User Guide

Red Hat OpenShift Service on AWS

Red Hat OpenShift Service on AWS: User Guide

Table of Contents

What is Red Hat OpenShift Service on AWS?

Red Hat OpenShift Service on AWS (ROSA) is a managed service that you can use to build, scale, and deploy containerized applications with the Red Hat OpenShift enterprise Kubernetes platform on AWS. ROSA streamlines moving on-premises Red Hat OpenShift workloads to AWS, and offers tight integration with other AWS services.

Features

ROSA is jointly supported and operated by AWS and Red Hat. Each ROSA cluster comes with 24-hour Red Hat site reliability engineer (SRE) support for cluster management, backed by Red Hat's 99.95% uptime service-level agreement (SLA). For more information about the service's support model, see the section called "Support for ROSA".

ROSA also provides the following features:

- Red Hat SRE-supported cluster installation, cluster maintenance, and cluster upgrades.
- AWS service integrations include AWS compute, database, analytics, machine learning, networking, and mobile.
- Run and scale the Kubernetes control plane across multiple AWS Availability Zones to ensure high availability.
- Operate clusters using OpenShift APIs and developer productivity tools, including Service Mesh, CodeReady Workspaces, and Serverless.

ROSA cluster deployment models

ROSA provides two cluster deployment models: ROSA with hosted control planes (ROSA with HCP) and ROSA classic. With ROSA with HCP, each cluster has a dedicated control plane that is isolated within Red Hat's AWS account and managed by Red Hat. With ROSA classic, cluster control plane infrastructure is hosted in the customer's AWS account.

ROSA with HCP offers a more efficient control plane architecture that helps reduce the AWS infrastructure fees incurred when running ROSA and allows for faster cluster creation times. For more information about ROSA with HCP and ROSA classic, see the section called "Deployment options".

Note

ROSA with hosted control planes does not offer compliance certifications or Federal Information Processing Standards (FIPS) at this time. For more information, see Compliance in the Red Hat documentation.

Accessing ROSA

You can define and configure your ROSA service deployments using the following interfaces.

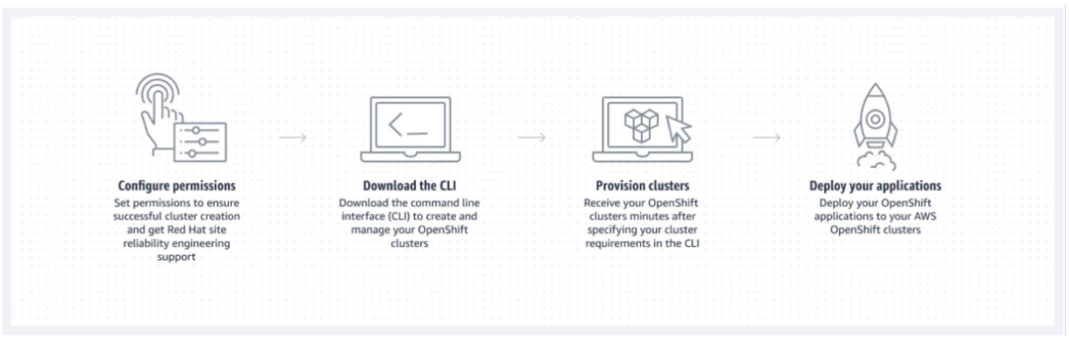
AWS

- **ROSA console** — Provides a web interface to enable the ROSA subscription and purchase a ROSA software contract.
- **AWS Command Line Interface (AWS CLI)** — Provides commands for a broad set of AWS services and is supported on Windows, macOS, and Linux. For more information, see AWS Command Line Interface.

Red Hat OpenShift

- **Red Hat Hybrid Cloud Console** — Provides a web interface to create, update, and manage ROSA clusters, install cluster add-ons, and create and deploy applications to a ROSA cluster.
- **ROSA CLI (rosa)** — Provides commands to create, update, and manage ROSA clusters.
- **OpenShift CLI (oc)** — Provides commands to create applications and manage OpenShift Container Platform projects.
- **Knative CLI (kn)** - Provides commands that can be used to interact with OpenShift Serverless components, such as Knative Serving and Eventing.
- **Pipelines CLI (tkn)** - Provides commands to interact with OpenShift Pipelines using the terminal.
- **opm CLI** - Provides commands that help Operator developers and cluster administrators create and maintain OpenShift Operator catalogs from the terminal.
- **Operator SDK CLI** - Provides commands that an Operator developer can use to build, test, and deploy an OpenShift operator.

How to get started with ROSA


Configure permissions
Set permissions to ensure successful cluster creation and get Red Hat site reliability engineering support
Download the CLI
Download the command line interface (CLI) to create and manage your OpenShift clusters
Provision clusters
Receive your OpenShift clusters minutes after specifying your cluster requirements in the CLI
Deploy your applications
Deploy your OpenShift applications to your AWS OpenShift clusters

The following summarizes the getting started process for ROSA. For detailed getting started instructions, see s.

AWS Management Console/AWS CLI

1. Configure permissions for AWS services that ROSA relies on to deliver service functionality. For more information, see the section called “Prerequisites”.
2. Install and configure the latest AWS CLI tool. For more information, see Installing our updating the latest version of the AWS CLI in the AWS CLI User Guide.
3. Enable ROSA in the ROSA console.

Red Hat Hybrid Cloud Console/ROSA CLI

1. Download the latest version of the ROSA CLI and OpenShift CLI from the Red Hat Hybrid Cloud Console. For more information, see Getting started with the ROSA CLI in the Red Hat documentation.
2. Create ROSA clusters in the Red Hat Hybrid Cloud Console or with the ROSA CLI.
3. When your cluster is ready, configure an identity provider to grant user access to the cluster.
4. Deploy and manage workloads on your ROSA cluster the same way that you would with any other OpenShift environment.

Pricing

The total cost of ROSA consists of two components: ROSA service fees and AWS infrastructure fees. For more information about pricing, see Red Hat OpenShift Service on AWS Pricing.

ROSA service fees

By default, ROSA service fees accrue on demand at an hourly rate per 4 vCPU used by worker nodes. Service fees are uniform across all supported AWS standard Regions. In addition to the worker node service fee, ROSA with hosted control planes (HCP) clusters incur an hourly cluster fee.

ROSA offers 1-year and 3-year service fee contracts that you can purchase for savings on the on-demand service fees for worker nodes. For more information, see the section called “ROSA contracts”.

AWS infrastructure fees

AWS infrastructure fees apply to the underlying worker nodes, infrastructure nodes, control plane nodes, storage, and network resources hosted on AWS global infrastructure. AWS infrastructure fees vary by AWS Region.

Overview of responsibilities for Red Hat OpenShift Service on AWS

This documentation outlines the responsibilities of Amazon Web Services (AWS), Red Hat, and customers for the Red Hat OpenShift Service on AWS (ROSA) managed service. For more information about ROSA and its components, see Policies and service definition in the Red Hat documentation.

The AWS shared responsibility model defines AWS responsibility for protecting the infrastructure that runs all of the services offered in the AWS Cloud, including ROSA. AWS infrastructure includes the hardware, software, networking, and facilities that run AWS Cloud services. This AWS responsibility is commonly referred to as the “security of the cloud”. To operate ROSA as a fully managed service, Red Hat and the customer are responsible for the elements of the service that the AWS responsibility model defines as “security in the cloud”.

Red Hat is responsible for the ongoing management and security of the ROSA cluster infrastructure, the underlying application platform, and the operating system. While ROSA clusters are hosted on AWS resources in the customer AWS accounts, they are accessed remotely by ROSA service components and Red Hat site reliability engineers (SREs) through IAM roles that the customer creates. Red Hat uses this access to manage the deployment and capacity of all control plane and infrastructure nodes on the cluster, and maintain versions for the control plane nodes, infrastructure nodes, and worker nodes.

Red Hat and the customer share responsibility for ROSA network management, cluster logging, cluster versioning, and capacity management. While Red Hat manages the ROSA service, the customer is fully responsible for managing and securing any applications, workloads, and data deployed to ROSA.

Overview

The following table provides an overview of AWS, Red Hat, and customer responsibilities for Red Hat OpenShift Service on AWS.

Note

If the `cluster-admin` role is added to a user, see the responsibilities and exclusion notes in the Red Hat Enterprise Agreement Appendix 4 (Online Subscription Services).

Resource	Incident and operations management	Change management	Access and identity authorization	Security and regulation compliance	Disaster recovery
Customer data	Customer	Customer	Customer	Customer	Customer
Customer applications	Customer	Customer	Customer	Customer	Customer
Developer services	Customer	Customer	Customer	Customer	Customer
Platform monitoring	Red Hat	Red Hat	Red Hat	Red Hat	Red Hat
Logging	Red Hat	Red Hat and customer	Red Hat and customer	Red Hat and customer	Red Hat
Application networking	Red Hat and customer	Red Hat and customer	Red Hat and customer	Red Hat	Red Hat
Cluster networking	Red Hat	Red Hat and customer	Red Hat and customer	Red Hat	Red Hat

Resource	Incident and operations management	Change management	Access and identity authorization	Security and regulation compliance	Disaster recovery
Virtual networking management	Red Hat and customer	Red Hat and customer	Red Hat and customer	Red Hat and customer	Red Hat and customer
Virtual compute management (control plane, infrastructure, and worker nodes)	Red Hat	Red Hat	Red Hat	Red Hat	Red Hat
Cluster version	Red Hat	Red Hat and customer	Red Hat	Red Hat	Red Hat
Capacity management	Red Hat	Red Hat and customer	Red Hat	Red Hat	Red Hat
Virtual storage management	Red Hat	Red Hat	Red Hat	Red Hat	Red Hat
AWS software (public AWS services)	AWS	AWS	AWS	AWS	AWS

Resource	Incident and operations management	Change management	Access and identity authorization	Security and regulation compliance	Disaster recovery
Hardware/ AWS global infrastructure	AWS	AWS	AWS	AWS	AWS

Tasks for shared responsibilities by area

AWS, Red Hat, and customers share responsibility for the monitoring and maintenance of ROSA components. This documentation defines ROSA service responsibilities by area and task.

Incident and operations management

AWS is responsible for protecting the hardware infrastructure that runs all of the services offered in the AWS Cloud. Red Hat is responsible for managing the service components necessary for default platform networking. The customer is responsible for incident and operations management of customer application data and any custom networking the customer may have configured.

Resource	Service responsibilities	Customer responsibilities
Application networking	**Red Hat** • Monitor native OpenShift router service, and respond to alerts.	**Customer** • Monitor health of application routes, and the endpoints behind them. • Report outages to AWS and Red Hat.
Virtual networking management	**Red Hat** • Monitor AWS load balancers, Amazon VPC subnets, and AWS service components necessary for	**Customer** • Monitor health of AWS load balancer endpoints.

Resource	Service responsibilities	Customer responsibilities
	default platform networking. Respond to alerts.	• Monitor network traffic that is optionally configured through Amazon VPC-to-VPC connection, AWS VPN connection, or AWS Direct Connect for potential issues or security threats.
Virtual storage management	**Red Hat** • Monitor Amazon EBS volumes used for cluster nodes, and Amazon S3 buckets used for the ROSA service's built-in container image registry. Respond to alerts.	**Customer** • Monitor health of application data. • If customer managed AWS KMS keys are used, create and control the key lifecycle and key policies for Amazon EBS encryption.
AWS software (public AWS services)	**AWS** • For information about AWS incident and operations management, see How AWS maintains operational resilience and continuity of service in the AWS whitepaper.	**Customer** • Monitor health of AWS resources in the customer account. • Use IAM tools to apply the appropriate permissions to AWS resources in the customer account.

Resource	Service responsibilities	Customer responsibilities
Hardware/AWS global infrastructure	**AWS** - For information about AWS incident and operations management, see How AWS maintains operational resilience and continuity of service in the AWS whitepaper.	**Customer** - Configure, manage, and monitor customer applications and data to ensure application and data security controls are properly enforced.

Change management

AWS is responsible for protecting the hardware infrastructure that runs all of the services offered in the AWS Cloud. Red Hat is responsible for enabling changes to the cluster infrastructure and services that the customer will control, as well as maintaining versions for the control plane nodes, infrastructure nodes, and worker nodes. The customer is responsible for initiating infrastructure changes. The customer is also responsible for installing and maintaining optional services, networking configurations on the cluster, and changes to customer data and applications.

Resource	Service responsibilities	Customer responsibilities
Logging	**Red Hat** - Centrally aggregate and monitor platform audit logs. - Provide and maintain a logging Operator to enable the customer to deploy a logging stack for default application logging. - Provide audit logs upon customer request.	**Customer** - Install the optional default application logging Operator on the cluster. - Install, configure, and maintain any optional app logging solutions, such as logging sidecar containers or third-party logging applications. - Tune size and frequency of application logs being produced by customer

Resource	Service responsibilities	Customer responsibilities
		applications if they are affecting the stability of the logging stack or the cluster. - Request platform audit logs through a support case for researching specific incidents.
Application networking	**Red Hat** - Set up public load balancers. Provide the ability to set up private load balancers and up to one additional load balancer when required. - Set up the native OpenShift router service. Provide the ability to set the router as private and add up to one additional router shard. - Install, configure, and maintain OpenShift SDN components for default internal pod traffic. - Provide the ability for the customer to manage `NetworkPolicy` and `EgressNetworkPolicy` (firewall) objects.	**Customer** - Configure non-default pod network permissions for project and pod networks, pod ingress, and pod egress using `NetworkPolicy` objects. - Use OpenShift Cluster Manager to request a private load balancer for default application routes. - Use OpenShift Cluster Manager to configure up to one additional public or private router shard and corresponding load balancer. - Request and configure any additional service load balancers for specific services. - Configure any necessary DNS forwarding rules.

Resource	Service responsibilities	Customer responsibilities
Cluster networking	**Red Hat** • Set up cluster management components, such as public or private service endpoints and necessary integration with Amazon VPCcomponents. • Set up internal networking components required for internal cluster communication between worker, infrastructure, and control plane nodes.	**Customer** • Provide optional non-default IP address ranges for machine CIDR, service CIDR, and pod CIDR if needed through OpenShift Cluster Manager when the cluster is provisioned. • Request that the API service endpoint be made public or private on cluster creation or after cluster creation through OpenShift Cluster Manager.

Resource	Service responsibilities	Customer responsibilities
Virtual networking management	**Red Hat** • Set up and configure Amazon VPC components required to provision the cluster, such as subnets, load balancers, internet gateways, and NAT gateways. • Provide the ability for the customer to manage AWS VPN connectivity with on-premises resources, Amazon VPC-to-VPC connectivity, and AWS Direct Connect as required through OpenShift Cluster Manager. • Enable customers to create and deploy AWS load balancers for use with service load balancers.	**Customer** • Set up and maintain optional Amazon VPC components, such as Amazon VPC-to-VPC connection, AWS VPN connection, or AWS Direct Connect. • Request and configure any additional load balancers for specific services.
Virtual compute management	**Red Hat** • Set up and configure the ROSA control plane and data plane to use Amazon EC2 instances for cluster compute. • Monitor and manage the deployment of Amazon EC2 control plane and infrastructure nodes on the cluster.	**Customer** • Monitor and manage Amazon EC2 worker nodes by creating a machine pool using the OpenShift Cluster Manager or ROSA CLI. • Manage changes to customer-deployed applications and application data.

Resource	Service responsibilities	Customer responsibilities
Cluster version	**Red Hat** - Enable upgrade scheduling process. - Monitor upgrade progress and remedy any issues encountered. - Publish change logs and release notes for minor and maintenance upgrades.	**Customer** - Schedule maintenance version upgrades either immediately, for the future, or have automatic upgrades. - Acknowledge and schedule minor version upgrades. - Ensure the cluster version stays on a supported minor version. - Test customer applications on minor and maintenan ce versions to ensure compatibility.

Resource	Service responsibilities	Customer responsibilities
Capacity management	**Red Hat** - Monitor the use of the control plane. Control planes include control plane nodes and infrastructure nodes. - Scale and resize control plane nodes to maintain quality of service.	**Customer** - Monitor worker node utilization and, if appropriate, enable the auto scaling feature. - Determine the scaling strategy of the cluster. See the additional resources for more information on machine pools. - Use the provided OpenShift Cluster Manager controls to add or remove additional worker nodes as required. - Respond to Red Hat notifications regarding cluster resource requirements.

Resource	Service responsibilities	Customer responsibilities
Virtual storage management	**Red Hat** • Set up and configure Amazon EBS to provision local node storage and persistent volume storage for the cluster. • Set up and configure the built-in image registry to use Amazon S3 bucket storage. • Regularly prune image registry resources in Amazon S3 to optimize Amazon S3 usage and cluster performance.	**Customer** • Optionally configure the Amazon EBS CSI driver or the Amazon EFS CSI driver to provision persistent volumes on the cluster.

Resource	Service responsibilities	Customer responsibilities
AWS software (public AWS services)	**AWS** **Compute** • Provide the Amazon EC2 service, used for ROSA control plane, infrastructure, and worker nodes. **Storage** • Provide Amazon EBS to allow the ROSA service to provision local node storage and persistent volume storage for the cluster. **Networking** • Provide the following AWS Cloud services to satisfy ROSA virtual networking infrastructure needs: • Amazon VPC • Elastic Load Balancing • IAM • Provide the following optional AWS service integrations for ROSA: • AWS VPN • AWS Direct Connect • AWS PrivateLink	**Customer** • Sign requests using an access key ID and secret access key associated with an IAM principal or AWS STS temporary security credentials. • Specify VPC subnets for the cluster to use during cluster creation. • Optionally configure a customer-managed VPC for use with ROSA clusters.

Resource	Service responsibilities	Customer responsibilities
	- AWS Transit Gateway	
Hardware/AWS global infrastructure	**AWS** - For information on management controls for AWS data centers, see Our Controls on the AWS Cloud Security page. - For information on change management best practices , see Guidance for Change Management on AWS in the AWS Solutions Library.	**Customer** - Implement change management best practices for customer applications and data hosted on the AWS Cloud.

Access and identity authorization

Access and identity authorization includes responsibilities for managing authorized access to clusters, applications, and infrastructure resources. This includes tasks such as providing access control mechanisms, authentication, authorization, and managing access to resources.

Resource	Service responsibilities	Customer responsibilities
Logging	**Red Hat** - Adhere to an industry standards-based tiered internal access process for platform audit logs. - Provide native OpenShift RBAC capabilities.	**Customer** - Configure OpenShift RBAC to control access to projects and by extension a project's application logs. - For third-party or custom application logging solutions, the customer is responsible for access management.

Resource	Service responsibilities	Customer responsibilities
Application networking	**Red Hat** - Provide native OpenShift RBAC and `dedicated-admin` capabilities.	**Customer** - Configure OpenShift `dedicated-admin` and RBAC to control access to route configuration as required. - Manage Red Hat organizat ion administrators for Red Hat to grant access to OpenShift Cluster Manager. The cluster manager is used to configure router options and provide service load balancer quota.
Cluster networking	**Red Hat** - Provide customer access controls through OpenShift Cluster Manager. Provide native OpenShift RBAC and `dedicated-admin` capabilities.	**Customer** - Configure OpenShift `dedicated-admin` and RBAC to control access to route configuration as required. - Manage Red Hat organizat ion membership of Red Hat accounts. - Manage organization administrators for Red Hat to grant access to OpenShift Cluster Manager.

Resource	Service responsibilities	Customer responsibilities
Virtual networking management	**Red Hat** • Provide customer access controls through OpenShift Cluster Manager.	**Customer** • Manage optional user access to AWS components through OpenShift Cluster Manager.
Virtual compute management	**Red Hat** • Provide customer access controls through OpenShift Cluster Manager.	**Customer** • Manage optional user access to AWS components through OpenShift Cluster Manager. • Create IAM roles and attached policies necessary to enable ROSA service access.
Virtual storage management	**Red Hat** • Provide customer access controls through OpenShift Cluster Manager.	**Customer** • Manage optional user access to AWS components through OpenShift Cluster Manager. • Create IAM roles and attached policies necessary to enable ROSA service access.

Resource	Service responsibilities	Customer responsibilities
AWS software (public AWS services)	**AWS** **Compute** • Provide the Amazon EC2 service, used for ROSA control plane, infrastructure, and worker nodes. **Storage** • Provide Amazon EBS, used to allow ROSA to provision local node storage and persistent volume storage for the cluster. • Provide Amazon S3, used for the service’s built-in image registry. **Networking** • Provide AWS Identity and Access Management (IAM), used by customers to control access to ROSA resources running on customer accounts.	**Customer** • Create IAM roles and attached policies necessary to enable ROSA service access. • Use IAM tools to apply the appropriate permissions to AWS resources in the customer account. • To enable ROSA across your AWS organization, the customer is responsible for managing AWS Organizations administrators. • To enable ROSA across your AWS organization, the customer is responsible for distributing the ROSA entitlement grant using AWS License Manager.

Resource	Service responsibilities	Customer responsibilities
Hardware/AWS global infrastructure	**AWS** - For information on physical access controls for AWS data centers, see Our Controls on the AWS Cloud Security page.	**Customer** - Customer is not responsib le for AWS global infrastru cture.

Security and regulation compliance

The following are the responsibilities and controls related to compliance:

Resource	Service responsibilities	Customer responsibilities
Logging	**Red Hat** - Send cluster audit logs to a Red Hat SIEM to analyze for security events. Retain audit logs for a defined period of time to support forensic analysis.	**Customer** - Analyze application logs for security events. - Send application logs to an external endpoint through logging sidecar container s or third- party logging applications if longer retention is required than is offered by the default logging stack.
Virtual networking management	**Red Hat** - Monitor virtual networkin g components for potential issues and security threats. - Use public AWS tools for additional monitoring and protection.	**Customer** - Monitor optional configure d virtual networking components for potential issues and security threats. - Configure any necessary firewall rules or customer

Resource	Service responsibilities	Customer responsibilities
		data center protections as required.
Virtual compute management	**Red Hat** - Monitor virtual compute components for potential issues and security threats. - Use public AWS tools for additional monitoring and protection.	**Customer** - Monitor optional configure d virtual networking components for potential issues and security threats. - Configure any necessary firewall rules or customer data center protections as required.

Resource	Service responsibilities	Customer responsibilities
Virtual storage management	**Red Hat** • Monitor virtual storage components for potential issues and security threats. • Use public AWS tools for additional monitoring and protection. • Configure the ROSA service to encrypt control plane, infrastructure, and worker node volume data by default using the AWS managed KMS key that Amazon EBS provides. • Configure the ROSA service to encrypt customer persistent volumes that use the default storage class with the AWS managed KMS key that Amazon EBS provides. • Provide the ability for the customer to use a customer managed KMS key to encrypt persistent volumes. • Configure the container image registry to encrypt image registry data at rest using server-side encryption with Amazon S3 managed keys (SSE-3). • Provide the ability for the customer to create a public	**Customer** • Provision Amazon EBS volumes. • Manage Amazon EBS volume storage to ensure enough storage is available to mount as a volume in ROSA. • Create the persistent volume claim and generate a persistent volume though OpenShift Cluster Manager.

Resource	Service responsibilities	Customer responsibilities
	or private Amazon S3 image registry to protect their container images from unauthorized user access.	

Resource	Service responsibilities	Customer responsibilities
AWS software (public AWS services)	**AWS** **Compute** • Provide Amazon EC2, used for ROSA control plane, infrastructure, and worker nodes. For more informati on, see Infrastructure security in Amazon EC2 in the Amazon EC2 User Guide. **Storage** • Provide Amazon EBS, used for ROSA control plane, infrastructure, and worker node volumes, as well as Kubernetes persistent volumes. For more information, see Data protection in Amazon EC2 in the Amazon EC2 User Guide. • Provide AWS KMS, which ROSA uses to encrypt control plane, infrastru cture, and worker node volumes and persisten t volumes. For more information, see Amazon EBS encryption in the Amazon EC2 User Guide.	**Customer** • Ensure security best practices and the principle of least privilege are followed to protect data on the Amazon EC2 instance. For more information, see Infrastructure security in Amazon EC2 and Data protection in Amazon EC2. • Monitor optional configure d virtual networking components for potential issues and security threats. • Configure any necessary firewall rules or customer data center protections as required. • Create an optional customer managed KMS key and encrypt the Amazon EBS persistent volume using the KMS key. • Monitor the customer data in virtual storage for potential issues and security threats. For more information, see the AWS Shared Responsibility Model.

Resource	Service responsibilities	Customer responsibilities
	- Provide Amazon S3, used for the ROSA service's built-in container image registry. For more information, see Amazon S3 security in the Amazon S3 User Guide. **Networking** - Provide security capabilit ies and services to increase privacy and control network access on AWS global infrastructure, including network firewalls built into Amazon VPC, private or dedicated network connections, and automatic encryption of all traffic on the AWS global and regional networks between AWS secured facilities. For more information, see the AWS Shared Responsibility Model and Infrastructure security in the Introduction to AWS Security whitepape r.	

