Support for Ansible Automation Platform 2.4 has been added.
- Added internal encryption to the web servers
  - Addition of custom certificates is now supported.
- Support has been added for custom tagging and labelling.
  - Support has been added to AWS to add or remove tag support for resources owned by the deployment.
- There are improvements to backup and restore functionality.
  - Support has been added for taking multiple backups.
  - An operational playbook has been added to list available backups.
  - An operational playbook has been added to delete a selected backup.
  - The capability to restore an existing VPC has been added.
- Operational functionality to check the current version has been added.
  - Operational playbooks now check to ensure that the Ansible Automation Platform environment they are operating upon is at the same version before beginning their operation.

Deprecated and removed features

Some features available in previous releases have been deprecated or removed. Deprecated functionality is still included in Ansible Automation Platform and continues to be supported; however, it will be removed in a future release of this product and is not recommended for new deployments. The following is a list of major functionality deprecated and removed within Ansible Automation Platform 2.3:

- On-premise component Automation Services Catalog is now removed from Ansible Automation Platform 2.4 onwards.
- With the Ansible Automation Platform 2.4 release, the Execution Environment container image for Ansible 2.9 (ee-29-rhel-8) is no longer loaded into the Automation Controller configuration by default.

Additional Release Notes related to Ansible Automation Platform

- Check out latest features in Red Hat Enterprise Linux 9

- Read more about latest Ansible Automation Platform features