Resource	Service responsibilities	Customer responsibilities
Hardware/AWS global infrastructure	**AWS** • Provide the AWS global infrastructure that ROSA uses to deliver service functionality. For more information about AWS security controls, see Security of the AWS Infrastructure in the AWS whitepaper. • Provide documentation for the customer to manage compliance needs and check their security state in AWS using tools such as AWS Artifact and AWS Security Hub.	**Customer** • Configure, manage, and monitor customer applications and data to ensure application and data security controls are properly enforced. • Use IAM tools to apply the appropriate permissions to AWS resources in the customer account.

Disaster recovery

Disaster recovery includes data and configuration backup, data replication and configuration of the disaster recovery environment, and failover on disaster events.

Resource	Service responsibilities	Customer responsibilities
Virtual networking management	**Red Hat** • Restore or recreate affected virtual network components that are necessary for the platform to function.	**Customer** • Configure virtual networking connections with more than one tunnel where possible for protection against outages. • Maintain failover DNS and load balancing if using a

Resource	Service responsibilities	Customer responsibilities
		global load balancer with multiple clusters.
Virtual compute management	**Red Hat** • Monitor the cluster and replace failed Amazon EC2 control plane or infrastru cture nodes. • Provide the ability for the customer to manually or automatically replace failed worker nodes.	**Customer** • Replace failed Amazon EC2 worker nodes by editing the machine pool configuration through OpenShift Cluster Manager or the ROSA CLI.
Virtual storage management	**Red Hat** • For ROSA clusters created with AWS IAM user credentials, back up all Kubernetes objects on the cluster through hourly, daily, and weekly volume snapshots.	**Customer** • Back up customer applicati ons and application data.

Resource	Service responsibilities	Customer responsibilities
AWS software (public AWS services)	**AWS** **Compute** • Provide Amazon EC2 features that support data resiliency such as Amazon EBS snapshots and Amazon EC2 Auto Scaling. For more information, see Resilience in Amazon EC2 in the Amazon EC2 User Guide. **Storage** • Provide the ability for the ROSA service and customers to back up the Amazon EBS volume on the cluster through Amazon EBS volume snapshots. • For information about Amazon S3 features that support data resiliency, see Resilience in Amazon S3. **Networking** • For information about Amazon VPC features that support data resiliency, see Resilience in Amazon Virtual Private Cloud in the Amazon VPC User Guide.	**Customer** • Configure ROSA multi-AZ clusters to improve fault tolerance and cluster availability. • Provision persistent volumes using the Amazon EBS CSI driver to enable volume snapshots. • Create CSI volume snapshots of Amazon EBS persistent volumes.

Resource	Service responsibilities	Customer responsibilities
Hardware/AWS global infrastructure	**AWS** • Provide AWS global infrastructure that allows ROSA to scale control plane, infrastructure, and worker nodes across Availability Zones. This functionality enables ROSA to orchestrate automatic failover between zones without interruption. • For more information about disaster recovery best practices, see Disaster recovery options in the cloud in the AWS Well-Architected Framework.	**Customer** • Configure ROSA multi-AZ clusters to improve fault tolerance and cluster availability.

Customer responsibilities for data and applications

The customer is responsible for the applications, workloads, and data that they deploy to Red Hat OpenShift Service on AWS. However, AWS and Red Hat provide various tools to help the customer manage data and applications on the platform.

Resource	How AWS and Red Hat helps	Customer responsibilities
Customer data	**Red Hat** • Maintain platform-level standards for data encryption as defined by industry security and compliance standards.	**Customer** • Maintain responsibility for all customer data stored on the platform and how customer applications consume and expose this data.

Resource	How AWS and Red Hat helps	Customer responsibilities
	• Provide OpenShift components to help manage application data, such as secrets. • Enable integration with data services such as Amazon RDS to store and manage data outside of the cluster and/or AWS. **AWS** • Provide Amazon RDS to allow customers to store and manage data outside of the cluster.	

Resource	How AWS and Red Hat helps	Customer responsibilities
Customer applications	**Red Hat** • Provision clusters with OpenShift components installed so that customers can access the OpenShift and Kubernetes APIs to deploy and manage containerized applications • Create clusters with image pull secrets so that customer deployments can pull images from the Red Hat Container Catalog registry. • Provide access to OpenShift APIs that a customer can use to set up Operators to add community, third-party, AWS, and Red Hat services to the cluster. • Provide storage classes and plugins to support persistent volumes for use with customer applications. • Provide a container image registry so customers can securely store applicati on container images on the cluster to deploy and manage applications. **AWS**	**Customer** • Maintain responsibility for customer and third-par ty applications, data, and the complete application lifecycle. • If a customer adds Red Hat, community, third-party, their own, or other services to the cluster by using Operators or external images, the customer is responsible for these services and for working with the appropriate provider (including Red Hat) to troubleshoot any issues. • Use the provided tools and features to configure and deploy; keep up to date; set up resource requests and limits; size the cluster to have enough resources to run apps; set up permissio ns; integrate with other services; manage any image streams or templates that the customer deploys; externally serve; save, back up, and restore data; and otherwise manage

Resource	How AWS and Red Hat helps	Customer responsibilities
	• Provide Amazon EBS to support persistent volumes for use with customer applications. • Provide Amazon S3 to support Red Hat provision ing of the container image registry.	their highly available and resilient workloads. • Maintain responsibility for monitoring the applications run on Red Hat OpenShift Service on AWS, including installing and operating software to gather metrics, create alerts, and protect secrets in the application.

Deployment options

ROSA provides two cluster deployment models: ROSA with hosted control planes (ROSA with HCP) and ROSA classic. With ROSA with HCP, each cluster has a dedicated control plane that is isolated within Red Hat's AWS account and managed by Red Hat. With ROSA classic, cluster control plane infrastructure is hosted in the customer's AWS account.

ROSA with HCP offers a more efficient control plane architecture that helps reduces the AWS infrastructure fees incurred when running ROSA and allows for faster cluster creation times. Both cluster deployment models can be enabled in the AWS ROSA console. You have the choice to select which deployment model you want to use when you provision ROSA clusters using the ROSA CLI.

> **Note**
>
> ROSA with hosted control planes does not offer compliance certifications or Federal Information Processing Standards (FIPS) at this time. For more information, see Compliance in the Red Hat documentation.

Differences between ROSA with HCP and ROSA classic

There are several technical differences between ROSA with HCP and ROSA classic.

	ROSA with HCP	ROSA classic
Cluster infrastructure hosting	• Control plane components, such as etcd, API server, and oauth, are hosted on Red Hat-owned and managed AWS accounts. Worker node infrastructure is hosted on the customer's AWS account. Does not use dedicated infrastructure nodes; platform components are deployed to worker nodes.	• Control plane components are hosted on the customer's AWS account, alongside infrastructure and worker nodes.
Provisioning time	• Approximately 10 minutes.	• Approximately 40 minutes.
Architecture	• Control plane infrastructure is fully managed by Red Hat. Control plane infrastructure is not directly available to end customers, except through dedicated and explicitly exposed endpoints. • Worker nodes are hosted on the customer's AWS account.	• Control plane infrastructure is hosted in the customer's AWS account. • Worker nodes are hosted on the customer's AWS account.
AWS Identity and Access Management	• Uses AWS managed policies.	• Uses customer managed policies that are defined by the service.
Minimum Amazon EC2 footprint	• One cluster requires a minimum of two nodes hosted on the customer's AWS account.	• One cluster requires a minimum of seven nodes hosted on the customer's AWS account.

	ROSA with HCP	ROSA classic
Cluster provisioning	• Provision clusters using the ROSA CLI. • Customers provision clusters that deploy the control plane components into Red Hat's AWS account. • Customers provision machine pools that deploy worker nodes into the customer's AWS account.	• Provision clusters using the ROSA CLI or web UI. • Cluster control plane, worker nodes, and infrastructure nodes are provisioned into the customer's AWS account.
Upgrades	• Upgrade control plane and machine pools separately.	• Entire cluster must be upgraded at the same time.
AWS Regions	• For AWS Region availability, see Red Hat OpenShift Service on AWS endpoints and quotas in the AWS General Reference Guide.	• For AWS Region availability, see Red Hat OpenShift Service on AWS endpoints and quotas in the AWS General Reference Guide.
Compliance	• For compliance information, see Compliance in the Red Hat documentation.	• For compliance information, see Compliance in the Red Hat documentation.

Getting started with ROSA

Red Hat OpenShift Service on AWS (ROSA) is a managed service that you can use to build, scale, and deploy containerized applications with the Red Hat OpenShift enterprise Kubernetes platform on AWS.

ROSA cluster deployment models

ROSA supports two cluster deployment models: ROSA with hosted control planes (ROSA with HCP) and ROSA classic. ROSA with HCP offers a more efficient control plane architecture that reduces the AWS infrastructure costs for ROSA and allows for faster cluster creation times. For more information about ROSA with HCP and ROSA classic, see the section called “Deployment options”.

> **Note**
>
> ROSA with hosted control planes does not offer FIPS at this time.

Getting started guides

There are four getting started guides available for deploying an application to a newly created ROSA cluster. Each tutorial covers the following:

- Enabling the ROSA service and configuring AWS prerequisites
- Creating the necessary IAM roles and policies
- Creating the ROSA cluster
- Creating a cluster administrator for quick cluster access
- Configuring an identity provider
- Granting user access to the cluster
- Deploy an application to the cluster
- Deleting the cluster and cluster resources

Getting started with ROSA with HCP

With ROSA with HCP, you can use AWS STS and the ROSA CLI to create a cluster with the necessary IAM roles and policies attached. For more information about IAM policies for ROSA with HCP, see the section called " AWS managed IAM policies".

Once the cluster is created, you can deploy public application workloads to the cluster using the Red Hat Hybrid Cloud Console or the OpenShift CLI. For steps to deploy an application to a newly created ROSA with HCP cluster, see the section called "Using ROSA with HCP and the ROSA CLI in auto mode".

Getting started with ROSA classic

With ROSA classic, you can use AWS STS and the ROSA CLI to create a cluster with the necessary IAM roles and policies attached. Once the cluster is created, you can then deploy public application workloads to the cluster using the Red Hat Hybrid Cloud Console or the OpenShift CLI. For steps to get started using the ROSA CLI's automatic cluster creation (`auto`) mode, see the section called "Using ROSA classic with the ROSA CLI in auto mode". For steps to get started using the ROSA CLI's manual cluster creation (`manual`) mode, see the section called "Using ROSA classic with the ROSA CLI in manual mode".

If you require the ROSA classic cluster and application workloads to be private, see the section called "Using ROSA classic with AWS PrivateLink ".

Getting started with ROSA with HCP using the ROSA CLI in auto mode

The following sections describe how to get started with ROSA with hosted control planes (ROSA with HCP) using AWS STS and the ROSA CLI. For more information about ROSA with HCP, see the section called "Deployment options".

The ROSA CLI uses `auto` mode or `manual` mode to create the IAM resources and OpenID Connect (OIDC) configuration required to create a ROSA cluster. `auto` mode automatically creates the required IAM roles and policies and OIDC provider. `manual` mode outputs the AWS CLI commands that are needed to create the IAM resources manually. By using `manual` mode, you can review the generated AWS CLI commands before running them manually. With `manual` mode, you can also pass the commands to another administrator or group in your organization so they can create the resources.

The procedures in this document use the `auto` mode of the ROSA CLI to create the required IAM resources and OIDC configuration for ROSA with HCP. For more options to get started, see *Getting started with ROSA* .

Topics

Prerequisites

Before getting started, make sure you completed these actions:

- Install and configure the latest AWS CLI. For more information, see Installing or updating the latest version of the AWS CLI.
- Install and configure the latest ROSA CLI and OpenShift Container Platform CLI. For more information, see Getting started with the ROSA CLI.
- Service Quotas must have the required service quotas set for Amazon EC2, Amazon VPC, Amazon EBS, and Elastic Load Balancing that are needed to create and run a ROSA cluster. AWS or Red Hat may request service quota increases on your behalf as required for issue resolution. To view the required quotas, see Red Hat OpenShift Service on AWS endpoints and quotas in the *AWS General Reference*.
- To receive AWS support for ROSA, you must enable AWS Business, Enterprise On-Ramp, or Enterprise support plans. Red Hat may request AWS support on your behalf as required for issue resolution. For more information, see the section called "Support for ROSA". To enable AWS Support, see the AWS Support page.

- If you're using AWS Organizations to manage the AWS accounts that host the ROSA service, the organization's service control policy (SCP) must be configured to allow Red Hat to perform policy actions that's listed in the SCP without restriction. For more information, see the the section called "AWS Organizations service control policy denies required AWS Marketplace permissions". For more information about SCPs, see Service control policies (SCPs).
- If deploying a ROSA cluster with AWS STS into an enabled AWS Region that's disabled by default, you must update the security token to version 2 for all the Regions in the AWS account with the following command.

  ```
  aws iam set-security-token-service-preferences --global-endpoint-token-version
   v2Token
  ```

  For more information about enabling Regions, see Managing AWS Regions in the *AWS General Reference*.

Step 1: Enable ROSA and configure prerequisites

To create a ROSA cluster, you must first enable the ROSA service in the AWS ROSA console. The AWS ROSA console verifies if your AWS account has the necessary AWS Marketplace permissions, service quotas, and the Elastic Load Balancing (ELB) service-linked role named `AWSServiceRoleForElasticLoadBalancing`. If any of these prerequisites are missing, the console provides guidance on how to configure your account to meet the prerequisites.

1. Navigate to the ROSA console.
2. Choose **Get started**.
3. On the **Verify ROSA prerequisites** page, select **I agree to share my contact information with Red Hat**.
4. Choose **Enable ROSA** .
5. Once the page has verified your service quotas meet ROSA prerequisites and the ELB service-linked role is created, open a new terminal session to create your first ROSA cluster using the ROSA CLI.

Step 2: Create Amazon VPC architecture for ROSA with HCP clusters

To create a ROSA with HCP cluster, you must first configure your own Amazon VPC architecture to deploy your solution into. ROSA with HCP requires that customers configure at least one public and private subnet per Availability Zone used to create clusters. For single-AZ clusters, only use Availability Zone is used. For multi-AZ clusters, three Availability Zones are needed.

> ⚠ **Important**
>
> If Amazon VPC requirements are not met, cluster creation fails.

The following procedure uses the AWS CLI to create both a public and private subnet into a single Availability Zone for a Single-AZ cluster. All cluster resources are in the private subnet. The public subnet routes outbound traffic by using a NAT gateway to the internet.

This example uses the CIDR block `10.0.0.0/16` for the Amazon VPC. However, you can choose a different CIDR block. For more information, see VPC sizing.

1. Set an environment variable for the cluster name by running the following command.

```
ROSA_CLUSTER_NAME=rosa-hcp
```

2. Create a VPC with a `10.0.0.0/16` CIDR block.

```
aws ec2 create-vpc --cidr-block 10.0.0.0/16 --query Vpc.VpcId --output text
```

The preceding command returns the ID of the new VPC. The following is an example output.

```
vpc-0410832ee325aafea
```

3. Using the VPC ID from the previous step, tag the VPC using the `ROSA_CLUSTER_NAME` variable.

```
aws ec2 create-tags --resources <VPC_ID_VALUE> --tags Key=Name,Value=
$ROSA_CLUSTER_NAME
```

4. Enable DNS hostname support on the VPC.

```
aws ec2 modify-vpc-attribute --vpc-id <VPC_ID_VALUE> --enable-dns-hostnames
```

5. Create a public subnet in the VPC with a `10.0.1.0/24` CIDR block, specifying the Availability Zone where the resource should be created.

Important

When creating subnets, make sure that subnets are created to an Availability Zone that has ROSA instance types available. If you don't choose a specific Availability Zone, the subnet is created in any one of the Availability Zones in the AWS Region that you specify. To specify a specific Availability Zone, use the `--availability zone` argument in the `create-subnet` command. You can use the `rosa list instance-types` command to list all ROSA instance types available. To check if an instance type is available for a given Availability Zone, use the following command.

```
aws ec2 describe-instance-type-offerings --location-type availability-zone --
filters Name=location,Values=<availability_zone> --region <region> --output
 text | egrep "<instance_type>"
```

Important

ROSA with HCP requires that customers configure at least one public and private subnet per Availability Zone used to create clusters. For single-AZ clusters, only one Availability Zone is needed. For multi-AZ clusters, three Availability Zones are needed. If these requirements are not met, cluster creation fails.

```
aws ec2 create-subnet --vpc-id <VPC_ID_VALUE> --cidr-block 10.0.1.0/24 --
availability-zone <AZ_NAME> --query Subnet.SubnetId --output text
```

The preceding command returns the ID of the new subnet. The following is an example output.

```
subnet-0b6a7e8cbc8b75920
```

6. Using the subnet ID from the previous step, tag the subnet using the `ROSA_CLUSTER_NAME-public` variable.

```
aws ec2 create-tags --resources <PUBLIC_SUBNET_ID> --tags Key=Name,Value=
$ROSA_CLUSTER_NAME-public
```

7. Create a private subnet in the VPC with a `10.0.0.0/24` CIDR block, specifying the same Availability Zone that the public subnet deployed into.

```
aws ec2 create-subnet --vpc-id <VPC_ID_VALUE> --cidr-block 10.0.0.0/24 --
availability-zone <AZ_NAME> --query Subnet.SubnetId --output text
```

   The preceding command returns the ID of the new subnet. The following is an example output.

```
subnet-0b6a7e8cbc8b75920
```

8. Using the subnet ID from the previous step, tag the subnet using the `ROSA_CLUSTER_NAME-private` variable.

```
aws ec2 create-tags --resources <PRIVATE_SUBNET_ID> --tags Key=Name,Value=
$ROSA_CLUSTER_NAME-private
```

9. Create an internet gateway for outbound traffic and attach it to the VPC.

```
aws ec2 create-internet-gateway --query InternetGateway.InternetGatewayId --output
 text

aws ec2 attach-internet-gateway --vpc-id <VPC_ID_VALUE> --internet-gateway-id
 <IG_ID_VALUE>
```

10. Tag the internet gateway with the `ROSA_CLUSTER_NAME` variable.

```
aws ec2 create-tags --resources <IG_ID_VALUE> --tags Key=Name,Value=
$ROSA_CLUSTER_NAME
```

11. Create a route table for outbound traffic, associate it to the public subnet, and configure traffic to route to the internet gateway.

```
aws ec2 create-route-table --vpc-id <VPC_ID_VALUE> --query RouteTable.RouteTableId --
output text

aws ec2 associate-route-table --subnet-id <PUBLIC_SUBNET_ID> --route-table-id
 <PUBLIC_RT_ID>

aws ec2 create-route --route-table-id <PUBLIC_RT_ID> --destination-cidr-block
 0.0.0.0/0 --gateway-id <IG_ID_VALUE>
```

12. Tag the public route table with the ROSA_CLUSTER_NAME variable and verify that the route table was properly configured.

```
aws ec2 create-tags --resources <PUBLIC_RT_ID> --tags Key=Name,Value=
$ROSA_CLUSTER_NAME

aws ec2 describe-route-tables --route-table-id <PUBLIC_RT_ID>
```

13. Create a NAT gateway in the public subnet with an elastic IP address to enable traffic to the private subnet.

```
aws ec2 allocate-address --domain vpc --query AllocationId --output text

aws ec2 create-nat-gateway --subnet-id <PUBLIC_SUBNET_ID> --allocation-id
 <EIP_ADDRESS>  --query NatGateway.NatGatewayId --output text
```

14. Tag the NAT gateway and elastic IP address with the $ROSA_CLUSTER_NAME variable.

```
aws ec2 create-tags --resources <EIP_ADDRESS>  --resources <NAT_GATEWAY_ID> --tags
 Key=Name,Value=$ROSA_CLUSTER_NAME
```

15. Create a route table for private subnet traffic, associate it to the private subnet, and configure traffic to route to the NAT gateway.

```
aws ec2 create-route-table --vpc-id <VPC_ID_VALUE> --query RouteTable.RouteTableId --
output text

aws ec2 associate-route-table --subnet-id <PRIVATE_SUBNET_ID> --route-table-id
 <PRIVATE_RT_ID>

aws ec2 create-route --route-table-id <PRIVATE_RT_ID> --destination-cidr-block
 0.0.0.0/0 --gateway-id <NAT_GATEWAY_ID>
```

16. Tag the private route table and elastic IP address with the $ROSA_CLUSTER_NAME-private variable.

```
aws ec2 create-tags --resources <PRIVATE_RT_ID> <EIP_ADDRESS> --tags Key=Name,Value=
$ROSA_CLUSTER_NAME-private
```

Step 3: Create the required IAM roles and OpenID Connect configuration

Before creating a ROSA with HCP cluster, you must create the necessary IAM roles and policies and the OpenID Connect (OIDC) configuration. For more information about IAM roles and policies for ROSA with HCP, see the section called “ AWS managed IAM policies”.

This procedure uses the `auto` mode of the ROSA CLI to automatically create the OIDC configuration necessary to create a ROSA with HCP cluster.

1. Create the required IAM account roles and policies.

   ```
   rosa create account-roles --force-policy-creation
   ```

   The --force-policy-creation parameter updates any existing roles and policies that are present. If no roles and policies are present, the command creates these resources instead.

   > **Note**
   >
   > If your offline access token has expired, the ROSA CLI outputs an error message stating that your authorization token needs updated. For steps to troubleshoot, see the section called “Troubleshoot ROSA CLI expired offline access tokens”.

2. Create the OpenID Connect (OIDC) configuration that enables user authentication to the cluster. This configuration is registered to be used with OpenShift Cluster Manager (OCM).

   ```
   rosa create oidc-config --mode=auto
   ```

3. Copy the OIDC config ID provided in the ROSA CLI output. The OIDC config ID needs to be provided later to create the ROSA with HCP cluster.

4. To verify the OIDC configurations available for clusters associated with your user organization, run the following command.

   ```
   rosa list oidc-config
   ```

5. Create the required IAM operator roles, replacing `<OIDC_CONFIG_ID>` with the OIDC config ID copied previously.

Example

> ⚠ **Important**
>
> You must supply a prefix in <PREFIX_NAME> when creating the Operator roles. Failing to do so produces an error.

```
rosa create operator-roles --prefix <PREFIX_NAME> --oidc-config-id <OIDC_CONFIG_ID>
 --hosted-cp
```

6. To verify the IAM operator roles were created, run the following command:

```
rosa list operator-roles
```

Step 4: Create a ROSA with HCP cluster with AWS STS and the ROSA CLI `auto` mode

You can create a ROSA with HCP cluster using AWS Security Token Service (AWS STS) and the `auto` mode that's provided in the ROSA CLI. You have the option to create a cluster with a public API and Ingress or a private API and Ingress.

You can create a cluster with a single Availability Zone (Single-AZ) or multiple Availability Zones (Multi-AZ). In either case, your machine's CIDR value must match your VPC's CIDR value.

The following procedure uses the `rosa create cluster --hosted-cp` command to create a Single-AZ ROSA with HCP cluster. To create a Multi-AZ cluster, specify `multi-az` in the command and the private subnet IDs for each private subnet you want you to deploy to.

1. Create a ROSA with HCP cluster with one of the following commands.

   - Create a ROSA with HCP cluster with a public API and Ingress, specifying the cluster name, operator role prefix, OIDC config ID, and public and private subnet IDs.

     ```
     rosa create cluster --cluster-name=<CLUSTER_NAME> --sts --mode=auto --hosted-cp --
     operator-roles-prefix <OPERATOR_ROLE_PREFIX> --oidc-config-id <OIDC_CONFIG_ID> --
     subnet-ids=<PUBLIC_SUBNET_ID>,<PRIVATE_SUBNET_ID>
     ```

- Create a ROSA with HCP cluster with a private API and Ingress, specifying the cluster name, operator role prefix, OIDC config ID, and private subnet IDs.

```
rosa create cluster --private --cluster-name=<CLUSTER_NAME> --sts --mode=auto --
hosted-cp --subnet-ids=<PRIVATE_SUBNET_ID>
```

2. Check the status of your cluster.

```
rosa describe cluster -c <CLUSTER_NAME>
```

> **Note**
>
> If the creation process fails or the `State` field doesn't change to a ready status after 10 minutes, see the section called " ROSA cluster creation issues".
> To contact AWS Support or Red Hat support for assistance, see the section called "Support for ROSA".

3. Track the progress of the cluster creation by watching the OpenShift installer logs.

```
rosa logs install -c <CLUSTER_NAME> --watch
```

Step 5: Configure an identity provider and grant cluster access

ROSA includes a built-in OAuth server. After your cluster is created, you must configure OAuth to use an identity provider. You can then add users to your configured identity provider to grant them access to your cluster. You can grant these users `cluster-admin` or `dedicated-admin` permissions as required.

You can configure different identity provider types for your ROSA cluster. Supported types include GitHub, GitHub Enterprise, GitLab, Google, LDAP, OpenID Connect, and HTPasswd identity providers.

> **Important**
>
> The HTPasswd identity provider is included only to enable a single, static administrator user to be created. HTPasswd isn't supported as a general-use identity provider for ROSA.

The following procedure configures a GitHub identity provider as an example. For instructions on how to configure each of the supported identity provider types, see Configuring identity providers for AWS STS.

1. Navigate to github.com and log in to your GitHub account.
2. If you don't have a GitHub organization to use for identity provisioning for your cluster, create one. For more information, see the steps in the GitHub documentation.
3. Using the ROSA CLI's interactive mode, configure an identity provider for your cluster.

```
rosa create idp --cluster=<CLUSTER_NAME> --interactive
```

4. Follow the configuration prompts in the output to restrict cluster access to members of your GitHub organization.

```
I: Interactive mode enabled.
Any optional fields can be left empty and a default will be selected.
? Type of identity provider: github
? Identity provider name: github-1
? Restrict to members of: organizations
? GitHub organizations: <GITHUB_ORG_NAME>
? To use GitHub as an identity provider, you must first register the application:
  - Open the following URL:
    https://github.com/organizations/<GITHUB_ORG_NAME>/settings/
applications/new?oauth_application%5Bcallback_url%5D=https%3A%2F%2Foauth-
openshift.apps.<CLUSTER_NAME>/<RANDOM_STRING>.p1.openshiftapps.com%2Foauth2callback
%2Fgithub-1&oauth_application%5Bname%5D=<CLUSTER_NAME>&oauth_application
%5Burl%5D=https%3A%2F%2Fconsole-openshift-console.apps.<CLUSTER_NAME>/
<RANDOM_STRING>.p1.openshiftapps.com
  - Click on 'Register application'
...
```

5. Open the URL in the output, replacing `<GITHUB_ORG_NAME>` with the name of your GitHub organization.
6. On the GitHub web page, choose **Register application** to register a new OAuth application in your GitHub organization.
7. Use the information from the GitHub OAuth page to populate the remaining `rosa create idp` interactive prompts by running the following command. Replace `<GITHUB_CLIENT_ID>` and `<GITHUB_CLIENT_SECRET>` with the credentials from your GitHub OAuth application.

```
...
```

```
? Client ID: <GITHUB_CLIENT_ID>
? Client Secret: [? for help] <GITHUB_CLIENT_SECRET>
? GitHub Enterprise Hostname (optional):
? Mapping method: claim
I: Configuring IDP for cluster '<CLUSTER_NAME>'
I: Identity Provider 'github-1' has been created.
   It will take up to 1 minute for this configuration to be enabled.
   To add cluster administrators, see 'rosa grant user --help'.
   To login into the console, open https://console-openshift-
console.apps.<CLUSTER_NAME>.<RANDOM_STRING>.p1.openshiftapps.com and click on
 github-1.
```

> **Note**
>
> It might take approximately two minutes for the identity provider configuration to become active. If you configured a `cluster-admin` user, you can run `oc get pods -n openshift-authentication --watch` to watch the OAuth pods redeploy with the updated configuration.

8. Verify that the identity provider is configured correctly.

```
rosa list idps --cluster=<CLUSTER_NAME>
```

Step 6: Grant user access to a cluster

You can grant a user access to your cluster by adding them to the configured identity provider.

The following procedure adds a user to a GitHub organization that’s configured for identity provisioning to the cluster.

1. Navigate to github.com and log in to your GitHub account.
2. Invite users that require cluster access to your GitHub organization. For more information, see Inviting users to join your organization in the GitHub documentation.

Step 7: Grant administrator permissions to a user

After you add a user to your configured identity provider, you can grant the user `cluster-admin` or `dedicated-admin` permissions for your cluster.

Configure `cluster-admin` permissions

1. Grant the `cluster-admin` permissions by running the following command. Replace `<IDP_USER_NAME>` and `<CLUSTER_NAME>` with your user and cluster name.

```
rosa grant user cluster-admin --user=<IDP_USER_NAME> --cluster=<CLUSTER_NAME>
```

2. Verify that the user is listed as a member of the `cluster-admins` group.

```
rosa list users --cluster=<CLUSTER_NAME>
```

Configure `dedicated-admin` permissions

1. Grant the `dedicated-admin` permissions by using the following command. Replace `<IDP_USER_NAME>` and `<CLUSTER_NAME>` with your user and cluster name by running the following command.

```
rosa grant user dedicated-admin --user=<IDP_USER_NAME> --cluster=<CLUSTER_NAME>
```

2. Verify that the user is listed as a member of the `cluster-admins` group.

```
rosa list users --cluster=<CLUSTER_NAME>
```

Step 8: Access a cluster through the Red Hat Hybrid Cloud Console

Log in to your cluster through the Red Hat Hybrid Cloud Console.

1. Obtain the console URL for your cluster using the following command. Replace `<CLUSTER_NAME>` with the name of your cluster.

```
rosa describe cluster -c <CLUSTER_NAME> | grep Console
```

2. Navigate to the console URL in the output and log in.

   In the **Log in with...** dialog, choose the identity provider name and complete any authorization requests presented by your provider.

Step 9: Deploy an application from the Developer Catalog

From the Red Hat Hybrid Cloud Console, you can deploy a Developer Catalog test application and expose it with a route.

1. Navigate to Red Hat Hybrid Cloud Console and choose the cluster you want to deploy the app into.
2. On the cluster's page, choose **Open console**.
3. In the **Administrator** perspective, choose **Home** > **Projects** > **Create Project**.
4. Enter a name for your project and optionally add a **Display Name** and **Description**.
5. Choose **Create** to create the project.
6. Switch to the **Developer** perspective and choose **+Add**. Make sure that the selected project is the one that was just created.
7. In the **Developer Catalog** dialog, choose **All services**.
8. In the **Developer Catalog** page, choose **Languages** > **JavaScript** from the menu.
9. Choose **Node.js**, and then choose **Create Application** to open the **Create Source-to-Image Application** page.

   > **Note**
   >
   > You might need to choose **Clear All Filters** to display the **Node.js** option.

10. In the **Git** section, choose **Try Sample**.
11. In the **Name** field, add a unique name.
12. Choose **Create**.

   > **Note**
   >
   > The new application takes several minutes to deploy.

13. When the deployment is complete, choose the route URL for the application.

    A new tab in the browser opens with a message that's similar to the following.

```
Welcome to your Node.js application on OpenShift
```

14(Optional) Delete the application and clean up resources:

a. In the **Administrator** perspective, choose **Home** > **Projects**.

b. Open the action menu for your project and choose **Delete Project**.

Step 10: Delete a cluster and AWS STS resources

You can use the ROSA CLI to delete a cluster that uses AWS Security Token Service (AWS STS). You can also use the ROSA CLI to delete the IAM roles and OIDC provider created by ROSA. To delete the IAM policies created by ROSA, you can use the IAM console.

> ⚠ **Important**
>
> IAM roles and policies created by ROSA might be used by other ROSA clusters in the same account.

1. Delete the cluster and watch the logs. Replace <CLUSTER_NAME> with the name or ID of your cluster.

```
rosa delete cluster --cluster=<CLUSTER_NAME> --watch
```

> ⚠ **Important**
>
> You must wait for the cluster to delete completely before you remove the IAM roles, policies, and OIDC provider. The account IAM roles are required to delete the resources created by the installer. The operator IAM roles are required to clean up the resources created by the OpenShift operators. The operators use the OIDC provider to authenticate.

2. Delete the OIDC provider that the cluster operators use to authenticate by running the following command.

```
rosa delete oidc-provider -c <CLUSTER_ID> --mode auto
```

3. Delete the cluster-specific operator IAM roles.

```
rosa delete operator-roles -c <CLUSTER_ID> --mode auto
```

4. Delete the account IAM roles using the following command. Replace <PREFIX> with the prefix of the account IAM roles to delete. If you specified a custom prefix when creating the account IAM roles, specify the default ManagedOpenShift prefix.

```
rosa delete account-roles --prefix <PREFIX> --mode auto
```

5. Delete the IAM policies created by ROSA.

   a. Log in to the IAM console.

   b. On the left menu under **Access management**, choose **Policies**.

   c. Select the policy that you want to delete and choose **Actions** > **Delete**.

   d. Enter the policy name and choose **Delete**.

   e. Repeat this step to delete each of the IAM policies for the cluster.

Getting started with ROSA classic using the ROSA CLI in auto mode

The following sections describe how to get started with ROSA classic using AWS STS and the ROSA CLI. For more information about ROSA classic, see the section called "Deployment options".

The ROSA CLI uses `auto` mode or `manual` mode to create the IAM resources required to provision a ROSA cluster. `auto` mode immediately creates the required IAM roles and policies and an OpenID Connect (OIDC) provider. `manual` mode outputs the AWS CLI commands that are needed to create the IAM resources. By using `manual` mode, you can review the generated AWS CLI commands before running them manually. With `manual` mode, you can also pass the commands to another administrator or group in your organization so they can create the resources.

The procedures in this document use the `auto` mode of the ROSA CLI to create the required IAM resources for ROSA classic. For more options to get started, see *Getting started with ROSA* .

Topics

- Step 7: Deploy an application from the Developer Catalog
- Step 8: Revoke administrator permissions and user access
- Step 9: Delete a cluster and AWS STS resources

Prerequisites

Before getting started, make sure you completed these actions:

- Install and configure the latest AWS CLI. For more information, see Installing or updating the latest version of the AWS CLI.
- Install and configure the latest ROSA CLI and OpenShift Container Platform CLI. For more information, see Getting started with the ROSA CLI.
- Service Quotas must have the required service quotas set for Amazon EC2, Amazon VPC, Amazon EBS, and Elastic Load Balancing that are needed to create and run a ROSA cluster. AWS or Red Hat may request service quota increases on your behalf as required for issue resolution. To view the required quotas, see Red Hat OpenShift Service on AWS endpoints and quotas in the *AWS General Reference*.
- To receive AWS support for ROSA, you must enable AWS Business, Enterprise On-Ramp, or Enterprise support plans. Red Hat may request AWS support on your behalf as required for issue resolution. For more information, see the section called "Support for ROSA". To enable AWS Support, see the AWS Support page.
- If you're using AWS Organizations to manage the AWS accounts that host the ROSA service, the organization's service control policy (SCP) must be configured to allow Red Hat to perform policy actions that's listed in the SCP without restriction. For more information, see the the section called "AWS Organizations service control policy denies required AWS Marketplace permissions". For more information about SCPs, see Service control policies (SCPs).
- If deploying a ROSA cluster with AWS STS into an enabled AWS Region that's disabled by default, you must update the security token to version 2 for all the Regions in the AWS account with the following command.

  ```
  aws iam set-security-token-service-preferences --global-endpoint-token-version
   v2Token
  ```

  For more information about enabling Regions, see Managing AWS Regions in the *AWS General Reference*.

Step 1: Enable ROSA and configure prerequisites

To create a ROSA cluster, you must first enable the ROSA service in the AWS ROSA console and verify that AWS prerequisites have been met. The AWS ROSA console verifies if your AWS account has the necessary AWS Marketplace permissions, service quotas, and the Elastic Load Balancing (ELB) service-linked role named `AWSServiceRoleForElasticLoadBalancing`. If any of these prerequisites are missing, the console provides guidance on how to configure your account to meet the prerequisites.

1. Navigate to the ROSA console.
2. Choose **Get started**.
3. On the **Verify ROSA prerequisites** page, select **I agree to share my contact information with Red Hat**.
4. Choose **Enable ROSA** .
5. Once the page has verified your service quotas meet ROSA prerequisites and the ELB service-linked role is created, open a new terminal session to create your first ROSA classic cluster using the ROSA CLI.

Step 2: Create a ROSA classic cluster with AWS STS and the ROSA CLI `auto` mode

You can create a ROSA classic cluster using AWS Security Token Service (AWS STS) and the `auto` mode that's provided in the ROSA CLI.

1. Create the required IAM account roles and policies.

```
rosa create account-roles --mode auto
```

> **Note**
>
> If your offline access token has expired, the ROSA CLI outputs an error message stating that your authorization token needs updated. For steps to troubleshoot, see the section called "Troubleshoot ROSA CLI expired offline access tokens".

2. Create a cluster with AWS STS using the defaults in the ROSA CLI's `auto` mode. When using the defaults, the latest stable OpenShift version is installed.

```
rosa create cluster --cluster-name <CLUSTER_NAME> --sts --mode auto
```

> **Note**
>
> When you specify `--mode auto`, the `rosa create cluster` command creates the cluster-specific operator IAM roles and the OIDC provider automatically. The operators use the OIDC provider to authenticate.

3. Check the status of your cluster.

```
rosa describe cluster -c <CLUSTER_NAME>
```

> **Note**
>
> If the provisioning process fails or the `State` field doesn’t change to a ready status after 40 minutes, see the section called “ ROSA cluster creation issues”.
> To contact AWS Support or Red Hat support for assistance, see the section called “Support for ROSA”.

4. Track the progress of the cluster creation by watching the OpenShift installer logs.

```
rosa logs install -c <CLUSTER_NAME> --watch
```

Step 3: Configure an identity provider and grant cluster access

ROSA includes a built-in OAuth server. After your cluster is created, you must configure OAuth to use an identity provider. You can then add users to your configured identity provider to grant them access to your cluster. You can grant these users `cluster-admin` or `dedicated-admin` permissions as required.

You can configure different identity provider types for your ROSA cluster. Supported types include GitHub, GitHub Enterprise, GitLab, Google, LDAP, OpenID Connect, and HTPasswd identity providers.

Important

The HTPasswd identity provider is included only to enable a single, static administrator user to be created. HTPasswd isn't supported as a general-use identity provider for ROSA.

The following procedure configures a GitHub identity provider as an example. For instructions on how to configure each of the supported identity provider types, see Configuring identity providers for AWS STS.

1. Navigate to github.com and log in to your GitHub account.
2. If you don't have a GitHub organization to use for identity provisioning for your cluster, create one. For more information, see the steps in the GitHub documentation.
3. Using the ROSA CLI's interactive mode, configure an identity provider for your cluster.

```
rosa create idp --cluster=<CLUSTER_NAME> --interactive
```

4. Follow the configuration prompts in the output to restrict cluster access to members of your GitHub organization.

```
I: Interactive mode enabled.
Any optional fields can be left empty and a default will be selected.
? Type of identity provider: github
? Identity provider name: github-1
? Restrict to members of: organizations
? GitHub organizations: <GITHUB_ORG_NAME>
? To use GitHub as an identity provider, you must first register the application:
  - Open the following URL:
    https://github.com/organizations/<GITHUB_ORG_NAME>/settings/
applications/new?oauth_application%5Bcallback_url%5D=https%3A%2F%2Foauth-
openshift.apps.<CLUSTER_NAME>/<RANDOM_STRING>.p1.openshiftapps.com%2Foauth2callback
%2Fgithub-1&oauth_application%5Bname%5D=<CLUSTER_NAME>&oauth_application
%5Burl%5D=https%3A%2F%2Fconsole-openshift-console.apps.<CLUSTER_NAME>/
<RANDOM_STRING>.p1.openshiftapps.com
  - Click on 'Register application'
...
```

5. Open the URL in the output, replacing `<GITHUB_ORG_NAME>` with the name of your GitHub organization.

6. On the GitHub web page, choose **Register application** to register a new OAuth application in your GitHub organization.
7. Use the information from the GitHub OAuth page to populate the remaining `rosa create idp` interactive prompts by running the following command. Replace `<GITHUB_CLIENT_ID>` and `<GITHUB_CLIENT_SECRET>` with the credentials from your GitHub OAuth application.

```
...
? Client ID: <GITHUB_CLIENT_ID>
? Client Secret: [? for help] <GITHUB_CLIENT_SECRET>
? GitHub Enterprise Hostname (optional):
? Mapping method: claim
I: Configuring IDP for cluster '<CLUSTER_NAME>'
I: Identity Provider 'github-1' has been created.
   It will take up to 1 minute for this configuration to be enabled.
   To add cluster administrators, see 'rosa grant user --help'.
   To login into the console, open https://console-openshift-
console.apps.<CLUSTER_NAME>.<RANDOM_STRING>.p1.openshiftapps.com and click on
 github-1.
```

Note

It might take approximately two minutes for the identity provider configuration to become active. If you configured a `cluster-admin` user, you can run `oc get pods -n openshift-authentication --watch` to watch the OAuth pods redeploy with the updated configuration.

8. Verify that the identity provider is configured correctly.

```
rosa list idps --cluster=<CLUSTER_NAME>
```

Step 4: Grant user access to a cluster

You can grant a user access to your cluster by adding them to the configured identity provider.

The following procedure adds a user to a GitHub organization that's configured for identity provisioning to the cluster.

1. Navigate to github.com and log in to your GitHub account.

2. Invite users that require cluster access to your GitHub organization. For more information, see Inviting users to join your organization in the GitHub documentation.

Step 5: Grant administrator permissions to a user

After you add a user to your configured identity provider, you can grant the user `cluster-admin` or `dedicated-admin` permissions for your cluster.

Configure `cluster-admin` permissions

1. Grant the `cluster-admin` permissions by running the following command. Replace <IDP_USER_NAME> and <CLUSTER_NAME> with your user and cluster name.

```
rosa grant user cluster-admin --user=<IDP_USER_NAME> --cluster=<CLUSTER_NAME>
```

2. Verify that the user is listed as a member of the `cluster-admins` group.

```
rosa list users --cluster=<CLUSTER_NAME>
```

Configure `dedicated-admin` permissions

1. Grant the `dedicated-admin` permissions by using the following command. Replace <IDP_USER_NAME> and <CLUSTER_NAME> with your user and cluster name by running the following command.

```
rosa grant user dedicated-admin --user=<IDP_USER_NAME> --cluster=<CLUSTER_NAME>
```

2. Verify that the user is listed as a member of the `cluster-admins` group.

```
rosa list users --cluster=<CLUSTER_NAME>
```

Step 6: Access a cluster through the web console

After you create a cluster administrator user or added a user to your configured identity provider, you can log in to your cluster through the Red Hat Hybrid Cloud Console.

1. Obtain the console URL for your cluster using the following command. Replace <CLUSTER_NAME> with the name of your cluster.

```
rosa describe cluster -c <CLUSTER_NAME> | grep Console
```

2. Navigate to the console URL in the output and log in.
   - If you created a `cluster-admin` user, log in using the provided credentials.
   - If you configured an identity provider for your cluster, choose the identity provider name in the **Log in with...** dialog and complete any authorization requests presented by your provider.

Step 7: Deploy an application from the Developer Catalog

From the Red Hat Hybrid Cloud Console, you can deploy a Developer Catalog test application and expose it with a route.

1. Navigate to Red Hat Hybrid Cloud Console and choose the cluster you want to deploy the app into.
2. On the cluster's page, choose **Open console**.
3. In the **Administrator** perspective, choose **Home** > **Projects** > **Create Project**.
4. Enter a name for your project and optionally add a **Display Name** and **Description**.
5. Choose **Create** to create the project.
6. Switch to the **Developer** perspective and choose **+Add**. Make sure that the selected project is the one that was just created.
7. In the **Developer Catalog** dialog, choose **All services**.
8. In the **Developer Catalog** page, choose **Languages** > **JavaScript** from the menu.
9. Choose **Node.js**, and then choose **Create Application** to open the **Create Source-to-Image Application** page.

> **Note**
>
> You might need to choose **Clear All Filters** to display the **Node.js** option.

10. In the **Git** section, choose **Try Sample**.
11. In the **Name** field, add a unique name.
12. Choose **Create**.

> **Note**
>
> The new application takes several minutes to deploy.

13. When the deployment is complete, choose the route URL for the application.

    A new tab in the browser opens with a message that's similar to the following.

    ```
    Welcome to your Node.js application on OpenShift
    ```

14. (Optional) Delete the application and clean up resources:

    a. In the **Administrator** perspective, choose **Home** > **Projects**.

    b. Open the action menu for your project and choose **Delete Project**.

Step 8: Revoke administrator permissions and user access

You can revoke `cluster-admin` or `dedicated-admin` permissions from a user by using the ROSA CLI.

To revoke access from a user, you must remove the user from your configured identity provider.

Revoke `cluster-admin` permissions from a user

1. Revoke the `cluster-admin` permissions using the following command. Replace <IDP_USER_NAME> and <CLUSTER_NAME> with your user and cluster name.

   ```
   rosa revoke user cluster-admin --user=<IDP_USER_NAME> --cluster=<CLUSTER_NAME>
   ```

2. Verify that the user isn't listed as a member of the `cluster-admins` group.

   ```
   rosa list users --cluster=<CLUSTER_NAME>
   ```

Revoke `dedicated-admin` permissions from a user

1. Revoke the `dedicated-admin` permissions by using the following command. Replace <IDP_USER_NAME> and <CLUSTER_NAME> with your user and cluster name.

```
rosa revoke user dedicated-admin --user=<IDP_USER_NAME> --cluster=<CLUSTER_NAME>
```

2. Verify that the user isn’t listed as a member of the `dedicated-admins` group.

```
rosa list users --cluster=<CLUSTER_NAME>
```

Revoke user access to a cluster

You can revoke cluster access for an identity provider user by removing them from the configured identity provider.

You can configure different types of identity providers for your cluster. The following procedure revokes cluster access for a member of a GitHub organization.

1. Navigate to github.com and log in to your GitHub account.
2. Remove the user from your GitHub organization. For more information, see Removing a member from your organization in the GitHub documentation.

Step 9: Delete a cluster and AWS STS resources

You can use the ROSA CLI to delete a cluster that uses AWS Security Token Service (AWS STS). You can also use the ROSA CLI to delete the IAM roles and OIDC provider created by ROSA. To delete the IAM policies created by ROSA, you can use the IAM console.

> ⚠ **Important**
>
> IAM roles and policies created by ROSA might be used by other ROSA clusters in the same account.

1. Delete the cluster and watch the logs. Replace `<CLUSTER_NAME>` with the name or ID of your cluster.

```
rosa delete cluster --cluster=<CLUSTER_NAME> --watch
```

Important

You must wait for the cluster to delete completely before you remove the IAM roles, policies, and OIDC provider. The account IAM roles are required to delete the resources created by the installer. The operator IAM roles are required to clean up the resources created by the OpenShift operators. The operators use the OIDC provider to authenticate.

2. Delete the OIDC provider that the cluster operators use to authenticate by running the following command.

   ```
   rosa delete oidc-provider -c <CLUSTER_ID> --mode auto
   ```

3. Delete the cluster-specific operator IAM roles.

   ```
   rosa delete operator-roles -c <CLUSTER_ID> --mode auto
   ```

4. Delete the account IAM roles using the following command. Replace `<PREFIX>` with the prefix of the account IAM roles to delete. If you specified a custom prefix when creating the account IAM roles, specify the default `ManagedOpenShift` prefix.

   ```
   rosa delete account-roles --prefix <PREFIX> --mode auto
   ```

5. Delete the IAM policies created by ROSA.

   a. Log in to the IAM console.

   b. On the left menu under **Access management**, choose **Policies**.

   c. Select the policy that you want to delete and choose **Actions** > **Delete**.

   d. Enter the policy name and choose **Delete**.

   e. Repeat this step to delete each of the IAM policies for the cluster.

Getting started with ROSA classic using the ROSA CLI in manual mode

The following sections describe how to get started with ROSA classic using AWS STS and the ROSA CLI. For more information about ROSA classic, see the section called “Deployment options”.

The ROSA CLI uses `auto` mode or `manual` mode to create the IAM resources that are required to provision a ROSA cluster. `auto` mode immediately creates the required IAM roles and policies and an OpenID Connect (OIDC) provider. `manual` mode outputs the AWS CLI commands that are needed to create the IAM resources. By using `manual` mode, you can review the generated AWS CLI commands before running them manually. You can also use `manual` to pass the commands to another administrator or group in your organization so they can create the resources.

The procedures in this document use the `manual` mode of the ROSA CLI to create the required IAM resources for ROSA classic. For more options to get started, see *Getting started with ROSA* .

Topics

Prerequisites

Before getting started, make sure you completed these actions:

- Install and configure the latest AWS CLI. For more information, see Installing or updating the latest version of the AWS CLI.
- Install and configure the latest ROSA CLI and OpenShift Container Platform CLI. For more information, see Getting started with the ROSA CLI.
- Service Quotas must have the required service quotas set for Amazon EC2, Amazon VPC, Amazon EBS, and Elastic Load Balancing that are needed to create and run a ROSA cluster. AWS or Red Hat may request service quota increases on your behalf as required for issue resolution. To view the required quotas, see Red Hat OpenShift Service on AWS endpoints and quotas in the *AWS General Reference*.
- To receive AWS support for ROSA, you must enable AWS Business, Enterprise On-Ramp, or Enterprise support plans. Red Hat may request AWS support on your behalf as required for issue

resolution. For more information, see the section called "Support for ROSA". To enable AWS Support, see the AWS Support page.

- If you're using AWS Organizations to manage the AWS accounts that host the ROSA service, the organization's service control policy (SCP) must be configured to allow Red Hat to perform policy actions that's listed in the SCP without restriction. For more information, see the the section called "AWS Organizations service control policy denies required AWS Marketplace permissions". For more information about SCPs, see Service control policies (SCPs).
- If deploying a ROSA cluster with AWS STS into an enabled AWS Region that's disabled by default, you must update the security token to version 2 for all the Regions in the AWS account with the following command.

  ```
  aws iam set-security-token-service-preferences --global-endpoint-token-version
   v2Token
  ```

  For more information about enabling Regions, see Managing AWS Regions in the *AWS General Reference*.

Step 1: Enable ROSA and configure prerequisites

To create a ROSA cluster, you must first enable the ROSA service in the AWS ROSA console. The AWS ROSA console verifies if your AWS account has the necessary AWS Marketplace permissions, service quotas, and the Elastic Load Balancing (ELB) service-linked role named `AWSServiceRoleForElasticLoadBalancing`. If any of these prerequisites are missing, the console provides guidance on how to configure your account to meet the prerequisites.

1. Navigate to the ROSA console.
2. Choose **Get started**.
3. On the **Verify ROSA prerequisites** page, select **I agree to share my contact information with Red Hat**.
4. Choose **Enable ROSA** .
5. Once the page has verified your service quotas meet ROSA prerequisites and the ELB service-linked role is created, open a new terminal session to create your first ROSA cluster using the ROSA CLI.

Step 2: Create a ROSA classic cluster with AWS STS and the ROSA CLI `manual` mode

You can create a ROSA classic cluster using AWS Security Token Service (AWS STS) and the `manual` mode that's provided in the ROSA CLI.

When you create a cluster, you can run `rosa create cluster --interactive` to customize your deployment with a series of interactive prompts. For more information, see Interactive cluster creation mode reference in the Red Hat documentation.

After the cluster is provisioned, a single command is provided in the output. Run this command to deploy further clusters that use the exact same custom configuration.

> **Note**
>
> AWS shared VPCs aren't currently supported for ROSA installations.

1. Create the required IAM account roles and policies.

   ```
   rosa create account-roles --mode manual
   ```

   > **Note**
   >
   > If your offline access token has expired, the ROSA CLI outputs an error message stating that your authorization token needs updated. For steps to troubleshoot, see the section called "Troubleshoot ROSA CLI expired offline access tokens".

2. Run the AWS CLI commands generated in the output to create the roles and policies.
3. Create a cluster with AWS STS in `--interactive` mode to specify any custom settings.

   ```
   rosa create cluster --interactive --sts
   ```

   > **Important**
   >
   > After you enable etcd encryption for the key values in etcd, you incur a performance overhead of approximately 20%. The overhead is a result of introducing this second

layer of encryption, in addition to the default Amazon EBS encryption that encrypts the etcd volumes.

4. To create the cluster-specific operator IAM roles, generate the operator policy JSON files in the current working directory and output the AWS CLI commands for review.

```
rosa create operator-roles --mode manual --cluster <CLUSTER_NAME|CLUSTER_ID>
```

5. Run the AWS CLI commands from the output.
6. Create the OpenID Connect (OIDC) provider the cluster operators use to authenticate.

```
rosa create oidc-provider --mode auto --cluster <CLUSTER_NAME|CLUSTER_ID>
```

7. Check the status of your cluster.

```
rosa describe cluster -c <CLUSTER_NAME>
```

> **Note**
>
> If the creation process fails or the `State` field doesn't change to a ready status after 40 minutes, see the section called " ROSA cluster creation issues".
> To contact AWS Support or Red Hat support for assistance, see the section called "Support for ROSA".

8. Track the progress of the cluster creation by watching the OpenShift installer logs.

```
rosa logs install -c <CLUSTER_NAME> --watch
```

Step 3: Configure an identity provider and grant cluster access

ROSA includes a built-in OAuth server. After your cluster is created, you must configure OAuth to use an identity provider. You can then add users to your configured identity provider to grant them access to your cluster. You can grant these users `cluster-admin` or `dedicated-admin` permissions as required.

You can configure different identity provider types for your cluster. Supported types include GitHub, GitHub Enterprise, GitLab, Google, LDAP, OpenID Connect, and HTPasswd identity providers.

Important

The HTPasswd identity provider is included only to enable a single, static administrator user to be created. HTPasswd isn't supported as a general-use identity provider for ROSA.

The following procedure configures a GitHub identity provider as an example. For instructions on how to configure each of the supported identity provider types, see Configuring identity providers for AWS STS.

1. Navigate to github.com and log in to your GitHub account.
2. If you don't have a GitHub organization to use for identity provisioning for your ROSA cluster, create one. For more information, see the steps in the GitHub documentation.
3. Using the ROSA CLI's interactive mode, configure an identity provider for your cluster.

```
rosa create idp --cluster=<CLUSTER_NAME> --interactive
```

4. Follow the configuration prompts in the output to restrict cluster access to members of your GitHub organization.

```
I: Interactive mode enabled.
Any optional fields can be left empty and a default will be selected.
? Type of identity provider: github
? Identity provider name: github-1
? Restrict to members of: organizations
? GitHub organizations: <GITHUB_ORG_NAME>
? To use GitHub as an identity provider, you must first register the application:
  - Open the following URL:
    https://github.com/organizations/<GITHUB_ORG_NAME>/settings/
applications/new?oauth_application%5Bcallback_url%5D=https%3A%2F%2Foauth-
openshift.apps.<CLUSTER_NAME>/<RANDOM_STRING>.p1.openshiftapps.com%2Foauth2callback
%2Fgithub-1&oauth_application%5Bname%5D=<CLUSTER_NAME>&oauth_application
%5Burl%5D=https%3A%2F%2Fconsole-openshift-console.apps.<CLUSTER_NAME>/
<RANDOM_STRING>.p1.openshiftapps.com
  - Click on 'Register application'
...
```

5. Open the URL in the output with the following command. Replace `<GITHUB_ORG_NAME>` with the name of your GitHub organization.

6. On the GitHub web page, choose **Register application** to register a new OAuth application in your GitHub organization.
7. Use the information from the GitHub OAuth page to populate the remaining `rosa create idp` interactive prompts using the following command. Replace <GITHUB_CLIENT_ID> and <GITHUB_CLIENT_SECRET> with the credentials from your GitHub OAuth application.

```
...
? Client ID: <GITHUB_CLIENT_ID>
? Client Secret: [? for help] <GITHUB_CLIENT_SECRET>
? GitHub Enterprise Hostname (optional):
? Mapping method: claim
I: Configuring IDP for cluster '<CLUSTER_NAME>'
I: Identity Provider 'github-1' has been created.
   It will take up to 1 minute for this configuration to be enabled.
   To add cluster administrators, see 'rosa grant user --help'.
   To login into the console, open https://console-openshift-
console.apps.<CLUSTER_NAME>.<RANDOM_STRING>.p1.openshiftapps.com and click on
 github-1.
```

> **Note**
>
> It might take about two minutes for the identity provider configuration to become active. If you configured a `cluster-admin` user, you can run the `oc get pods -n openshift-authentication --watch` command to watch the OAuth pods redeploy with the updated configuration.

8. Verify the identity provider has been configured correctly using the following command.

```
rosa list idps --cluster=<CLUSTER_NAME>
```

Step 4: Grant user access to a cluster

You can grant a user access to your cluster by adding them to the configured identity provider.

The following procedure adds a user to a GitHub organization that's configured for identity provisioning to the cluster.

1. Navigate to github.com and log in to your GitHub account.

2. Invite users that require cluster access to your GitHub organization. For more information, see Inviting users to join your organization in the documentation on Github.

Step 5: Grant administrator permissions to a user

After you add a user to your configured identity provider, you can grant the user `cluster-admin` or `dedicated-admin` permissions for your cluster.

Configure `cluster-admin` permissions

1. Grant the `cluster-admin` permissions using the following command. Replace <IDP_USER_NAME> and <CLUSTER_NAME> with your user and cluster name.

```
rosa grant user cluster-admin --user=<IDP_USER_NAME> --cluster=<CLUSTER_NAME>
```

2. Verify that the user is listed as a member of the `cluster-admins` group.

```
rosa list users --cluster=<CLUSTER_NAME>
```

Configure `dedicated-admin` permissions

1. Grant the `dedicated-admin` permissions using the following command. Replace <IDP_USER_NAME> and <CLUSTER_NAME> with your user and cluster name.

```
rosa grant user dedicated-admin --user=<IDP_USER_NAME> --cluster=<CLUSTER_NAME>
```

2. Verify that the user is listed as a member of the `cluster-admins` group.

```
rosa list users --cluster=<CLUSTER_NAME>
```

Step 6: Access a cluster through the web console

After you create a cluster administrator user or add a user to your configured identity provider, you can log in to your cluster through the Red Hat Hybrid Cloud Console.

1. Obtain the console URL for your cluster by using the following command. Replace <CLUSTER_NAME> with the name of your cluster.

```
rosa describe cluster -c <CLUSTER_NAME> | grep Console
```

2. Navigate to the console URL in the output and log in.
   - If you create a `cluster-admin` user, log in using the provided credentials.
   - If you configure an identity provider for your cluster, choose the identity provider name in the **Log in with…** dialog and complete any authorization requests that are presented by your provider.

Step 7: Deploy an application from the Developer Catalog

From the Red Hat Hybrid Cloud Console, you can deploy a Developer Catalog test application and expose it with a route.

1. Navigate to Red Hat Hybrid Cloud Console and choose the cluster you want to deploy the app into.
2. On the cluster's page, choose **Open console**.
3. In the **Administrator** perspective, choose **Home** > **Projects** > **Create Project**.
4. Enter a name for your project and optionally add a **Display Name** and **Description**.
5. Choose **Create** to create the project.
6. Switch to the **Developer** perspective and choose **+Add**. Make sure that the selected project is the one that was just created.
7. In the **Developer Catalog** dialog, choose **All services**.
8. In the **Developer Catalog** page, choose **Languages** > **JavaScript** from the menu.
9. Choose **Node.js**, and then choose **Create Application** to open the **Create Source-to-Image Application** page.

> **Note**
>
> You might need to choose **Clear All Filters** to display the **Node.js** option.

10. In the **Git** section, choose **Try Sample**.
11. In the **Name** field, add a unique name.
12. Choose **Create**.

> **Note**
>
> The new application takes several minutes to deploy.

13. When the deployment is complete, choose the route URL for the application.

    A new tab in the browser opens with a message similar to the following.

```
Welcome to your Node.js application on OpenShift
```

14. (Optional) Delete the application and clean up resources.

    a. In the **Administrator** perspective, choose **Home** > **Projects**.

    b. Open the action menu for your project and choose **Delete Project**.

Step 8: Revoke administrator permissions and user access

You can revoke `cluster-admin` or `dedicated-admin` permissions from a user by using the ROSA CLI.

To revoke access from a user, you must remove the user from your configured identity provider.

Revoke `cluster-admin` permissions from a user

1. Revoke the `cluster-admin` permission by using the following command. Replace <IDP_USER_NAME> and <CLUSTER_NAME> with your user and cluster name.

```
rosa revoke user cluster-admin --user=<IDP_USER_NAME> --cluster=<CLUSTER_NAME>
```

2. Verify that the user isn't listed as a member of the `cluster-admins` group.

```
rosa list users --cluster=<CLUSTER_NAME>
```

Revoke `dedicated-admin` permissions from a user

1. Revoke the `dedicated-admin` permission using the following command. Replace <IDP_USER_NAME> and <CLUSTER_NAME> with your user and cluster name.

```
rosa revoke user dedicated-admin --user=<IDP_USER_NAME> --cluster=<CLUSTER_NAME>
```

2. Verify that the user isn't listed as a member of the `dedicated-admins` group.

    ```
    rosa list users --cluster=<CLUSTER_NAME>
    ```

Revoke user access to a cluster

You can revoke cluster access for an identity provider user by removing them from the configured identity provider.

You can configure different types of identity providers for your cluster. The following procedure revokes cluster access for a member of a GitHub organization.

1. Navigate to github.com and log in to your GitHub account.
2. Remove the user from your GitHub organization. For more information, see Removing a member from your organization in the GitHub documentation.

Step 9: Delete a cluster and AWS STS resources

You can use the ROSA CLI to delete a cluster that uses AWS Security Token Service (AWS STS). You can also use the ROSA CLI to delete the IAM roles and OIDC provider created by ROSA. To delete the IAM policies created by ROSA, you can use the IAM console.

> ⚠ **Important**
>
> IAM roles and policies created by ROSA might be used by other ROSA clusters in the same account.

1. Delete the cluster and watch the logs. Replace `<CLUSTER_NAME>` with the name or ID of your cluster.

    ```
    rosa delete cluster --cluster=<CLUSTER_NAME> --watch
    ```

> **Important**
>
> You must wait for the cluster to delete completely before you remove the IAM roles, policies, and OIDC provider. The account IAM roles are required to delete the resources created by the installer. The operator IAM roles are required to clean up the resources created by the OpenShift operators. The operators use the OIDC provider to authenticate.

2. Delete the OIDC provider that the cluster operators use to authenticate by running the following command.

```
rosa delete oidc-provider -c <CLUSTER_ID> --mode auto
```

3. Delete the cluster-specific operator IAM roles.

```
rosa delete operator-roles -c <CLUSTER_ID> --mode auto
```

4. Delete the account IAM roles using the following command. Replace `<PREFIX>` with the prefix of the account IAM roles to delete. If you specified a custom prefix when creating the account IAM roles, specify the default `ManagedOpenShift` prefix.

```
rosa delete account-roles --prefix <PREFIX> --mode auto
```

5. Delete the IAM policies created by ROSA.

   a. Log in to the IAM console.

   b. On the left menu under **Access management**, choose **Policies**.

   c. Select the policy that you want to delete and choose **Actions** > **Delete**.

   d. Enter the policy name and choose **Delete**.

   e. Repeat this step to delete each of the IAM policies for the cluster.

Getting started with ROSA classic using AWS PrivateLink

ROSA classic clusters can be deployed in a few different ways: public, private, or private with AWS PrivateLink. For more information about ROSA classic, see the section called "Deployment options". For both public and private cluster configurations, the OpenShift cluster has access to the internet, and privacy is set on the application workloads at the application layer.

If you require both the cluster and the application workloads to be private, you can configure AWS PrivateLink with ROSA classic. AWS PrivateLink is a highly available, scalable technology that ROSA uses to create a private connection between the ROSA service and cluster resources in the AWS customer account. With AWS PrivateLink, the Red Hat site reliability engineering (SRE) team can access the cluster for support and remediation purposes by using a private subnet connected to the cluster's AWS PrivateLink endpoint.

For more information about AWS PrivateLink, see What is AWS PrivateLink?

Topics

Prerequisites

Before getting started, make sure you completed these actions:

- Install and configure the latest AWS CLI. For more information, see Installing or updating the latest version of the AWS CLI.
- Install and configure the latest ROSA CLI and OpenShift Container Platform CLI. For more information, see Getting started with the ROSA CLI.
- Service Quotas must have the required service quotas set for Amazon EC2, Amazon VPC, Amazon EBS, and Elastic Load Balancing that are needed to create and run a ROSA cluster. AWS or Red Hat may request service quota increases on your behalf as required for issue resolution. To view

the required quotas, see Red Hat OpenShift Service on AWS endpoints and quotas in the *AWS General Reference*.

- To receive AWS support for ROSA, you must enable AWS Business, Enterprise On-Ramp, or Enterprise support plans. Red Hat may request AWS support on your behalf as required for issue resolution. For more information, see the section called "Support for ROSA". To enable AWS Support, see the AWS Support page.
- If you're using AWS Organizations to manage the AWS accounts that host the ROSA service, the organization's service control policy (SCP) must be configured to allow Red Hat to perform policy actions that's listed in the SCP without restriction. For more information, see the the section called "AWS Organizations service control policy denies required AWS Marketplace permissions". For more information about SCPs, see Service control policies (SCPs).
- If deploying a ROSA cluster with AWS STS into an enabled AWS Region that's disabled by default, you must update the security token to version 2 for all the Regions in the AWS account with the following command.

  ```
  aws iam set-security-token-service-preferences --global-endpoint-token-version
   v2Token
  ```

  For more information about enabling Regions, see Managing AWS Regions in the *AWS General Reference*.

Step 1: Enable ROSA and configure prerequisites

To create a ROSA cluster, you must first enable the ROSA service in the AWS ROSA console. The AWS ROSA console verifies if your AWS account has the necessary AWS Marketplace permissions, service quotas, and the Elastic Load Balancing (ELB) service-linked role named `AWSServiceRoleForElasticLoadBalancing`. If any of these prerequisites are missing, the console provides guidance on how to configure your account to meet the prerequisites.

1. Navigate to the ROSA console.
2. Choose **Get started**.
3. On the **Verify ROSA prerequisites** page, select **I agree to share my contact information with Red Hat**.
4. Choose **Enable ROSA** .

5. Once the page has verified your service quotas meet ROSA prerequisites and the ELB service-linked role is created, open a new terminal session to create your first ROSA cluster using the ROSA CLI.

Step 2: Create Amazon VPC architecture for the cluster

To create a ROSA cluster that uses AWS PrivateLink, you must first configure your own Amazon VPC architecture to deploy your solution into. ROSA requires that customers configure at least one public and private subnet per Availability Zone used to create clusters. For single-AZ clusters, only use Availability Zone is used. For multi-AZ clusters, three Availability Zones are needed.

> **Important**
>
> If Amazon VPC requirements are not met, cluster creation fails.

The following procedure uses the AWS CLI to create both a public and private subnet into a single Availability Zone for a Single-AZ cluster. All cluster resources are in the private subnet. The public subnet routes outbound traffic by using a NAT gateway to the internet.

This example uses the CIDR block `10.0.0.0/16` for the Amazon VPC. However, you can choose a different CIDR block. For more information, see VPC sizing.

1. Set an environment variable for the cluster name by running the following command.

   ```
   ROSA_CLUSTER_NAME=rosa-privatelink
   ```

2. Create a VPC with a `10.0.0.0/16` CIDR block.

   ```
   aws ec2 create-vpc --cidr-block 10.0.0.0/16 --query Vpc.VpcId --output text
   ```

   The preceding command returns the ID of the new VPC. The following is an example output.

   ```
   vpc-0410832ee325aafea
   ```

3. Using the VPC ID from the previous step, tag the VPC using the `ROSA_CLUSTER_NAME` variable.

   ```
   aws ec2 create-tags --resources <VPC_ID_VALUE> --tags Key=Name,Value=
   $ROSA_CLUSTER_NAME
   ```

4. Enable DNS hostname support on the VPC.

```
aws ec2 modify-vpc-attribute --vpc-id <VPC_ID_VALUE> --enable-dns-hostnames
```

5. Create a public subnet in the VPC with a `10.0.1.0/24` CIDR block, specifying the Availability Zone where the resource should be created.

> **Important**
>
> When creating subnets, make sure that subnets are created to an Availability Zone that has ROSA instance types available. If you don’t choose a specific Availability Zone, the subnet is created in any one of the Availability Zones in the AWS Region that you specify. To specify a specific Availability Zone, use the `--availability zone` argument in the `create-subnet` command. You can use the `rosa list instance-types` command to list all ROSA instance types available. To check if an instance type is available for a given Availability Zone, use the following command.
>
> ```
> aws ec2 describe-instance-type-offerings --location-type availability-zone --
> filters Name=location,Values=<availability_zone> --region <region> --output
>  text | egrep "<instance_type>"
> ```

> **Important**
>
> ROSA requires that customers configure at least one public and private subnet per Availability Zone used to create clusters. For single-AZ clusters, only one Availability Zone is needed. For multi-AZ clusters, three Availability Zones are needed. If these requirements are not met, cluster creation fails.

```
aws ec2 create-subnet --vpc-id <VPC_ID_VALUE> --cidr-block 10.0.1.0/24 --
availability-zone <AZ_NAME> --query Subnet.SubnetId --output text
```

The preceding command returns the ID of the new subnet. The following is an example output.

```
subnet-0b6a7e8cbc8b75920
```

6. Using the subnet ID from the previous step, tag the subnet using the `ROSA_CLUSTER_NAME-public` variable.

   ```
   aws ec2 create-tags --resources <PUBLIC_SUBNET_ID> --tags Key=Name,Value=
   $ROSA_CLUSTER_NAME-public
   ```

7. Create a private subnet in the VPC with a `10.0.0.0/24` CIDR block, specifying the same Availability Zone that the public subnet deployed into.

   ```
   aws ec2 create-subnet --vpc-id <VPC_ID_VALUE> --cidr-block 10.0.0.0/24 --
   availability-zone <AZ_NAME> --query Subnet.SubnetId --output text
   ```

   The preceding command returns the ID of the new subnet. The following is an example output.

   ```
   subnet-0b6a7e8cbc8b75920
   ```

8. Using the subnet ID from the previous step, tag the subnet using the `ROSA_CLUSTER_NAME-private` variable.

   ```
   aws ec2 create-tags --resources <PRIVATE_SUBNET_ID> --tags Key=Name,Value=
   $ROSA_CLUSTER_NAME-private
   ```

9. Create an internet gateway for outbound traffic and attach it to the VPC.

   ```
   aws ec2 create-internet-gateway --query InternetGateway.InternetGatewayId --output
    text

   aws ec2 attach-internet-gateway --vpc-id <VPC_ID_VALUE> --internet-gateway-id
    <IG_ID_VALUE>
   ```

10. Tag the internet gateway with the `ROSA_CLUSTER_NAME` variable.

    ```
    aws ec2 create-tags --resources <IG_ID_VALUE> --tags Key=Name,Value=
    $ROSA_CLUSTER_NAME
    ```

11. Create a route table for outbound traffic, associate it to the public subnet, and configure traffic to route to the internet gateway.

    ```
    aws ec2 create-route-table --vpc-id <VPC_ID_VALUE> --query RouteTable.RouteTableId --
    output text
    ```

```
aws ec2 associate-route-table --subnet-id <PUBLIC_SUBNET_ID> --route-table-id
 <PUBLIC_RT_ID>

aws ec2 create-route --route-table-id <PUBLIC_RT_ID> --destination-cidr-block
 0.0.0.0/0 --gateway-id <IG_ID_VALUE>
```

12. Tag the public route table with the `ROSA_CLUSTER_NAME` variable and verify that the route table was properly configured.

```
aws ec2 create-tags --resources <PUBLIC_RT_ID> --tags Key=Name,Value=
$ROSA_CLUSTER_NAME

aws ec2 describe-route-tables --route-table-id <PUBLIC_RT_ID>
```

13. Create a NAT gateway in the public subnet with an elastic IP address to enable traffic to the private subnet.

```
aws ec2 allocate-address --domain vpc --query AllocationId --output text

aws ec2 create-nat-gateway --subnet-id <PUBLIC_SUBNET_ID> --allocation-id
 <EIP_ADDRESS>  --query NatGateway.NatGatewayId --output text
```

14. Tag the NAT gateway and elastic IP address with the `$ROSA_CLUSTER_NAME` variable.

```
aws ec2 create-tags --resources <EIP_ADDRESS>  --resources <NAT_GATEWAY_ID> --tags
 Key=Name,Value=$ROSA_CLUSTER_NAME
```

15. Create a route table for private subnet traffic, associate it to the private subnet, and configure traffic to route to the NAT gateway.

```
aws ec2 create-route-table --vpc-id <VPC_ID_VALUE> --query RouteTable.RouteTableId --
output text

aws ec2 associate-route-table --subnet-id <PRIVATE_SUBNET_ID> --route-table-id
 <PRIVATE_RT_ID>

aws ec2 create-route --route-table-id <PRIVATE_RT_ID> --destination-cidr-block
 0.0.0.0/0 --gateway-id <NAT_GATEWAY_ID>
```

16. Tag the private route table and elastic IP address with the `$ROSA_CLUSTER_NAME-private` variable.

```
aws ec2 create-tags --resources <PRIVATE_RT_ID> <EIP_ADDRESS> --tags Key=Name,Value=
$ROSA_CLUSTER_NAME-private
```

Step 3: Create a cluster with AWS PrivateLink

You can use AWS PrivateLink and the ROSA CLI to create a cluster with a single Availability Zone (Single-AZ) or multiple Availability Zones (Multi-AZ). In either case, your machine's CIDR value must match your VPC's CIDR value.

The following procedure uses the `rosa create cluster` command to create a Single-AZ ROSA cluster. To create a Multi-AZ cluster, specify `multi-az` in the command and the private subnet IDs for each private subnet you want you to deploy to.

> **Note**
>
> If you use a firewall, you must configure it so that ROSA can access the sites that it requires to function.
> For more information, see AWS firewall prerequisites in the Red Hat OpenShift documentation.

1. Create a Single-AZ cluster by running the following command.

```
rosa create cluster --private-link --cluster-name=<CLUSTER_NAME> --machine-
cidr=10.0.0.0/16 --subnet-ids=<PRIVATE_SUBNET_ID>
```

> **Note**
>
> To create a cluster that uses AWS PrivateLink with AWS Security Token Service (AWS STS) short-lived credentials, append `--sts --mode auto` or `--sts --mode manual` to the end of the `rosa create cluster` command.

2. Create the cluster operator IAM roles by following the interactive prompts.

```
rosa create operator-roles --interactive -c <CLUSTER_NAME>
```

3. Create the OpenID Connect (OIDC) provider the cluster operators use to authenticate.

```
rosa create oidc-provider --interactive -c <CLUSTER_NAME>
```

4. Check the status of your cluster.

   ```
   rosa describe cluster -c <CLUSTER_NAME>
   ```

   Example

   > **Note**
   >
   > It may take up to 40 minutes for the cluster `State` field to show the `ready` status. If provisioning fails or doesn't show as `ready` after 40 minutes, see the section called " ROSA cluster creation issues".

   To contact AWS Support or Red Hat support for assistance, see the section called "Support for ROSA".

5. Track the progress of the cluster creation by watching the OpenShift installer logs.

   ```
   rosa logs install -c <CLUSTER_NAME> --watch
   ```

Step 4: Configure AWS PrivateLink DNS forwarding

Clusters that use AWS PrivateLink create a public hosted zone and a private hosted zone in Route 53. Records within the Route 53 private hosted zone are resolvable only from within the VPC that it's assigned to.

The Let's Encrypt DNS-01 validation requires a public zone so that valid and publicly trusted certificates can be issued for the domain. The validation records are deleted after Let's Encrypt validation is complete. The zone is still required for issuing and renewing these certificates, which are typically required every 60 days. Although these zones usually appear empty, a public zone serves a critical role in the validation process.

For more information about AWS private hosted zones, see Working with private zones. For more information about public hosted zones, see Working with public hosted zones.

Configure a Route 53 Resolver inbound endpoint

To allow for records such as `api.<cluster_domain>` and `*.apps.<cluster_domain>` to resolve outside of the VPC, configure a Route 53 Resolver inbound endpoint.

1. Open the Route 53 console.
2. In the navigation pane under **Resolver**, choose **Inbound endpoints**.
3. Choose **Configure endpoints**.
4. In the upper right, use the AWS Region selector to choose the AWS Region that contains the VPC used for the cluster.
5. Under **Basic configuration**, choose **Inbound only** and then choose **Next**.
6. On the **Configure inbound endpoint** page, complete the **General settings for inbound endpoint** section. Under **Security group for this endpoint**, choose a security group that allows inbound UDP and TCP traffic from the remote network on destination port 53.
7. In the **IP address** section, choose the Availability Zones and private subnets that were used when creating the cluster and choose **Next**.
8. (Optional) Complete the **Tags** section.
9. Choose **Submit**.

Configure DNS forwarding for the cluster

After the Route 53 Resolver internal endpoint is associated and operational, configure DNS forwarding so DNS queries can be handled by the designated servers on your network.

1. Configure your corporate network to forward DNS queries to those IP addresses for the top-level domain, such as `drow-pl-01.htno.p1.openshiftapps.com`.
2. If you’re forwarding DNS queries from one VPC to another VPC, follow the instructions in Managing forwarding rules.
3. If you’re configuring your remote network DNS server, see your specific DNS server documentation to configure selective DNS forwarding for the installed cluster domain.

Step 5: Configure an identity provider and grant cluster access

ROSA includes a built-in OAuth server. After your ROSA cluster is created, you must configure OAuth to use an identity provider. You can then add users to your configured identity provider to grant them access to your cluster. You can grant these users `cluster-admin` or `dedicated-admin` permissions as required.

You can configure different identity provider types for your cluster. The supported types include GitHub, GitHub Enterprise, GitLab, Google, LDAP, OpenID Connect, and HTPasswd identity providers.

> **Important**
>
> The HTPasswd identity provider is included only to enable a single, static administrator user to be created. HTPasswd isn't supported as a general-use identity provider for ROSA.

The following procedure configures a GitHub identity provider as an example. For instructions on how to configure each of the supported identity provider types, see Configuring identity providers for AWS STS.

1. Navigate to github.com and log in to your GitHub account.
2. If you don't have a GitHub organization to use for identity provisioning for your ROSA cluster, create one. For more information, see the steps in the GitHub documentation.
3. Using the ROSA CLI's interactive mode, configure an identity provider for your cluster by running the following command.

```
rosa create idp --cluster=<CLUSTER_NAME> --interactive
```

4. Follow the configuration prompts in the output to restrict cluster access to members of your GitHub organization.

```
I: Interactive mode enabled.
Any optional fields can be left empty and a default will be selected.
? Type of identity provider: github
? Identity provider name: github-1
? Restrict to members of: organizations
? GitHub organizations: <GITHUB_ORG_NAME>
? To use GitHub as an identity provider, you must first register the application:
  - Open the following URL:
    https://github.com/organizations/<GITHUB_ORG_NAME>/settings/
applications/new?oauth_application%5Bcallback_url%5D=https%3A%2F%2Foauth-
openshift.apps.<CLUSTER_NAME>/<RANDOM_STRING>.p1.openshiftapps.com%2Foauth2callback
%2Fgithub-1&oauth_application%5Bname%5D=<CLUSTER_NAME>&oauth_application
```

```
%5Burl%5D=https%3A%2F%2Fconsole-openshift-console.apps.<CLUSTER_NAME>/
<RANDOM_STRING>.p1.openshiftapps.com
  - Click on 'Register application'
...
```

5. Open the URL in the output, replacing `<GITHUB_ORG_NAME>` with the name of your GitHub organization.

6. On the GitHub web page, choose **Register application** to register a new OAuth application in your GitHub organization.

7. Use the information from the GitHub OAuth page to populate the remaining `rosa create idp` interactive prompts, replacing `<GITHUB_CLIENT_ID>` and `<GITHUB_CLIENT_SECRET>` with the credentials from your GitHub OAuth application.

```
...
? Client ID: <GITHUB_CLIENT_ID>
? Client Secret: [? for help] <GITHUB_CLIENT_SECRET>
? GitHub Enterprise Hostname (optional):
? Mapping method: claim
I: Configuring IDP for cluster '<CLUSTER_NAME>'
I: Identity Provider 'github-1' has been created.
   It will take up to 1 minute for this configuration to be enabled.
   To add cluster administrators, see 'rosa grant user --help'.
   To login into the console, open https://console-openshift-
console.apps.<CLUSTER_NAME>.<RANDOM_STRING>.p1.openshiftapps.com and click on
 github-1.
```

Note

It might take around two minutes for the identity provider configuration to become active. If you configured a `cluster-admin` user, you can run the `oc get pods -n openshift-authentication --watch` command to watch the OAuth pods redeploy with the updated configuration.

8. Verify the identity provider has been configured correctly.

```
rosa list idps --cluster=<CLUSTER_NAME>
```

Step 6: Grant user access to a cluster

You can grant a user access to your cluster by adding them to the configured identity provider.

The following procedure adds a user to a GitHub organization that’s configured for identity provisioning to the cluster.

1. Navigate to github.com and log in to your GitHub account.
2. Invite users that require cluster access to your GitHub organization. For more information, see Inviting users to join your organization in the GitHub documentation.

Step 7: Grant administrator permissions to a user

After you added a user to your configured identity provider, you can grant the user `cluster-admin` or `dedicated-admin` permissions for your cluster.

Configure `cluster-admin` permissions

1. Grant the `cluster-admin` permissions using the following command. Replace `<IDP_USER_NAME>` and `<CLUSTER_NAME>` with your user and cluster name.

```
rosa grant user cluster-admin --user=<IDP_USER_NAME> --cluster=<CLUSTER_NAME>
```

2. Verify the user is listed as a member of the `cluster-admins` group.

```
rosa list users --cluster=<CLUSTER_NAME>
```

Configure `dedicated-admin` permissions

1. Grant the `dedicated-admin` permissions with the following command. Replace `<IDP_USER_NAME>` and `<CLUSTER_NAME>` with your user and cluster name.

```
rosa grant user dedicated-admin --user=<IDP_USER_NAME> --cluster=<CLUSTER_NAME>
```

2. Verify the user is listed as a member of the `cluster-admins` group.

```
rosa list users --cluster=<CLUSTER_NAME>
```

Step 8: Access a cluster through the web console

After you created a cluster administrator user or added a user to your configured identity provider, you can log in to your cluster through the Red Hat Hybrid Cloud Console.

1. Obtain the console URL for your cluster using the following command. Replace `<CLUSTER_NAME>` with the name of your cluster.

```
rosa describe cluster -c <CLUSTER_NAME> | grep Console
```

2. Navigate to the console URL in the output and log in.
   - If you created a `cluster-admin` user, log in using the provided credentials.
   - If you configured an identity provider for your cluster, choose the identity provider name in the **Log in with...** dialog and complete any authorization requests presented by your provider.

Step 9: Deploy an application from the Developer Catalog

From the Red Hat Hybrid Cloud Console, you can deploy a Developer Catalog test application and expose it with a route.

1. Navigate to Red Hat Hybrid Cloud Console and choose the cluster that you want to deploy the app into.
2. On the cluster's page, choose **Open console**.
3. In the **Administrator** perspective, choose **Home** > **Projects** > **Create Project**.
4. Enter a name for your project and optionally add a **Display Name** and **Description**.
5. Choose **Create** to create the project.
6. Switch to the **Developer** perspective and choose **+Add**. Make sure that the selected project is the one that was just created.
7. In the **Developer Catalog** dialog, choose **All services**.
8. In the **Developer Catalog** page, choose **Languages** > **JavaScript** from the menu.
9. Choose **Node.js**, and then choose **Create Application** to open the **Create Source-to-Image Application** page.

> **Note**
>
> You might need to choose **Clear All Filters** to display the **Node.js** option.

10. In the **Git** section, choose **Try Sample**.

11. In the **Name** field, add a unique name.

12. Choose **Create**.

> **Note**
>
> The new application takes several minutes to deploy.

13. When the deployment is complete, choose the route URL for the application.

    A new tab in the browser opens with a message that's similar to the following.

```
Welcome to your Node.js application on OpenShift
```

14. (Optional) Delete the application and clean up resources.

    a. In the **Administrator** perspective, choose **Home** > **Projects**.

    b. Open the action menu for your project and choose **Delete Project**.

Step 10: Revoke administrator permissions and user access

You can revoke `cluster-admin` or `dedicated-admin` permissions from a user by using the ROSA CLI.

To revoke access from a user, you must remove the user from your configured identity provider.

Revoke `cluster-admin` permissions from a user

1. Revoke the `cluster-admin` permissions using the following command. Replace <IDP_USER_NAME> and <CLUSTER_NAME> with your user and cluster name.

```
rosa revoke user cluster-admin --user=<IDP_USER_NAME> --cluster=<CLUSTER_NAME>
```

2. Verify that the user isn't listed as a member of the `cluster-admins` group.

```
rosa list users --cluster=<CLUSTER_NAME>
```

Revoke `dedicated-admin` permissions from a user

1. Revoke the `dedicated-admin` permissions using the following command. Replace `<IDP_USER_NAME>` and `<CLUSTER_NAME>` with your user and cluster name.

   ```
   rosa revoke user dedicated-admin --user=<IDP_USER_NAME> --cluster=<CLUSTER_NAME>
   ```

2. Verify that the user isn't listed as a member of the `dedicated-admins` group.

   ```
   rosa list users --cluster=<CLUSTER_NAME>
   ```

Revoke user access to a cluster

You can revoke cluster access for an identity provider user by removing them from the configured identity provider.

You can configure different types of identity providers for your cluster. The following procedure revokes cluster access for a member of a GitHub organization.

1. Navigate to github.com and log in to your GitHub account.
2. Remove the user from your GitHub organization. For more information, see Removing a member from your organization in the GitHub documentation.

Step 11: Delete a cluster and AWS STS resources

You can use the ROSA CLI to delete a cluster that uses AWS Security Token Service (AWS STS). You can also use the ROSA CLI to delete the IAM roles and OIDC provider created by ROSA. To delete the IAM policies created by ROSA, you can use the IAM console.

> ⚠ **Important**
>
> IAM roles and policies created by ROSA might be used by other ROSA clusters in the same account.

1. Delete the cluster and watch the logs. Replace `<CLUSTER_NAME>` with the name or ID of your cluster.

```
rosa delete cluster --cluster=<CLUSTER_NAME> --watch
```

> ⚠ **Important**
>
> You must wait for the cluster to delete completely before you remove the IAM roles, policies, and OIDC provider. The account IAM roles are required to delete the resources created by the installer. The operator IAM roles are required to clean up the resources created by the OpenShift operators. The operators use the OIDC provider to authenticate.

2. Delete the OIDC provider that the cluster operators use to authenticate by running the following command.

```
rosa delete oidc-provider -c <CLUSTER_ID> --mode auto
```

3. Delete the cluster-specific operator IAM roles.

```
rosa delete operator-roles -c <CLUSTER_ID> --mode auto
```

4. Delete the account IAM roles using the following command. Replace `<PREFIX>` with the prefix of the account IAM roles to delete. If you specified a custom prefix when creating the account IAM roles, specify the default `ManagedOpenShift` prefix.

```
rosa delete account-roles --prefix <PREFIX> --mode auto
```

5. Delete the IAM policies created by ROSA.

   a. Log in to the IAM console.

   b. On the left menu under **Access management**, choose **Policies**.

   c. Select the policy that you want to delete and choose **Actions** > **Delete**.

   d. Enter the policy name and choose **Delete**.

   e. Repeat this step to delete each of the IAM policies for the cluster.

Security in ROSA

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security of the cloud and security in the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS Compliance Programs. To learn about the compliance programs that apply to ROSA, see AWS services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using ROSA. It shows you how to configure ROSA to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your ROSA resources.

Contents

Data protection in ROSA

The the section called "Responsibilities" documentation and AWS shared responsibility model define data protection in ROSA. AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. Red Hat is responsible for protecting the cluster infrastructure and underlying service platform. The customer is responsible for maintaining control over content that is hosted on this infrastructure. This content includes the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual users with AWS Identity and Access Management (IAM). That way each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We require TLS 1.2 and recommend TLS 1.3.
- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.
- If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-2.

We strongly recommend that you never put sensitive identifying information, such as your customers' account numbers, into free-form fields such as a **Name** field. This includes when you work with ROSA or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into ROSA or other services might get picked up for inclusion in diagnostic logs. When you provide a URL to an external server, don’t include credentials information in the URL to validate your request to that server.

Topics

Protecting data using encryption

Data protection refers to protecting data while in transit (as it travels to and from ROSA) and at rest (while it is stored on disks in AWS data centers).

Red Hat OpenShift Service on AWS provides secure access to Amazon Elastic Block Store (Amazon EBS) storage volumes attached to Amazon EC2 instances for ROSA control plane, infrastructure, and worker nodes, as well as Kubernetes persistent volumes for persistent storage. ROSA encrypts volume data at rest and in transit, and uses AWS Key Management Service (AWS KMS) to help protect your encrypted data. The service uses Amazon S3 for container image registry storage, which is encrypted at rest by default.

⚠ **Important**

Because ROSA is a managed service, AWS and Red Hat manage the infrastructure that ROSA uses. Customers should not attempt to manually shut down the Amazon EC2 instances that ROSA uses from the AWS console or CLI. This action can lead to customer data loss.

Data encryption for Amazon EBS-backed storage volumes

Red Hat OpenShift Service on AWS uses the Kubernetes persistent volume (PV) framework to allow cluster administrators to provision a cluster with persistent storage. Persistent volumes, as well as the control plane, infrastructure, and worker nodes, are backed by Amazon Elastic Block Store (Amazon EBS) storage volumes attached to Amazon EC2 instances.

For ROSA persistent volumes and nodes backed by Amazon EBS, encryption operations occur on the servers that host EC2 instances, ensuring the security of both data at rest and data in transit between an instance and its attached storage. For more information, see Amazon EBS encryption in the *Amazon EC2 User Guide*.

Data encryption for the Amazon EBS CSI driver and Amazon EFS CSI driver

ROSA defaults to using the Amazon EBS CSI driver to provision Amazon EBS storage. The Amazon EBS CSI driver and Amazon EBS CSI Driver Operator are installed on the cluster by default in the `openshift-cluster-csi-drivers` namespace. The Amazon EBS CSI driver and operator allow you to dynamically provision persistent volumes and create volume snapshots.

ROSA is also capable of provisioning persistent volumes using the Amazon EFS CSI driver and Amazon EFS CSI Driver Operator. The Amazon EFS driver and operator also allow you to share file system data between pods or with other applications within or outside of Kubernetes.

Volume data is secured in transit for both the Amazon EBS CSI driver and Amazon EFS CSI driver. For more information, see Using Container Storage Interface (CSI) in the Red Hat documentation.

⚠ **Important**

When dynamically provisioning ROSA persistent volumes using the Amazon EFS CSI driver, Amazon EFS considers the user ID, group ID (GID), and secondary group IDs of the access point when evaluating file system permissions. Amazon EFS replaces the user and group IDs on files with the user and group IDs on the access point and ignores NFS client IDs. As a result, Amazon EFS silently ignores `fsGroup` settings. ROSA is not able to replace the GIDs of files by using `fsGroup`. Any pod that can access a mounted Amazon EFS access point can access any file on the volume. For more information, see Working with Amazon EFS access points in the *Amazon EFS User Guide*.

etcd encryption

ROSA provides the option to enable encryption of `etcd` key values within the `etcd` volume during cluster creation, adding an additional layer of encryption. Once `etcd` is encrypted, you will incur approximately 20% additional performance overhead. We recommend that you enable `etcd` encryption only if you specifically require it for your use case. For more information, see etcd encryption in the ROSA service definition.

Key management

ROSA uses KMS keys to securely manage control plane, infrastructure, and worker data volumes and persistent volumes for customer applications. During cluster creation, you have the choice of using the default AWS managed KMS key provided by Amazon EBS, or specifying your own customer managed key. For more information, see the section called “Key management”.

Data encryption for the built-in image registry

ROSA provides a built-in container image registry to store, retrieve, and share container images via Amazon S3 bucket storage. The registry is configured and managed by the OpenShift Image Registry Operator. It provides an out-of-the-box solution for users to manage the images that run their workloads, and runs on top of the existing cluster infrastructure. For more information, see Registry in the Red Hat documentation.

ROSA offers public and private image registries. For enterprise applications, we recommend using a private registry to protect your images from being used by unauthorized users. To protect your registry’s data at rest, ROSA uses server-side encryption by default with Amazon S3 managed keys (SSE-S3). This does not require any action on your part, and is offered at no additional charge. For more information, see Protecting data using server-side encryption with Amazon S3 managed encryption keys (SSE-S3) in the *Amazon S3 User Guide*.

ROSA uses Transport Layer Security (TLS) protocol to secure data in transit to and from the image registry. For more information, see Registry in the Red Hat documentation.

Data encryption using KMS

ROSA uses AWS KMS to securely manage keys for encrypted data. Control plane, infrastructure, and worker node volumes are encrypted by default using the AWS managed KMS key provided by Amazon EBS. This KMS key has the alias `aws/ebs`. Persistent volumes that use the default gp3 storage class are also encrypted by default using this KMS key.

Newly created ROSA clusters are configured to use the default gp3 storage class to encrypt persistent volumes. Persistent volumes created by using any other storage class are only encrypted if the storage class is configured to be encrypted. For more information about ROSA pre-built storage classes, see Configuring persistent storage in the Red Hat documentation. Newly created ROSA clusters are configured to use the default gp3 storage class to encrypt persistent volumes. Persistent volumes created by using any other storage class are only encrypted if the storage class is configured to be encrypted. For more information about ROSA pre-built storage classes, see Configuring persistent storage in the Red Hat documentation.

During cluster creation, you can choose to encrypt the persistent volumes in your cluster using the default Amazon EBS-provided key, or specify your own customer managed symmetric KMS key. For more information about creating keys, see Creating symmetric encryption KMS keys in the *AWS KMS Developer Guide*.

You can also encrypt persistent volumes for individual containers within a cluster by defining a KMS key. This is useful when you have explicit compliance and security guidelines when deploying to AWS. For more information, see Encrypting container persistent volumes on AWS with a KMS key in the Red Hat documentation.

The following points should be considered when encrypting persistent volumes using your own KMS keys:

- When you use KMS encryption with your own KMS key, the key must exist in the same AWS Region as your cluster.
- There is a cost associated with creating and using your own KMS keys. For more information, see AWS Key Management Service pricing.

Internetwork traffic privacy

Red Hat OpenShift Service on AWS uses Amazon Virtual Private Cloud (Amazon VPC) to create boundaries between resources in your ROSA cluster and control traffic between them, your on-premises network, and the internet. For more information about Amazon VPC security, see Internetwork traffic privacy in Amazon VPC in the *Amazon VPC User Guide*.

Within the VPC, you can configure your ROSA clusters to use an HTTP or HTTPS proxy server to deny direct internet access. If you are a cluster administrator, you can also define network policies at the pod level that restrict internetwork traffic to pods in your ROSA cluster. For more information, see the section called “Infrastructure security”.

Identity and access management for ROSA

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use ROSA resources. IAM is an AWS service that you can use with no additional charge.

Topics

- Audience
- Authenticating with identities
- Managing access using policies
- ROSA identity-based policy examples
- AWS managed IAM policies for ROSA
- Troubleshooting ROSA identity and access

Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work you do in ROSA.

Service user - If you use the ROSA service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more ROSA features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in ROSA, see the section called “Troubleshooting”.

Service administrator - If you’re in charge of ROSA resources at your company, you probably have full access to ROSA. It’s your job to determine which ROSA features and resources your service users should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM.

IAM administrator - If you’re an IAM administrator, you might want to learn details about the policies used to manage access to ROSA. To view example ROSA identity-based policies that you can use in IAM, see the section called “ ROSA identity-based policy examples”.

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. You must be *authenticated* (signed in to AWS) as the AWS account root user, an IAM user, or by assuming an IAM role.

You can sign in to AWS as a federated identity by using credentials provided through an identity source. AWS IAM Identity Center (IAM Identity Center) users, your company’s single sign-on authentication, and your Google or Facebook credentials are examples of federated identities. When you sign in as a federated identity, your administrator previously set up identity federation using IAM roles. When you access AWS by using federation, you are indirectly assuming a role.

Depending on the type of user you are, you can sign in to the AWS Management Console or the AWS access portal. For more information about signing in to AWS, see How to sign in to your AWS account in the *AWS Sign-In User Guide*.

If you access AWS programmatically, AWS provides a software development kit (SDK) and a command line interface (CLI) to cryptographically sign your requests using your credentials. If you don’t use AWS tools, you must sign requests yourself. For more information about using the recommended method to sign requests yourself, see Signature Version 4 signing process in the *AWS General Reference*.

Regardless of the authentication method that you use, you might also be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see Multi-factor authentication in the *AWS IAM Identity Center (successor to AWS Single Sign-On) User Guide* and Using multi-factor authentication (MFA) in AWS in the *IAM User Guide*.

AWS account root user

When you create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account root user and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you don't use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform tasks that only the root user can perform. For the complete list of tasks that require you to sign in as the root user, see Tasks that require root user credentials in the *Account Management Reference Guide*.

Federated identity

As a best practice, require human users, including users that require administrator access, to use federation with an identity provider to access AWS services by using temporary credentials.

A federated identity is a user from your enterprise user directory, a web identity provider, the AWS Directory Service, the Identity Center directory, or any user that accesses AWS services by using credentials provided through an identity source. When federated identities access AWS accounts, they assume roles, and the roles provide temporary credentials.

For centralized access management, we recommend that you use AWS IAM Identity Center. You can create users and groups in IAM Identity Center, or you can connect and synchronize to a set of users and groups in your own identity source for use across all your AWS accounts and applications. For information about IAM Identity Center, see What is IAM Identity Center? in the *AWS IAM Identity Center (successor to AWS Single Sign-On) User Guide*.

IAM users and groups

An *IAM user* is an identity within your AWS account that has specific permissions for a single person or application. Where possible, we recommend relying on temporary credentials instead of creating IAM users who have long-term credentials such as passwords and access keys. However, if you have specific use cases that require long-term credentials with IAM users, we recommend that you rotate access keys. For more information, see Rotate access keys regularly for use cases that require long-term credentials in the *IAM User Guide*.

An IAM group is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named *IAMAdmins* and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see When to create an IAM user (instead of a role) in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. You can temporarily assume an IAM role in the AWS Management Console by switching roles. You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see Using IAM roles in the *IAM User Guide*.

IAM roles with temporary credentials are useful in the following situations:

- **Federated user access** - To assign permissions to a federated identity, you create a role and define permissions for the role. When a federated identity authenticates, the identity is associated with the role and is granted the permissions that are defined by the role. For information about roles for federation, see Creating a role for a third-party Identity Provider in the *IAM User Guide*. If you use IAM Identity Center, you configure a permission set. To control what your identities can access after they authenticate, IAM Identity Center correlates the permission set to a role in IAM. For information about permissions sets, see Permission sets in the *AWS IAM Identity Center (successor to AWS Single Sign-On) User Guide*.
- **Temporary IAM user permissions** - An IAM user can assume an IAM role to temporarily take on different permissions for a specific task.
- **Cross-account access** - You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see How IAM roles differ from resource-based policies in the *IAM User Guide*.
- **Cross-service access** - Some AWS services use features in other AWS services. For example, when you make a call in a service, it’s common for that service to run applications in Amazon EC2 or store objects in Amazon S3. A service might do this using the calling principal’s permissions, using a service role, or using a service-linked role.
  - **Forward access sessions** (FAS) - When you use an IAM user or role to perform actions in AWS, you are considered a principal. When you use some services, you might perform an

action that then initiates another action in a different service. FAS uses the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. FAS requests are only made when a service receives a request that requires interactions with other AWS services or resources to complete. In this case, you must have permissions to perform both actions. For policy details when making FAS requests, see Forward access sessions.

  - **Service role** - A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Creating a role to delegate permissions to an AWS service in the *IAM User Guide*.
  - **Service-linked role** - A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your IAM account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.
- **Applications running on Amazon EC2** - You can use an IAM role to manage temporary credentials for applications that are running on an Amazon EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the Amazon EC2 instance. To assign an AWS role to an Amazon EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the Amazon EC2 instance to get temporary credentials. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances in the *IAM User Guide*.

To learn whether to use IAM roles or IAM users, see When to create an IAM role (instead of a user) in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to AWS identities or resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when a principal (user, root user, or role session) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see Overview of JSON policies in the *IAM User Guide*.

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

By default, users and roles have no permissions. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, role, or group. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Creating IAM policies in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see Choosing between managed policies and inline policies in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

Access control lists (ACLs)

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see Access Control List (ACL) overview in the *Amazon Simple Storage Service Developer Guide*.

Other policy types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** - A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see Permissions boundaries for IAM entities in the *IAM User Guide*.
- **Service control policies (SCPs)** - SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see How SCPs work in the *AWS Organizations User Guide*.
- **Session policies** - Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see Session policies in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see Policy evaluation logic in the *IAM User Guide*.

ROSA identity-based policy examples

By default, IAM users and roles don't have permission to create or modify AWS resources. They also can't perform tasks using the AWS Management Console, AWS CLI, or AWS API. An IAM administrator must create IAM policies that grant users and roles permission to perform specific API operations on the specified resources they need. The administrator must then attach those policies to the IAM users or groups that require those permissions.

To learn how to create an IAM identity-based policy using these example JSON policy documents, see Creating policies on the JSON tab in the *IAM User Guide*.

Using the ROSA console

To subscribe to ROSA from the console, your IAM principal must have the required AWS Marketplace permissions. The permissions allow the principal to subscribe and unsubscribe to the ROSA product listing in AWS Marketplace and view AWS Marketplace subscriptions. To add the required permissions, go to the ROSA console and attach the AWS managed policy `ROSAManageSubscription` to your IAM principal. For more information about `ROSAManageSubscription`, see xref:security-iam-awsmanpol-rosamanagesubscription.

AWS managed policies for ROSA with HCP

ROSA with hosted control planes (HCP) uses AWS managed policies with permissions that are required for service operation and support. You use the ROSA CLI or IAM console to attach these policies to service roles in your AWS account.

For more information, see xref:security-iam-awsmanpol.

Customer managed policies for ROSA classic

ROSA classic uses customer managed IAM policies with permissions that are pre-defined by the service. You use the ROSA CLI to create these policies and attach them to service roles in your AWS account. ROSA requires that these policies are configured as defined by the service to ensure continuous operation and service support.

Note

You should not alter ROSA classic policies without first consulting Red Hat. Doing so may void Red Hat’s 99.95% cluster uptime service-level agreement. ROSA with hosted control planes uses AWS managed policies with a more limited set of permissions. For more information, see the section called “ AWS managed IAM policies”.

There are two types of customer managed policies for ROSA: account policies and operator policies. Account policies are attached to IAM roles that the service uses to establish a trust relationship with Red Hat for site reliability engineer (SRE) support, cluster creation, and compute functionality. Operator policies are attached to IAM roles that OpenShift operators use for cluster operations related to ingress, storage, image registry, and node management. Account policies are created once per AWS account, whereas operator polices are created once per cluster.

For more information, see the section called “ROSA classic account policies” and the section called “ROSA classic operator policies”.

Allow users to view their own permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

ROSA classic account policies

This section provides details about the account policies that are required for ROSA classic. These permissions are needed for ROSA classic to manage the AWS resources that clusters run on and enable Red Hat site reliability engineer support for clusters. You can assign a custom prefix to the policy names, but these policies should otherwise be named as defined on this page (for example, `ManagedOpenShift-Installer-Role-Policy`).

The account policies are specific to an OpenShift minor release version and are backward compatible. Before creating or upgrading a cluster, you should verify that the policy version and cluster version are the same by running `rosa list account-roles`. If the policy version is less than the cluster version, run `rosa upgrade account-roles` to upgrade the roles and attached policies. You can use the same account policies and roles for multiple clusters of the same minor release version.

[Prefix]-Installer-Role-Policy

You can attach `[Prefix]-Installer-Role-Policy` to your IAM entities. Before you can create a ROSA classic cluster, you must first attach this policy to an IAM role named `[Prefix]-Installer-Role`. This policy grants required permissions that allow the ROSA installer to manage the AWS resources that are needed for cluster creation.

Permissions policy

Permissions defined in this policy document specify which actions are allowed or denied.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "autoscaling:DescribeAutoScalingGroups",
                "ec2:AllocateAddress",
                "ec2:AssociateAddress",
                "ec2:AssociateDhcpOptions",
                "ec2:AssociateRouteTable",
                "ec2:AttachInternetGateway",
                "ec2:AttachNetworkInterface",
                "ec2:AuthorizeSecurityGroupEgress",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:CopyImage",
                "ec2:CreateDhcpOptions",
                "ec2:CreateInternetGateway",
                "ec2:CreateNatGateway",
                "ec2:CreateNetworkInterface",
                "ec2:CreateRoute",
                "ec2:CreateRouteTable",
                "ec2:CreateSecurityGroup",
                "ec2:CreateSubnet",
                "ec2:CreateTags",
                "ec2:CreateVolume",
                "ec2:CreateVpc",
                "ec2:CreateVpcEndpoint",
                "ec2:DeleteDhcpOptions",
                "ec2:DeleteInternetGateway",
                "ec2:DeleteNatGateway",
                "ec2:DeleteNetworkInterface",
                "ec2:DeleteRoute",
                "ec2:DeleteRouteTable",
                "ec2:DeleteSecurityGroup",
                "ec2:DeleteSnapshot",
                "ec2:DeleteSubnet",
                "ec2:DeleteTags",
                "ec2:DeleteVolume",
                "ec2:DeleteVpc",
                "ec2:DeleteVpcEndpoints",
                "ec2:DeregisterImage",
                "ec2:DescribeAccountAttributes",
                "ec2:DescribeAddresses",
                "ec2:DescribeAvailabilityZones",
                "ec2:DescribeDhcpOptions",
                "ec2:DescribeImages",
                "ec2:DescribeInstanceAttribute",
                "ec2:DescribeInstanceCreditSpecifications",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeInstanceTypeOfferings",
                "ec2:DescribeInstanceTypes",
                "ec2:DescribeInternetGateways",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeNatGateways",
                "ec2:DescribeNetworkAcls",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribePrefixLists",
                "ec2:DescribeRegions",
                "ec2:DescribeReservedInstancesOfferings",
                "ec2:DescribeRouteTables",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSecurityGroupRules",
                "ec2:DescribeSubnets",
                "ec2:DescribeTags",
                "ec2:DescribeVolumes",
                "ec2:DescribeVpcAttribute",
                "ec2:DescribeVpcClassicLink",
                "ec2:DescribeVpcClassicLinkDnsSupport",
                "ec2:DescribeVpcEndpoints",
                "ec2:DescribeVpcs",
                "ec2:DetachInternetGateway",
                "ec2:DisassociateRouteTable",
                "ec2:GetConsoleOutput",
                "ec2:GetEbsDefaultKmsKeyId",
                "ec2:ModifyInstanceAttribute",
                "ec2:ModifyNetworkInterfaceAttribute",
                "ec2:ModifySubnetAttribute",
                "ec2:ModifyVpcAttribute",
                "ec2:ReleaseAddress",
                "ec2:ReplaceRouteTableAssociation",
                "ec2:RevokeSecurityGroupEgress",
                "ec2:RevokeSecurityGroupIngress",
                "ec2:RunInstances",
                "ec2:StartInstances",
                "ec2:StopInstances",
                "ec2:TerminateInstances",
                "elasticloadbalancing:AddTags",
                "elasticloadbalancing:ApplySecurityGroupsToLoadBalancer",
                "elasticloadbalancing:AttachLoadBalancerToSubnets",
                "elasticloadbalancing:ConfigureHealthCheck",
                "elasticloadbalancing:CreateListener",
                "elasticloadbalancing:CreateLoadBalancer",
                "elasticloadbalancing:CreateLoadBalancerListeners",
                "elasticloadbalancing:CreateTargetGroup",
                "elasticloadbalancing:DeleteLoadBalancer",
                "elasticloadbalancing:DeleteTargetGroup",
                "elasticloadbalancing:DeregisterInstancesFromLoadBalancer",
                "elasticloadbalancing:DeregisterTargets",
                "elasticloadbalancing:DescribeAccountLimits",
                "elasticloadbalancing:DescribeInstanceHealth",
                "elasticloadbalancing:DescribeListeners",
                "elasticloadbalancing:DescribeLoadBalancerAttributes",
                "elasticloadbalancing:DescribeLoadBalancers",
                "elasticloadbalancing:DescribeTags",
                "elasticloadbalancing:DescribeTargetGroupAttributes",
                "elasticloadbalancing:DescribeTargetGroups",
                "elasticloadbalancing:DescribeTargetHealth",
                "elasticloadbalancing:ModifyLoadBalancerAttributes",
                "elasticloadbalancing:ModifyTargetGroup",
                "elasticloadbalancing:ModifyTargetGroupAttributes",
                "elasticloadbalancing:RegisterInstancesWithLoadBalancer",
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:SetLoadBalancerPoliciesOfListener",
                "iam:AddRoleToInstanceProfile",
                "iam:CreateInstanceProfile",
                "iam:DeleteInstanceProfile",
                "iam:GetInstanceProfile",
                "iam:TagInstanceProfile",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:GetUser",
                "iam:ListAttachedRolePolicies",
                "iam:ListInstanceProfiles",
                "iam:ListInstanceProfilesForRole",
                "iam:ListRolePolicies",
                "iam:ListRoles",
                "iam:ListUserPolicies",
                "iam:ListUsers",
                "iam:PassRole",
                "iam:RemoveRoleFromInstanceProfile",
                "iam:SimulatePrincipalPolicy",
                "iam:TagRole",
                "iam:UntagRole",
                "route53:ChangeResourceRecordSets",
                "route53:ChangeTagsForResource",
                "route53:CreateHostedZone",
                "route53:DeleteHostedZone",
                "route53:GetAccountLimit",
                "route53:GetChange",
                "route53:GetHostedZone",
                "route53:ListHostedZones",
                "route53:ListHostedZonesByName",
                "route53:ListResourceRecordSets",
                "route53:ListTagsForResource",
                "route53:UpdateHostedZoneComment",
                "s3:CreateBucket",
                "s3:DeleteBucket",
                "s3:DeleteObject",
                "s3:DeleteObjectVersion",
                "s3:GetAccelerateConfiguration",
                "s3:GetBucketAcl",
                "s3:GetBucketCORS",
                "s3:GetBucketLocation",
                "s3:GetBucketLogging",
                "s3:GetBucketObjectLockConfiguration",
                "s3:GetBucketPolicy",
                "s3:GetBucketReplication",
                "s3:GetBucketRequestPayment",
                "s3:GetBucketTagging",
                "s3:GetBucketVersioning",
                "s3:GetBucketWebsite",
                "s3:GetEncryptionConfiguration",
                "s3:GetLifecycleConfiguration",
                "s3:GetObject",
                "s3:GetObjectAcl",
                "s3:GetObjectTagging",
                "s3:GetObjectVersion",
                "s3:GetReplicationConfiguration",
                "s3:ListBucket",
                "s3:ListBucketVersions",
                "s3:PutBucketAcl",
                "s3:PutBucketTagging",
                "s3:PutBucketVersioning",
                "s3:PutEncryptionConfiguration",
                "s3:PutObject",
                "s3:PutObjectAcl",
                "s3:PutObjectTagging",
                "servicequotas:GetServiceQuota",
                "servicequotas:ListAWSDefaultServiceQuotas",
                "sts:AssumeRole",
                "sts:AssumeRoleWithWebIdentity",
                "sts:GetCallerIdentity",
                "tag:GetResources",
                "tag:UntagResources",
                "ec2:CreateVpcEndpointServiceConfiguration",
                "ec2:DeleteVpcEndpointServiceConfigurations",
                "ec2:DescribeVpcEndpointServiceConfigurations",
                "ec2:DescribeVpcEndpointServicePermissions",
                "ec2:DescribeVpcEndpointServices",
                "ec2:ModifyVpcEndpointServicePermissions",
                "kms:DescribeKey",
                "cloudwatch:GetMetricData"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/red-hat-managed": "true"
                }
            }
        }
    ]
}
```

[Prefix]-ControlPlane-Role-Policy

You can attach `[Prefix]-ControlPlane-Role-Policy` to your IAM entities. Before you can create a ROSA classic cluster, you must first attach this policy to an IAM role named `[Prefix]-ControlPlane-Role`. This policy grants required permissions to ROSA classic to manage Amazon

EC2 and Elastic Load Balancing resources that host the ROSA control plane, as well as read KMS keys.

Permissions policy

Permissions defined in this policy document specify which actions are allowed or denied.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ec2:AttachVolume",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:CreateSecurityGroup",
                "ec2:CreateTags",
                "ec2:CreateVolume",
                "ec2:DeleteSecurityGroup",
                "ec2:DeleteVolume",
                "ec2:Describe*",
                "ec2:DetachVolume",
                "ec2:ModifyInstanceAttribute",
                "ec2:ModifyVolume",
                "ec2:RevokeSecurityGroupIngress",
                "elasticloadbalancing:AddTags",
                "elasticloadbalancing:AttachLoadBalancerToSubnets",
                "elasticloadbalancing:ApplySecurityGroupsToLoadBalancer",
                "elasticloadbalancing:CreateListener",
                "elasticloadbalancing:CreateLoadBalancer",
                "elasticloadbalancing:CreateLoadBalancerPolicy",
                "elasticloadbalancing:CreateLoadBalancerListeners",
                "elasticloadbalancing:CreateTargetGroup",
                "elasticloadbalancing:ConfigureHealthCheck",
                "elasticloadbalancing:DeleteListener",
                "elasticloadbalancing:DeleteLoadBalancer",
                "elasticloadbalancing:DeleteLoadBalancerListeners",
                "elasticloadbalancing:DeleteTargetGroup",
                "elasticloadbalancing:DeregisterInstancesFromLoadBalancer",
                "elasticloadbalancing:DeregisterTargets",
                "elasticloadbalancing:Describe*",
                "elasticloadbalancing:DetachLoadBalancerFromSubnets",
                "elasticloadbalancing:ModifyListener",
                "elasticloadbalancing:ModifyLoadBalancerAttributes",
                "elasticloadbalancing:ModifyTargetGroup",
                "elasticloadbalancing:ModifyTargetGroupAttributes",
                "elasticloadbalancing:RegisterInstancesWithLoadBalancer",
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer",
                "elasticloadbalancing:SetLoadBalancerPoliciesOfListener",
                "kms:DescribeKey"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

[Prefix]-Worker-Role-Policy

You can attach `[Prefix]-Worker-Role-Policy` to your IAM entities. Before you can create a ROSA classic cluster, you must first attach this policy to an IAM role named `[Prefix]-Worker-Role`. This policy grants required permissions to ROSA classic to describe the EC2 instances running as worker nodes.

Permissions policy

Permissions defined in this policy document specify which actions are allowed or denied.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeRegions"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

[Prefix]-Support-Role-Policy

You can attach `[Prefix]-Support-Role-Policy` to your IAM entities. Before you can create a ROSA classic cluster, you must first attach this policy to an IAM role named `[Prefix]-Support-Role`. This policy grants required permissions to Red Hat site reliability engineering to observe, diagnose, and support the AWS resources that ROSA classic clusters use, including the ability to change cluster node state.

Permissions policy

Permissions defined in this policy document specify which actions are allowed or denied.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "cloudtrail:DescribeTrails",
                "cloudtrail:LookupEvents",
                "cloudwatch:GetMetricData",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics",
                "ec2-instance-connect:SendSerialConsoleSSHPublicKey",
                "ec2:CopySnapshot",
                "ec2:CreateNetworkInsightsPath",
                "ec2:CreateSnapshot",
                "ec2:CreateSnapshots",
                "ec2:CreateTags",
                "ec2:DeleteNetworkInsightsAnalysis",
                "ec2:DeleteNetworkInsightsPath",
                "ec2:DeleteTags",
                "ec2:DescribeAccountAttributes",
                "ec2:DescribeAddresses",
                "ec2:DescribeAddressesAttribute",
                "ec2:DescribeAggregateIdFormat",
                "ec2:DescribeAvailabilityZones",
                "ec2:DescribeByoipCidrs",
                "ec2:DescribeCapacityReservations",
                "ec2:DescribeCarrierGateways",
                "ec2:DescribeClassicLinkInstances",
                "ec2:DescribeClientVpnAuthorizationRules",
                "ec2:DescribeClientVpnConnections",
                "ec2:DescribeClientVpnEndpoints",
                "ec2:DescribeClientVpnRoutes",
                "ec2:DescribeClientVpnTargetNetworks",
                "ec2:DescribeCoipPools",
                "ec2:DescribeCustomerGateways",
                "ec2:DescribeDhcpOptions",
                "ec2:DescribeEgressOnlyInternetGateways",
                "ec2:DescribeIamInstanceProfileAssociations",
                "ec2:DescribeIdentityIdFormat",
                "ec2:DescribeIdFormat",
                "ec2:DescribeImageAttribute",
                "ec2:DescribeImages",
                "ec2:DescribeInstanceAttribute",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeInstanceTypeOfferings",
                "ec2:DescribeInstanceTypes",
                "ec2:DescribeInternetGateways",
                "ec2:DescribeIpv6Pools",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeLaunchTemplates",
                "ec2:DescribeLocalGatewayRouteTables",
                "ec2:DescribeLocalGatewayRouteTableVirtualInterfaceGroupAssociations",
                "ec2:DescribeLocalGatewayRouteTableVpcAssociations",
                "ec2:DescribeLocalGateways",
                "ec2:DescribeLocalGatewayVirtualInterfaceGroups",
                "ec2:DescribeLocalGatewayVirtualInterfaces",
                "ec2:DescribeManagedPrefixLists",
                "ec2:DescribeNatGateways",
                "ec2:DescribeNetworkAcls",
                "ec2:DescribeNetworkInsightsAnalyses",
                "ec2:DescribeNetworkInsightsPaths",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribePlacementGroups",
                "ec2:DescribePrefixLists",
                "ec2:DescribePrincipalIdFormat",
                "ec2:DescribePublicIpv4Pools",
                "ec2:DescribeRegions",
                "ec2:DescribeReservedInstances",
                "ec2:DescribeRouteTables",
                "ec2:DescribeScheduledInstances",
                "ec2:DescribeSecurityGroupReferences",
                "ec2:DescribeSecurityGroupRules",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSnapshotAttribute",
                "ec2:DescribeSnapshots",
                "ec2:DescribeSpotFleetInstances",
                "ec2:DescribeStaleSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeTags",
                "ec2:DescribeTransitGatewayAttachments",
                "ec2:DescribeTransitGatewayConnectPeers",
                "ec2:DescribeTransitGatewayConnects",
                "ec2:DescribeTransitGatewayMulticastDomains",
                "ec2:DescribeTransitGatewayPeeringAttachments",
                "ec2:DescribeTransitGatewayRouteTables",
                "ec2:DescribeTransitGateways",
                "ec2:DescribeTransitGatewayVpcAttachments",
                "ec2:DescribeVolumeAttribute",
                "ec2:DescribeVolumes",
                "ec2:DescribeVolumesModifications",
                "ec2:DescribeVolumeStatus",
                "ec2:DescribeVpcAttribute",
                "ec2:DescribeVpcClassicLink",
                "ec2:DescribeVpcClassicLinkDnsSupport",
                "ec2:DescribeVpcEndpointConnectionNotifications",
                "ec2:DescribeVpcEndpointConnections",
                "ec2:DescribeVpcEndpoints",
                "ec2:DescribeVpcEndpointServiceConfigurations",
                "ec2:DescribeVpcEndpointServicePermissions",
                "ec2:DescribeVpcEndpointServices",
                "ec2:DescribeVpcPeeringConnections",
                "ec2:DescribeVpcs",
                "ec2:DescribeVpnConnections",
                "ec2:DescribeVpnGateways",
                "ec2:GetAssociatedIpv6PoolCidrs",
                "ec2:GetConsoleOutput",
                "ec2:GetManagedPrefixListEntries",
                "ec2:GetSerialConsoleAccessStatus",
                "ec2:GetTransitGatewayAttachmentPropagations",
                "ec2:GetTransitGatewayMulticastDomainAssociations",
                "ec2:GetTransitGatewayPrefixListReferences",
                "ec2:GetTransitGatewayRouteTableAssociations",
                "ec2:GetTransitGatewayRouteTablePropagations",
                "ec2:ModifyInstanceAttribute",
                "ec2:RebootInstances",
                "ec2:RunInstances",
                "ec2:SearchLocalGatewayRoutes",
                "ec2:SearchTransitGatewayMulticastGroups",
                "ec2:SearchTransitGatewayRoutes",
                "ec2:StartInstances",
                "ec2:StartNetworkInsightsAnalysis",
                "ec2:StopInstances",
                "ec2:TerminateInstances",
                "elasticloadbalancing:ConfigureHealthCheck",
                "elasticloadbalancing:DescribeAccountLimits",
                "elasticloadbalancing:DescribeInstanceHealth",
                "elasticloadbalancing:DescribeListenerCertificates",
                "elasticloadbalancing:DescribeListeners",
                "elasticloadbalancing:DescribeLoadBalancerAttributes",
                "elasticloadbalancing:DescribeLoadBalancerPolicies",
                "elasticloadbalancing:DescribeLoadBalancerPolicyTypes",
                "elasticloadbalancing:DescribeLoadBalancers",
                "elasticloadbalancing:DescribeRules",
                "elasticloadbalancing:DescribeSSLPolicies",
                "elasticloadbalancing:DescribeTags",
                "elasticloadbalancing:DescribeTargetGroupAttributes",
                "elasticloadbalancing:DescribeTargetGroups",
                "elasticloadbalancing:DescribeTargetHealth",
                "iam:GetRole",
                "iam:ListRoles",
                "kms:CreateGrant",
                "route53:GetHostedZone",
                "route53:GetHostedZoneCount",
                "route53:ListHostedZones",
                "route53:ListHostedZonesByName",
                "route53:ListResourceRecordSets",
                "s3:GetBucketTagging",
                "s3:GetObjectAcl",
                "s3:GetObjectTagging",
                "s3:ListAllMyBuckets",
                "sts:DecodeAuthorizationMessage",
                "tiros:CreateQuery",
                "tiros:GetQueryAnswer",
                "tiros:GetQueryExplanation"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "s3:ListBucket"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::managed-velero*",
                "arn:aws:s3:::*image-registry*"
            ]
        }
```

```
    ]
}
```

ROSA classic operator policies

This section provides details about the operator policies that are required for ROSA classic. Before you can create a ROSA classic cluster, you must first attach these policies to the relevant operator roles. A unique set of operator roles is required for each cluster.

These permissions are needed to allow the OpenShift operators to manage ROSA classic cluster nodes. You can assign a custom prefix to the policy names to simplify policy management (for example, `ManagedOpenShift-openshift-ingress-operator-cloud-credentials`).

[Prefix]-openshift-ingress-operator-cloud-credentials

You can attach `[Prefix]-openshift-ingress-operator-cloud-credentials` to your IAM entities. This policy grants required permissions to the Ingress Operator to provision and manage load balancers and DNS configurations for external cluster access. The policy also allows the Ingress Operator to read and filter Route 53 resource tag values to discover hosted zones. For more information about the operator, see OpenShift Ingress Operator in the OpenShift GitHub documentation.

Permissions policy

Permissions defined in this policy document specify which actions are allowed or denied.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "elasticloadbalancing:DescribeLoadBalancers",
                "route53:ListHostedZones",
                "route53:ListTagsForResources",
                "route53:ChangeResourceRecordSets",
                "tag:GetResources"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

[Prefix]-openshift-cluster-csi-drivers-ebs-cloud-credentials

You can attach `[Prefix]-openshift-cluster-csi-drivers-ebs-cloud-credentials` to your IAM entities. This policy grants required permissions to the Amazon EBS CSI Driver Operator to install and maintain the Amazon EBS CSI driver on a ROSA classic cluster. For more information about the operator, see aws-ebs-csi-driver-operator in the OpenShift GitHub documentation.

Permissions policy

Permissions defined in this policy document specify which actions are allowed or denied.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ec2:AttachVolume",
                "ec2:CreateSnapshot",
                "ec2:CreateTags",
                "ec2:CreateVolume",
                "ec2:DeleteSnapshot",
                "ec2:DeleteTags",
                "ec2:DeleteVolume",
                "ec2:DescribeAvailabilityZones",
                "ec2:DescribeInstances",
                "ec2:DescribeSnapshots",
                "ec2:DescribeTags",
                "ec2:DescribeVolumes",
                "ec2:DescribeVolumesModifications",
                "ec2:DetachVolume",
                "ec2:EnableFastSnapshotRestores",
                "ec2:ModifyVolume"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

[Prefix]-openshift-machine-api-aws-cloud-credentials

You can attach `[Prefix]-openshift-machine-api-aws-cloud-credentials` to your IAM entities. This policy grants required permissions to the Machine Config Operator to describe, run, and terminate Amazon EC2 instances managed as worker nodes. This policy also grants permissions to allow for disk encryption of the worker node root volume using AWS KMS keys. For more information about the operator, see machine-config-operator in the OpenShift GitHub documentation.

Permissions policy

Permissions defined in this policy document specify which actions are allowed or denied.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ec2:CreateTags",
                "ec2:DescribeAvailabilityZones",
                "ec2:DescribeDhcpOptions",
                "ec2:DescribeImages",
                "ec2:DescribeInstances",
                "ec2:DescribeInternetGateways",
                "ec2:DescribeInstanceTypes",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeRegions",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "ec2:RunInstances",
                "ec2:TerminateInstances",
                "elasticloadbalancing:DescribeLoadBalancers",
                "elasticloadbalancing:DescribeTargetGroups",
                "elasticloadbalancing:DescribeTargetHealth",
                "elasticloadbalancing:RegisterInstancesWithLoadBalancer",
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:DeregisterTargets",
                "iam:PassRole",
                "iam:CreateServiceLinkedRole"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "kms:Decrypt",
                "kms:Encrypt",
                "kms:GenerateDataKey",
                "kms:GenerateDataKeyWithoutPlainText",
                "kms:DescribeKey"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "kms:RevokeGrant",
                "kms:CreateGrant",
                "kms:ListGrants"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "Bool": {
                    "kms:GrantIsForAWSResource": true
                }
            }
        }
    ]
}
```

[Prefix]-openshift-cloud-credential-operator-cloud-credentials

You can attach `[Prefix]-openshift-cloud-credential-operator-cloud-credentials` to your IAM entities. This policy grants required permissions to the Cloud Credential Operator to retrieve IAM user details, including access key IDs, attached inline policy documents, user's creation date, path, user ID, and Amazon Resource Name (ARN). For more information about the operator, see cloud-credential-operator in the OpenShift GitHub documentation.

Permissions policy

Permissions defined in this policy document specify which actions are allowed or denied.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetUser",
                "iam:GetUserPolicy",
                "iam:ListAccessKeys"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

[Prefix]-openshift-image-registry-installer-cloud-credentials

You can attach `[Prefix]-openshift-image-registry-installer-cloud-credentials` to your IAM entities. This policy grants required permissions to the Image Registry Operator to provision and manage resources for ROSA classic's in-cluster image registry and dependent services, including Amazon S3. This is required so that the operator can install and maintain the internal registry of a ROSA classic cluster. For more information about the operator, see Image Registry Operator in the OpenShift GitHub documentation.

Permissions policy

Permissions defined in this policy document specify which actions are allowed or denied.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "s3:CreateBucket",
                "s3:DeleteBucket",
                "s3:PutBucketTagging",
                "s3:GetBucketTagging",
                "s3:PutBucketPublicAccessBlock",
                "s3:GetBucketPublicAccessBlock",
                "s3:PutEncryptionConfiguration",
                "s3:GetEncryptionConfiguration",
                "s3:PutLifecycleConfiguration",
                "s3:GetLifecycleConfiguration",
                "s3:GetBucketLocation",
                "s3:ListBucket",
                "s3:GetObject",
                "s3:PutObject",
                "s3:DeleteObject",
                "s3:ListBucketMultipartUploads",
                "s3:AbortMultipartUpload",
                "s3:ListMultipartUploadParts"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

[Prefix]-openshift-cloud-network-config-controller-cloud-cr

You can attach `[Prefix]-openshift-cloud-network-config-controller-cloud-cr` to your IAM entities. This policy grants required permissions to the Cloud Network Config Controller Operator to provision and manage networking resources for use by the ROSA classic cluster networking overlay. The operator uses these permissions to manage private IP addresses for Amazon EC2 instances as part of the ROSA classic cluster. For more information about the operator, see Cloud-network-config-controller in the OpenShift GitHub documentation.

Permissions policy

Permissions defined in this policy document specify which actions are allowed or denied.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeInstanceTypes",
                "ec2:UnassignPrivateIpAddresses",
                "ec2:AssignPrivateIpAddresses",
                "ec2:UnassignIpv6Addresses",
                "ec2:AssignIpv6Addresses",
                "ec2:DescribeSubnets",
                "ec2:DescribeNetworkInterfaces"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

AWS managed IAM policies for ROSA

An AWS managed policy is a standalone policy that is created and administered by AWS. AWS managed policies are designed to provide permissions for many common use cases so that you can start assigning permissions to users, groups, and roles.

Keep in mind that AWS managed policies might not grant least-privilege permissions for your specific use cases because they're available for all AWS customers to use. We recommend that you reduce permissions further by defining customer managed policies that are specific to your use cases.

You cannot change the permissions defined in AWS managed policies. If AWS updates the permissions defined in an AWS managed policy, the update affects all principal identities (users, groups, and roles) that the policy is attached to. AWS is most likely to update an AWS managed policy when a new AWS service is launched or new API operations become available for existing services.

For more information, see AWS managed policies in the *IAM User Guide*.

AWS managed policy: ROSAManageSubscription

You can attach the `ROSAManageSubscription` policy to your IAM entities. Before you enable ROSA in the AWS ROSA console, you must first attach this policy to a console role.

This policy grants the AWS Marketplace permissions required for you to manage the ROSA subscription.

Permissions details

This policy includes the following permissions.

- `aws-marketplace:Subscribe` - Grants permission to subscribe to the AWS Marketplace product for ROSA.
- `aws-marketplace:Unsubscribe` - Allows principals to remove subscriptions to AWS Marketplace products.
- `aws-marketplace:ViewSubscriptions` - Allows principals to view subscriptions from AWS Marketplace. This is required so that the IAM principal can view the available AWS Marketplace subscriptions.

To view the full JSON policy document, see ROSAManageSubscription in the *AWS Managed Policy Reference Guide*.

AWS managed policies for ROSA with HCP account roles

You can attach these AWS managed policies to the account roles needed to use ROSA with hosted control planes (HCP). The permissions are required for Red Hat site reliability engineering (SRE) support on the cluster, cluster creation, and compute functionality.

The following managed policies are required:

- the section called "AWS managed policy: ROSAWorkerInstancePolicy" — Allows the ROSA service to manage Amazon EC2 instance lifecycles in a ROSA cluster.
- the section called "AWS managed policy: ROSASRESupportPolicy" — Grants required permissions to Red Hat site reliability engineers (SREs) to directly observe, diagnose, and support AWS resources associated with ROSA clusters, including the ability to change ROSA cluster node state.
- the section called "AWS managed policy: ROSAInstallerPolicy" — Grants required permissions to the installer to manage AWS resources that support cluster installation.

AWS managed policies for ROSA with HCP operator roles

You can attach these AWS managed policies to the operator roles needed to use ROSA with hosted control planes (HCP). The permissions are required to allow OpenShift operators to manage ROSA with HCP cluster nodes.

The following managed policies are required:

- the section called "AWS managed policy: ROSAAmazonEBSCSIDriverOperatorPolicy" — Grants required permissions to the Amazon EBS CSI Driver Operator to install and maintain the Amazon EBS CSI driver on a ROSA cluster.
- the section called "AWS managed policy: ROSAIngressOperatorPolicy" — Grants required permissions to the Ingress Operator to provision and manage load balancers and DNS configurations for ROSA clusters. The policy allows read access to tag values. The operator then filters the tag values for Route 53 resources to discover hosted zones.
- the section called "AWS managed policy: ROSAImageRegistryOperatorPolicy" — Grants required permissions to the Image Registry Operator to provision and manage resources for the ROSA in-cluster image registry and dependent services, including S3.

- the section called “AWS managed policy: ROSACloudNetworkConfigOperatorPolicy” — Grants required permissions to the Cloud Network Config Controller Operator to provision and manage networking resources for the ROSA cluster networking overlay.
- the section called “AWS managed policy: ROSAKubeControllerPolicy” — Grants required permissions to kube controller to manage Amazon EC2, Elastic Load Balancing, and AWS KMS resources for a ROSA with hosted control planes cluster.
- the section called “AWS managed policy: ROSANodePoolManagementPolicy” — Grants required permissions to the NodePool controller to describe, run, and terminate Amazon EC2 instances managed as worker nodes. This policy also enables disk encryption of the worker node root volume using AWS KMS keys.
- the section called “AWS managed policy: ROSAKMSProviderPolicy” — Grants required permissions to the built-in AWS Encryption Provider to manage AWS KMS keys that support etcd data encryption. This policy allows Amazon EC2 to encrypt and decrypt `etcd` data using the KMS keys provided by the AWS Encryption Provider.
- the section called “AWS managed policy: ROSAControlPlaneOperatorPolicy” — Grants required permissions to the Control Plane Operator to manage Amazon EC2 and Route 53 resources for ROSA with hosted control planes clusters.

To view managed policy permissions, see AWS managed policies in the *AWS Managed Policy Reference Guide*.

ROSA updates to AWS managed policies

View details about updates to AWS managed policies for ROSA since this service began tracking these changes. For automatic alerts about changes to this page, subscribe to the RSS feed on the xref:doc-history page.

Change	Description	Date
ROSANodePoolManagementPolicy — Policy updated	ROSA updated the policy to allow the ROSA node pool manager to describe DHCP option sets in order to set the proper private DNS names. To learn more, see the section called “AWS managed policy:	May 2, 2024

Change	Description	Date
	ROSANodePoolManagementPolicy”.	
ROSAInstallerPolicy — Policy updated	ROSA updated the policy to allow the ROSA installer to add tags to subnets using tag keys matching `"kubernetes.io/cluster/*"` . To learn more, see the section called “AWS managed policy: ROSAInstallerPolicy”.	April 24, 2024
ROSASRESupportPolicy — Policy updated	ROSA updated the policy to allow the SRE role to retrieve information on instance profiles that have been tagged by ROSA as `red-hat-managed` . To learn more, see the section called “AWS managed policy: ROSASRESupportPolicy”.	April 10, 2024
ROSAInstallerPolicy — Policy updated	ROSA updated the policy to allow the ROSA installer to validate that AWS managed policies for ROSA are attached to IAM roles used by ROSA. This update also allows the installer to identify whether customer managed policies have been attached to ROSA roles. To learn more, see the section called “AWS managed policy: ROSAInstallerPolicy”.	April 10, 2024

Change	Description	Date
ROSAInstallerPolicy — Policy updated	ROSA updated the policy to allow the service to provide installer alert messages when cluster installation fails due to a missing customer-specified cluster OIDC provider. This update also allows the service to retrieve existing DNS name servers so that cluster p rovisioning operations are idempotent. To learn more, see the section called "AWS managed policy: ROSAInsta llerPolicy".	January 26, 2024
ROSASRESupportPolicy — Policy updated	ROSA updated the policy to allow the service to perform read operations on security groups using the DescribeS ecurityGroups API. To learn more, see the section called "AWS managed policy: ROSASRESupportPolicy".	January 22, 2024
ROSAImageRegistryO peratorPolicy — Policy updated	ROSA updated the policy to allow the Image Registry Operator to take actions on Amazon S3 buckets in Regions with 14-character names. To learn more, see the section called "AWS managed policy: ROSAImageRegistryO peratorPolicy".	December 12, 2023

Change	Description	Date
ROSAKubeControllerPolicy — Policy updated	ROSA updated the policy to allow the kube-controller-manager to describe Availability Zones, Amazon EC2 instances , route tables, security groups, VPCs, and subnets. To learn more, see the section called “AWS managed policy: ROSAKubeControllerPolicy”.	October 16, 2023
ROSAManageSubscription — Policy updated	ROSA updated the policy to add the ROSA with hosted control planes ProductId. To learn more, see the section called “AWS managed policy: ROSAManageSubscription”.	August 1, 2023
ROSAKubeControllerPolicy — Policy updated	ROSA updated the policy to allow the kube-controller-manager to create Network Load Balancers as Kubernetes service load balancers. Network Load Balancers provide greater ability to handle volatile workloads and support static IP addresses for the load balancer. To learn more, see the section called “AWS managed policy: ROSAKubeControllerPolicy”.	July 13, 2023

Change	Description	Date
ROSANodePoolManagementPolicy — New policy added	ROSA added a new policy to allow the NodePool controller to describe, run, and terminate Amazon EC2 instances managed as worker nodes. This policy also enables disk encryption of the worker node root volume using AWS KMS keys. To learn more, see the section called "AWS managed policy: ROSANodePoolManagementPolicy".	June 8, 2023
ROSAInstallerPolicy — New policy added	ROSA added a new policy to allow the installer to manage AWS resources that support cluster installation. To learn more, see the section called "AWS managed policy: ROSAInstallerPolicy".	June 6, 2023
ROSASRESupportPolicy — New policy added	ROSA added a new policy to allow Red Hat SREs to directly observe, diagnose and support AWS resources associated with ROSA clusters, including the ability to change ROSA cluster node state. To learn more, see the section called "AWS managed policy: ROSASRESupportPolicy".	June 1, 2023

Change	Description	Date
ROSAKMSProviderPolicy — New policy added	ROSA added a new policy to allow the built-in AWS Encryption Provider to manage AWS KMS keys to support etcd data encryption. To learn more, see the section called “AWS managed policy: ROSAKMSProviderPolicy”.	April 27, 2023
ROSAKubeControllerPolicy — New policy added	ROSA added a new policy to allow the kube controlle r to manage Amazon EC2, Elastic Load Balancing, and AWS KMS resources for ROSA with hosted control planes clusters. To learn more, see the section called “AWS managed policy: ROSAKubeC ontrollerPolicy”.	April 27, 2023
ROSAImageRegistryO peratorPolicy — New policy added	ROSA added a new policy to allow the Image Registry Operator to provision and manage resources for the ROSA in-cluster image registry and dependent services, including S3. To learn more, see the section called “AWS managed policy: ROSAImageRegistryO peratorPolicy”.	April 27, 2023

Change	Description	Date
ROSAControlPlaneOperatorPolicy — New policy added	ROSA added a new policy to allow the Control Plane Operator to manage Amazon EC2 and Route 53 resources for ROSA with hosted control planes clusters. To learn more, see the section called "AWS managed policy: ROSAControlPlaneOperatorPolicy".	April 24, 2023
ROSACloudNetworkConfigOperatorPolicy — New policy added	ROSA added a new policy to allow the Cloud Network Config Controller Operator to provision and manage networking resources for the ROSA cluster networking overlay. To learn more, see the section called "AWS managed policy: ROSACloudNetworkConfigOperatorPolicy".	April 20, 2023
ROSAIngressOperatorPolicy — New policy added	ROSA added a new policy to allow the Ingress Operator to provision and manage load balancers and DNS configurations for ROSA clusters. To learn more, see the section called "AWS managed policy: ROSAIngressOperatorPolicy".	April 20, 2023

Change	Description	Date
ROSAAmazonEBSCSIDriverOperatorPolicy — New policy added	ROSA added a new policy to allow the Amazon EBS CSI Driver Operator to install and maintain the Amazon EBS CSI driver on a ROSA cluster. To learn more, see the section called "AWS managed policy: ROSAAmazonEBSCSIDriverOperatorPolicy".	April 20, 2023
ROSAWorkerInstancePolicy — New policy added	ROSA added a new policy to allow the service to manage cluster resources. To learn more, see the section called "AWS managed policy: ROSAWorkerInstancePolicy".	April 20, 2023
ROSAManageSubscription – New policy added	ROSA added a new policy to grant the AWS Marketplace permissions required to manage the ROSA subscription. To learn more, see the section called "AWS managed policy: ROSAManageSubscription".	April 11, 2022
Red Hat OpenShift Service on AWS started tracking changes	Red Hat OpenShift Service on AWS started tracking changes for its AWS managed policies.	March 2, 2022

AWS managed policies for ROSA with HCP account roles

> **Note**
>
> These AWS managed policies are intended for use by ROSA with hosted control planes (HCP). ROSA classic clusters use customer managed IAM policies. For more information about ROSA classic policies, see the section called "ROSA classic account policies" and the section called "ROSA classic operator policies".

These AWS managed policies add permissions used by ROSA with hosted control planes (HCP) IAM roles. The permissions are required for Red Hat site reliability engineering (SRE) technical support, cluster installation, and control plane and compute functionality.

Topics

AWS managed policy: ROSAWorkerInstancePolicy

You can attach `ROSAWorkerInstancePolicy` to your IAM entities. Before you create a ROSA with hosted control planes cluster, you must first attach this policy to a worker IAM role.

Permissions details

This policy includes the following permissions that allow the ROSA service to complete the following tasks:

- `ec2` — Review AWS Region and Amazon EC2 instance details as part of the lifecycle management of worker nodes in a ROSA cluster.

To view the full JSON policy document, see ROSAWorkerInstancePolicy in the *AWS Managed Policy Reference Guide*.

AWS managed policy: ROSASRESupportPolicy

You can attach `ROSASRESupportPolicy` to your IAM entities.

Before you create a ROSA with hosted control planes cluster, you must first attach this policy to a support IAM role. This policy grants required permissions to Red Hat site reliability engineers (SREs) to directly observe, diagnose, and support AWS resources associated with ROSA clusters, including the ability to change ROSA cluster node state.

Permissions details

This policy includes the following permissions that allow Red Hat SREs to complete the following tasks:

- `cloudtrail` — Read AWS CloudTrail events and trails relevant to the cluster.
- `cloudwatch` — Read Amazon CloudWatch metrics relevant to the cluster.
- `ec2` — Read, describe, and review Amazon EC2 components related to the cluster's health such as security groups, VPC endpoint connections, and volume status. Launch, stop, reboot, and terminate Amazon EC2 instances.
- `elasticloadbalancing` — Read, describe, and review Elastic Load Balancing parameters related to the cluster's health.
- `iam` — Evaluate IAM roles that relate to the cluster's health.
- `route53` — Review DNS settings related to the cluster's health.
- `sts` — `DecodeAuthorizationMessage` — Read IAM messages for debugging purposes.

To view the full JSON policy document, see ROSASRESupportPolicy in the *AWS Managed Policy Reference Guide*.

AWS managed policy: ROSAInstallerPolicy

You can attach `ROSAInstallerPolicy` to your IAM entities.

Before you create a ROSA with hosted control planes cluster, you must first attach this policy to an IAM role named `[Prefix]-ROSA-Worker-Role`. This policy allows entities to add any role that follows the `[Prefix]-ROSA-Worker-Role` pattern to an instance profile. This policy grants necessary permissions to the installer to manage AWS resources that support ROSA cluster installation.

Permissions details

This policy includes the following permissions that allow the installer to complete the following tasks:

- `ec2` — Run Amazon EC2 instances using AMIs hosted in AWS accounts owned and managed by Red Hat. Describe Amazon EC2 instances, volumes, and network resources associated with Amazon EC2 nodes. This is required so that the Kubernetes control plane can join instances to a cluster. This is also required so that the cluster can evaluate its presence within Amazon VPC. Tag subnets using tag keys matching `"kubernetes.io/cluster/*"`. This is required to ensure that the load balancer used for cluster ingress is created only in applicable subnets.
- `elasticloadbalancing` — Add load balancers to target nodes on a cluster. Remove load balancers from target nodes on a cluster. This permission is required so that the Kubernetes control plane can dynamically provision load balancers requested by Kubernetes services and OpenShift application services.
- `kms` — Read an AWS KMS key, create and manage grants to Amazon EC2, and return a unique symmetric data key for use outside of AWS KMS. This is required for the use of encrypted `etcd` data when `etcd` encryption is enabled at cluster creation.
- `iam` — Validate IAM roles and policies. Dynamically provision and manage Amazon EC2 instance profiles relevant to the cluster. Add tags to an IAM instance profile by using the `iam:TagInstanceProfile` permission. Provide installer error messages when cluster installation fails due to a missing customer-specified cluster OIDC provider.
- `route53` — Manage Route 53 resources needed to create clusters.
- `servicequotas` — Evaluate service quotas required to create a cluster.
- `sts` — Create temporary AWS STS credentials for ROSA components. Assume the credentials for cluster creation.
- `secretsmanager` — Read a secret value to securely allow customer-managed OIDC configuration as part of cluster provisioning.

To view the full JSON policy document, see ROSAInstallerPolicy in the *AWS Managed Policy Reference Guide*.

AWS managed policies for ROSA with HCP operator roles

Note

These AWS managed policies are intended for use by ROSA with hosted control planes (HCP). ROSA classic clusters use customer managed IAM policies. For more information about ROSA classic policies, see the section called “ROSA classic account policies” and the section called “ROSA classic operator policies”.

These AWS managed policies add permissions used by ROSA with hosted control planes (HCP) IAM roles. The permissions are required for OpenShift operators on the ROSA with HCP cluster to manage cluster nodes.

Topics

AWS managed policy: ROSAAmazonEBSCSIDriverOperatorPolicy

You can attach `ROSAAmazonEBSCSIDriverOperatorPolicy` to your IAM entities. You must attach this policy to an operator IAM role to allow a ROSA with hosted control planes cluster to make calls to other AWS services. A unique set of operator roles is required for each cluster.

This policy grants necessary permissions to the Amazon EBS CSI Driver Operator to install and maintain the Amazon EBS CSI driver on a ROSA cluster. For more information about the operator, see aws-ebs-csi-driver operator in the OpenShift GitHub documentation.

Permissions details

This policy includes the following permissions that allow the Amazon EBS Driver Operator to complete the following tasks:

- `ec2` — Create, modify, attach, detach, and delete Amazon EBS volumes that are attached to Amazon EC2 instances. Create and delete Amazon EBS volume snapshots and list Amazon EC2 instances, volumes, and snapshots.

To view the full JSON policy document, see ROSAAmazonEBSCSIDriverOperatorPolicy in the *AWS Managed Policy Reference Guide*.

AWS managed policy: ROSAIngressOperatorPolicy

You can attach `ROSAIngressOperatorPolicy` to your IAM entities. You must attach this policy to an operator IAM role to allow a ROSA with hosted control planes cluster to make calls to other AWS services. A unique set of operator roles is required for each cluster.

This policy grants required permissions to the Ingress Operator to provision and manage load balancers and DNS configurations for ROSA clusters. The policy allows read access to tag values. The operator then filters the tag values for Route 53 resources to discover hosted zones. For more information about the operator, see OpenShift Ingress Operator in the OpenShift GitHub documentation.

Permissions details

This policy includes the following permissions that allow the Ingress Operator to complete the following tasks:

- `elasticloadbalancing` — Describe the state of provisioned load balancers.
- `route53` — List Route 53 hosted zones and edit records that manage the DNS controlled by the ROSA cluster.
- `tag` — Manage tagged resources by using the `tag:GetResources` permission.

To view the full JSON policy document, see ROSAIngressOperatorPolicy in the *AWS Managed Policy Reference Guide*.

AWS managed policy: ROSAImageRegistryOperatorPolicy

You can attach `ROSAImageRegistryOperatorPolicy` to your IAM entities. You must attach this policy to an operator IAM role to allow a ROSA with hosted control planes cluster to make calls to other AWS services. A unique set of operator roles is required for each cluster.

This policy grants required permissions to the Image Registry Operator to provision and manage resources for the ROSA in-cluster image registry and dependent services, including S3. This is required so that the operator can install and maintain the internal registry of a ROSA cluster. For more information about the operator, see Image Registry Operator in the OpenShift GitHub documentation.

Permissions details

This policy includes the following permissions that allow the Image Registry Operator to complete the following actions:

- `s3` — Manage and evaluate Amazon S3 buckets as persistent storage for container image content and cluster metadata.

To view the full JSON policy document, see ROSAImageRegistryOperatorPolicy in the *AWS Managed Policy Reference Guide*.

AWS managed policy: ROSACloudNetworkConfigOperatorPolicy

You can attach `ROSACloudNetworkConfigOperatorPolicy` to your IAM entities. You must attach this policy to an operator IAM role to allow a ROSA with hosted control planes cluster to make calls to other AWS services. A unique set of operator roles is required for each cluster.

This policy grants required permissions to the Cloud Network Config Controller Operator to provision and manage networking resources for the ROSA cluster networking overlay. The operator uses these permissions to manage private IP addresses for Amazon EC2 instances as part of the ROSA cluster. For more information about the operator, see Cloud-network-config-controller in the OpenShift GitHub documentation.

Permissions details

This policy includes the following permissions that allow the Cloud Network Config Controller Operator to complete the following tasks:

- `ec2` — Read, assign, and describe configurations for connecting Amazon EC2 instances, Amazon VPC subnets, and elastic network interfaces in a ROSA cluster.

To view the full JSON policy document, see ROSACloudNetworkConfigOperatorPolicy in the *AWS Managed Policy Reference Guide*.

AWS managed policy: ROSAKubeControllerPolicy

You can attach `ROSAKubeControllerPolicy` to your IAM entities. You must attach this policy to an operator IAM role to allow a ROSA with hosted control planes cluster to make calls to other AWS services. A unique set of operator roles is required for each cluster.

This policy grants required permissions to the kube controller to manage Amazon EC2, Elastic Load Balancing, and AWS KMS resources for a ROSA with hosted control planes cluster. For more information about this controller, see Controller architecture in the OpenShift documentation.

Permissions details

This policy includes the following permissions that allow the kube controller to complete the following tasks:

- `ec2` — Create, delete, and add tags to Amazon EC2 instance security groups. Add inbound rules to security groups. Describe Availability Zones, Amazon EC2 instances, route tables, security groups, VPCs, and subnets.
- `elasticloadbalancing` — Create and manage load balancers and their policies, create and manage load balancer listeners, register targets with target groups and manage target groups, register and de-register Amazon EC2 instances with a load balancer, and add tags to load balancers.
- `kms` — Retrieve detailed information about an AWS KMS key. This is required for the use of encrypted `etcd` data when `etcd` encryption is enabled at cluster creation.

To view the full JSON policy document, see ROSAKubeControllerPolicy in the *AWS Managed Policy Reference Guide*.

AWS managed policy: ROSANodePoolManagementPolicy

You can attach `ROSANodePoolManagementPolicy` to your IAM entities. You must attach this policy to an operator IAM role to allow a ROSA with hosted control planes cluster to make calls to other AWS services. A unique set of operator roles is required for each cluster.

This policy grants required permissions to the NodePool controller to describe, run, and terminate Amazon EC2 instances managed as worker nodes. This policy also grants permissions to allow for disk encryption of the worker node root volume using AWS KMS keys. For more information about this controller, see Controller architecture in the OpenShift documentation.

Permissions details

This policy includes the following permissions that allow the NodePool controller to complete the following tasks:

- `ec2` — Run Amazon EC2 instances using AMIs hosted in AWS accounts owned and managed by Red Hat. Manage EC2 lifecycles in the ROSA cluster. Dynamically create and integrate worker nodes with Elastic Load Balancing, Amazon VPC, Route 53, Amazon EBS, and Amazon EC2.
- `iam` — Use Elastic Load Balancing via the service-linked role named `AWSServiceRoleForElasticLoadBalancing`. Assign roles to Amazon EC2 instance profiles.
- `kms` — Read an AWS KMS key, create and manage grants to Amazon EC2, and return a unique symmetric data key for use outside of AWS KMS. This is required to allow for disk encryption of the worker node root volume.

To view the full JSON policy document, see ROSANodePoolManagementPolicy in the *AWS Managed Policy Reference Guide*.

AWS managed policy: ROSAKMSProviderPolicy

You can attach `ROSAKMSProviderPolicy` to your IAM entities. You must attach this policy to an operator IAM role to allow a ROSA with hosted control planes cluster to make calls to other AWS services. A unique set of operator roles is required for each cluster.

This policy grants required permissions to the built-in AWS Encryption Provider to manage AWS KMS keys that support `etcd` data encryption. This policy allows Amazon EC2 to use KMS keys that the AWS Encryption Provider provides to encrypt and decrypt `etcd` data. For more information about this provider, see AWS Encryption Provider in the Kubernetes GitHub documentation.

Permissions details

This policy includes the following permissions that allow the AWS Encryption Provider to complete the following tasks:

- `kms` — Encrypt, decrypt, and retrieve an AWS KMS key. This is required for the use of encrypted `etcd` data when `etcd` encryption is enabled at cluster creation.

To view the full JSON policy document, see ROSAKMSProviderPolicy in the *AWS Managed Policy Reference Guide*.

AWS managed policy: ROSAControlPlaneOperatorPolicy

You can attach `ROSAControlPlaneOperatorPolicy` to your IAM entities. You must attach this policy to an operator IAM role to allow a ROSA with hosted control planes cluster to make calls to other AWS services. A unique set of operator roles is required for each cluster.

This policy grants required permissions to the Control Plane Operator to manage Amazon EC2 and Route 53 resources for ROSA with hosted control planes clusters. For more information about this operator, see Controller architecture in the OpenShift documentation.

Permissions details

This policy includes the following permissions that allow the Control Plane Operator to complete the following tasks:

- `ec2` — Create and manage Amazon VPC endpoints.
- `route53` — List and change Route 53 record sets and list hosted zones.

To view the full JSON policy document, see ROSAControlPlaneOperatorPolicy in the *AWS Managed Policy Reference Guide*.

Troubleshooting ROSA identity and access

Use the following information to help you diagnose and fix common issues that you might encounter when working with ROSA and IAM.

AWS Organizations service control policy denies required AWS Marketplace permissions

If your AWS Organizations service control policy (SCP) doesn't allow the required AWS Marketplace subscription permissions when you attempt to enable ROSA, the following console error occurs:

```
An error occurred while enabling ROSA, because a service control policy (SCP) is
 denying required permissions. Contact your management account administrator, and
 consult the documentation for troubleshooting.
```

If you receive this error, then you must contact your administrator for assistance. Your administrator is the person that manages the accounts for your organization. Ask that person to do the following:

1. Configure the SCP to allow `aws-marketplace:Subscribe`, `aws-marketplace:Unsubscribe`, and `aws-marketplace:ViewSubscriptions` permissions. For more information, see Updating an SCP in the *AWS Organizations User Guide*.
2. Enable ROSA in the organization's management account.

3. Share the ROSA subscription to member accounts that require access within the organization. For more information, see Sharing subscriptions in an organization in the *AWS Marketplace Buyer Guide*.

User or role does not have the required AWS Marketplace permissions

If your IAM principal doesn't have the required AWS Marketplace subscription permissions when you attempt to enable ROSA, the following console error occurs:

```
An error occurred while enabling ROSA, because your user or role does not have the
 required permissions.
```

To resolve this issue, follow these steps:

1. Go to the IAM console and attach the AWS managed policy `ROSAManageSubscription` to your IAM identity. For more information, see ROSAManageSubscription in the *AWS Managed Policy Reference Guide*.
2. Follow the procedure in the section called "Step 1: Enable ROSA and configure prerequisites".

If you don't have permission to view or update your permission set in IAM or you receive an error, then you must contact your administrator for assistance. Ask that person to attach `ROSAManageSubscription` to your IAM identity and follow the procedure in the section called "Step 1: Enable ROSA and configure prerequisites". When an administrator performs this action, it enables ROSA by updating the permission set for all IAM identities under the AWS account.

Required AWS Marketplace permissions blocked by an administrator

If your account administrator blocked the required AWS Marketplace subscription permissions, the following console error occurs when you attempt to enable ROSA:

```
An error occurred while enabling ROSA because required permissions have been blocked by
 an administrator. ROSAManageSubscription includes the permissions required to enable
 ROSA. Consult the documentation and try again.
```

If you receive this error, then you must contact your administrator for assistance. Ask that person to do the following:

1. Go to the ROSA console and attach the AWS managed policy `ROSAManageSubscription` to your IAM identity. For more information, see ROSAManageSubscription in the *AWS Managed Policy Reference Guide*.
2. Follow the procedure in the section called "Step 1: Enable ROSA and configure prerequisites" to enable ROSA. This procedure enables ROSA by updating the permission set for all IAM identities under the AWS account.

Error creating load balancer: AccessDenied

If you haven't created a load balancer, the `AWSServiceRoleForElasticLoadBalacing` service-linked role may not exist in your account. The following error occurs if you attempt to create a ROSA cluster without the `AWSServiceRoleForElasticLoadBalacing` role in your account:

```
Error creating network Load Balancer: AccessDenied
```

To resolve this issue, follow these steps:

1. Check if your account has the `AWSServiceRoleForElasticLoadBalancing` role.

   ```
   aws iam get-role --role-name "AWSServiceRoleForElasticLoadBalancing"
   ```

2. If you don't have this role, follow the instructions to create the role found in Create the service-linked role in the *Elastic Load Balancing User Guide*.

Resilience in ROSA

AWS global infrastructure resilience

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected through low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

ROSA provides customers with the option to run the Kubernetes control plane and data plane in a single AWS Availability Zone, or across multiple Availability Zones. While Single-AZ clusters can be useful for experimentation, customers are encouraged to run their workloads in more than one Availability Zone. This ensures that applications can withstand even a complete Availability Zone failure - a very rare event in itself.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

ROSA cluster resilience

The ROSA control plane consists of at least three OpenShift control plane nodes. Each control plane node is made up of an API server instance, an `etcd` instance, and controllers. In the event of a control plane node failure, all API requests are automatically routed to the other available nodes to ensure cluster availability.

The ROSA data plane consists of at least two OpenShift infrastructure nodes and two OpenShift worker nodes. Infrastructure nodes run pods that support OpenShift cluster infrastructure components such as the default router, the built-in OpenShift registry, and the components for cluster metrics and monitoring. OpenShift worker nodes run end-user application pods.

Red Hat site reliability engineers (SREs) fully manage the control plane and infrastructure nodes. Red Hat SREs proactively monitor the ROSA cluster, and are responsible for replacing any failed control plane nodes and infrastructure nodes. For more information, see the section called "Responsibilities".

> ⚠ **Important**
>
> Because ROSA is a managed service, Red Hat is responsible for managing the underlying AWS infrastructure that ROSA uses. Customers should not attempt to manually shut down the Amazon EC2 instances that ROSA uses from the AWS console or AWS CLI. This action can lead to customer data loss.

If a worker node fails on the data plane, the control plane relocates unscheduled pods to the functioning worker node(s) until the failed node is recovered or replaced. Failed worker nodes can be replaced manually or automatically by enabling automatic scaling of machines in a cluster. For more information, see Cluster autoscaling in the Red Hat documentation.

Customer-deployed application resilience

Although ROSA provides many protections to ensure high availability of the service, customers are responsible for building their deployed applications for high availability to protect workloads against downtime. For more information, see About availability for ROSA in the Red Hat documentation.

Infrastructure security in ROSA

As a managed service, Red Hat OpenShift Service on AWS is protected by the AWS global network security. For information about AWS security services and how AWS protects infrastructure, see AWS Cloud Security. To design your AWS environment using the best practices for infrastructure security, see Infrastructure Protection in *Security Pillar — AWS Well‐Architected Framework*.

You use AWS published API calls to access ROSA through the AWS network. Clients must support the following:

- Transport Layer Security (TLS). We require TLS 1.2 and recommend TLS 1.3.
- Cipher suites with perfect forward secrecy (PFS) such as DHE (Ephemeral Diffie-Hellman) or ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the AWS Security Token Service (AWS STS) to generate temporary security credentials to sign requests.

Cluster network isolation

Red Hat site reliability engineers (SREs) are responsible for the ongoing management and network security of the cluster and underlying application platform. For more information on Red Hat responsibilities for ROSA, see the section called “Responsibilities”.

When you create a new cluster, ROSA provides the option of creating a public Kubernetes API server endpoint and application routes or a private Kubernetes API endpoint and application routes. This connection is used to communicate with your cluster (using OpenShift management tools such as the ROSA CLI and OpenShift CLI). A private connection allows all communication between your nodes and the API server to stay within your VPC. If you enable private access to the API server and application routes, you must use an existing VPC and AWS PrivateLink to connect the VPC to the OpenShift backend service.

Kubernetes API server access is secured using a combination of AWS Identity and Access Management (IAM) and native Kubernetes role-based access control (RBAC). For more information on Kubernetes RBAC, see Using RBAC Authorization in the Kubernetes documentation.

ROSA allows you to create secured application routes using several types of TLS termination to serve certificates to the client. For more information, see Secured routes in the Red Hat documentation.

If you create a ROSA cluster in an existing VPC, you specify the VPC subnets and Availability Zones for your cluster to use. You also define the CIDR ranges for the cluster network to use, and match these CIDR ranges to the VPC subnets. For more information, see CIDR range definitions in the Red Hat documentation.

For clusters that use the public API endpoint, ROSA requires that your VPC is configured with a public and private subnet for each Availability Zone that you want the cluster deployed into. For clusters that use the private API endpoint, only private subnets are required.

If you are using an existing VPC, you can configure your ROSA clusters to use an HTTP or HTTPS proxy server during or after cluster creation to encrypt cluster web traffic, adding another layer of security for your data. When you enable a proxy, the core cluster components are denied direct access to the internet. The proxy does not deny internet access for user workloads. For more information, see Configuring a cluster-wide proxy in the Red Hat documentation.

Pod network isolation

If you are a cluster administrator, you can define network policies at the pod level that restrict traffic to pods in your ROSA cluster.

ROSA service quotas

Red Hat OpenShift Service on AWS (ROSA) uses service quotas for Amazon EC2, Amazon Virtual Private Cloud (Amazon VPC), Amazon Elastic Block Store (Amazon EBS), and Elastic Load Balancing (ELB) to provision clusters.

Required minimum quotas for ROSA

For the following Amazon EC2 and Amazon EBS quotas, ROSA requires a higher quota than the default service provides. To use ROSA, you may need to request an increase for these quotas. For more information, see Requesting a quota increase in the *Service Quotas User Guide*.

⚠ **Important**

For On-Demand Standard (A, C, D, H, I, M, R, T, Z) Amazon EC2 instances, the default value of 5 vCPUs is not sufficient to create ROSA clusters. ROSA requires 100 vCPUs or greater for cluster creation. To increase this quota, open the Service Quotas console and request a quota increase.

ⓘ **Note**

You can check your quotas using the AWS SDKs, but the SDK calculation does not include existing ROSA resources. The quota check in the SDK may pass, and ROSA cluster creation may fail. To fix this issue, open the Service Quotas console and request a quota increase.

Name	Service code	Default	Minimum required	Adjustable	Description
Running On-Demand Standard (A, C, D, H, I, M, R, T, Z) instances	ec2	5	100	Yes	Maximum number of vCPUs assigned to the Running On-Demand

Name	Service code	Default	Minimum required	Adjustable	Description
					Standard (A, C, D, H, I, M, R, T, Z) instances. The default value of 5 vCPUs is not sufficient to create ROSA clusters. ROSA requires 100 vCPUs for cluster creation.

Name	Service code	Default	Minimum required	Adjustable	Description
Storage for General Purpose SSD (gp3) volumes, in TiB	ebs	50	300	Yes	The maximum aggregated amount of storage, in TiB, that can be provision ed across General Purpose SSD (gp3) volumes in this Region. 300 TiB of storage is required for optimal performance.

Name	Service code	Default	Minimum required	Adjustable	Description
Storage for General Purpose SSD (gp2) volumes, in TiB	ebs	50	300	Yes	The maximum aggregated amount of storage, in TiB, that can be provisioned across General Purpose SSD (gp2) volumes in this Region. 300 TiB of storage is required for optimal performance.

Name	Service code	Default	Minimum required	Adjustable	Description
Storage for Provisioned IOPS SSD (io1) volumes, in TiB	ebs	50	300	Yes	The maximum aggregated amount of storage, in TiB, that can be provisioned across Provisioned IOPS SSD (io1) volumes in this Region. 300 TiB of storage is required for optimal performance.

Note

The default values are the initial quotas set by AWS, which are separate from the actual applied quota value and maximum possible service quota. For more information, see Terminology in Service Quotas in the *Service Quotas User Guide*.

Default quotas for ROSA

ROSA uses the following default quotas for Amazon EC2, Amazon VPC, Amazon EBS, and Elastic Load Balancing. For information on increasing quotas, see Requesting a quota increase in the *Service Quotas User Guide*.

Amazon EC2

- EC2-VPC Elastic IPs

Amazon VPC

- VPCs per Region
- Network interfaces per Region
- Internet gateways per Region

Amazon EBS

- Snapshots per Region
- IOPS for Provisioned IOPS SSD (io1) volumes

Elastic Load Balancing

- Application Load Balancers per Region
- Classic Load Balancers per Region

AWS services integrated with ROSA

ROSA works with other AWS services to provide additional solutions for your business challenges. This topic identifies services that either use ROSA to add functionality, or services that ROSA uses to perform tasks.

Topics

How ROSA works with AWS Marketplace

AWS Marketplace is a curated digital catalog that you can use to find, buy, deploy, and manage third-party software, data, and services that you need to build solutions and run your business. AWS Marketplace simplifies software licensing and procurement with flexible pricing options and multiple deployment methods.

ROSA uses AWS Marketplace for service metering and billing. ROSA classic is metered and billed through an AWS Marketplace Amazon Machine Image (AMI)-based product, whereas ROSA with hosted control planes (HCP) is metered and billed through an AWS Marketplace software as a service (SaaS)-based product.

This page explains how ROSA works with AWS Marketplace for payments, billing, subscriptions, and contract purchases.

Terminology

This page uses the following terms when discussing ROSA's integration with AWS Marketplace.

Amazon Machine Image (AMI)

An image of a server, including an operating system and additional software, that runs on AWS.

AMI subscription

In AWS Marketplace, AMI-based software products such as ROSA classic use an hourly with annual subscription pricing model. Hourly pricing is the default pricing model, but you have the option to purchase a year's worth of usage upfront for one Amazon EC2 instance type.

SaaS subscription

In AWS Marketplace, software-as-a-service (SaaS) products such as ROSA with HCP adopt a usage-based subscription model. The software seller tracks your usage and you pay only for what you use.

Public offer

Public offers allow you to purchase AWS Marketplace software and services directly from the AWS Management Console.

Private offer

Private offers are a purchasing program that allow sellers and buyers to negotiate custom prices and end user licensing agreement (EULA) terms for purchases in AWS Marketplace.

ROSA service fees

Fees that ROSA charges for the OpenShift software and cluster management by Red Hat site reliability engineers (SREs). ROSA service fees are metered through AWS Marketplace and appear on your AWS bill.

AWS infrastructure fees

Standard fees that AWS charges for the AWS services underlying ROSA clusters, including Amazon EC2, Amazon EBS, Amazon S3, and Elastic Load Balancing. Fees are metered through the AWS service being used and appear on your AWS bill.

ROSA payments and billing

ROSA integrates with AWS Marketplace to enable metering and billing of ROSA service fees. ROSA service fees cover access to OpenShift software and cluster management by Red Hat site reliability engineers (SREs). ROSA service fees are uniform across all supported AWS standard Regions. ROSA with HCP service fees accrue on demand by default at a flat hourly rate based on the number of running clusters and worker node vCPUs running in those clusters. ROSA classic service fees accrue on demand based on the number of worker node vCPUs. ROSA classic does not charge service fees for the control plane or required infrastructure nodes.

ROSA customers also pay standard AWS infrastructure fees for the AWS services underlying ROSA clusters, including Amazon EC2, Amazon EBS, Amazon S3, and Elastic Load Balancing. AWS infrastructure fees are a separate billing item from the ROSA service fees that are metered through

AWS Marketplace. AWS infrastructure fees vary by AWS Region and are based on hourly usage by default. For additional AWS infrastructure cost savings, you can purchase Amazon EC2 savings plans or reserved instances. For more information, see Compute Savings Plans and Reserved Instances in the *Amazon EC2 User Guide*.

ROSA does not charge fees until you create a ROSA cluster, or purchase a ROSA contract. For more information, see Red Hat OpenShift Service on AWS pricing.

You can view ROSA service fees and AWS infrastructure fees and manage payments in the AWS Billing console. You can also view your costs and monitor usage using the AWS Cost Explorer Service interface free of charge. For more information, see Viewing your bill in the *AWS Billing and Cost Management User Guide* and Analyzing your costs with AWS Cost Explorer Service in the *AWS Cost Management User Guide*.

Subscribing to ROSA Marketplace listings through the console

When you enable ROSA in the ROSA console, your AWS account is subscribed to the ROSA classic and ROSA with HCP listings on AWS Marketplace. There is no fee for enabling ROSA subscriptions.

For AWS Organizations users, ROSA allows you to share ROSA classic subscriptions with other accounts in your organization. For more information, see Sharing subscriptions in an organization in the *AWS Marketplace Buyer Guide*.

ROSA contracts

ROSA uses AWS Marketplace to provide optional contracts for ROSA with HCP and ROSA classic. Contracts provide savings on ROSA worker node service fees. ROSA contracts do not impact the fees charged for AWS infrastructure.

12-month contracts

You can purchase 12-month public offer contracts for ROSA classic and ROSA with HCP from the ROSA console.

> **Note**
>
> ROSA classic must be enabled on your account before you can purchase 12-month contracts from the console.

Note

12-month contracts cannot be transferred to a private offer.

Purchasing a ROSA classic 12-month contract

When you purchase a ROSA classic 12-month contract, you make a upfront payment for an annual term and pay no hourly service fee for the next 12 months for the covered instances. Contract cost is based on the Amazon EC2 instance type and number of instances that you select. The contract does not cover the AWS infrastructure fees that ROSA charges for the underlying AWS services that are used. For more information, see Red Hat OpenShift Service on AWS Pricing.

The contract covers only the instance types that you specify during contract creation (m5.xlarge for example). You can purchase additional 12-month contracts for cost savings on more than one Amazon EC2 instance type. Usage outside of your 12-month contract incurs ROSA service fees at the on-demand rate.

Note

ROSA classic 12-month contracts do not auto renew.

To purchase a 12-month contract for ROSA classic

Note

If you are using the ROSA console in a Region that does not yet support ROSA with HCP, this workflow is not yet available. For a list of Regions that support ROSA with HCP, see the section called “Differences between ROSA with HCP and ROSA classic”.
To purchase ROSA classic contracts in Regions without ROSA with HCP support, go to the ROSA console and choose **Purchase a software contract and view existing contracts**.

1. Go to the ROSA console.
2. On the left navigation pane, choose **Contracts**.
3. Choose **Contracts for ROSA classic**.

4. Choose **Purchase contract**.
5. Select the EC2 instance type and number of instances that you need.
6. Choose **Review contract**.
7. Review the contract details and choose **Purchase contract**.

> **Note**
>
> ROSA 12-month contracts cannot be downgraded or cancelled after creation using the console. If you need to downgrade or cancel the contract during the active contract duration, go to AWS Support Center and open a support case.

Purchasing a ROSA with HCP 12-month contract

When you enable ROSA with HCP in the console, a no-cost 12-month ROSA with HCP contract is initially created on your account to facilitate on-demand billing. If you choose to purchase a ROSA with HCP contract to save on worker node service fees, the initial contract is modified to cover usage costs for the worker node vCPUs and control planes that you specify.

When you purchase a ROSA with HCP 12-month contract, you make an upfront payment for an annual term and pay no hourly usage fee for the next 12 months for the covered worker node vCPUs and control planes. Contract cost is based on the number of worker node vCPUs and control planes that you select. The contract covers only the worker node vCPUs and control planes that you specify during contract creation. The contract does not cover the AWS infrastructure fees that ROSA charges for the underlying AWS services that are used. For more information, see Red Hat OpenShift Service on AWS Pricing.

Monthly usage quota

Upon purchase, your prepaid vCPUs and control planes are converted to a monthly usage quota. Hourly on-demand usage rates apply for vCPU and control plane usage that exceeds the monthly quota. ROSA with HCP uses the following formulas to calculate the monthly quota associated with the contract:

- **Worker node vCPUs:** number of vCPUs x 24 hours x 365 days / 12 months
- **Control planes:** number of control planes x 24 hours x 365 days / 12 months

For example, a purchase of 4,000 worker node vCPUs and 8 control planes would convert to a monthly quota of 2,920,000 worker node vCPU hours and 5,840 control plane hours consumable per month.

To purchase a ROSA with HCP 12-month contract

> **Note**
>
> If you are using the Red Hat OpenShift Service on AWS console in a Region that doesn't yet support ROSA with hosted control planes, this workflow is not yet available. For a list of Regions that support ROSA with HCP, see the section called "Differences between ROSA with HCP and ROSA classic".

1. Go to the ROSA console.
2. On the left navigation pane, choose **Contracts**.
3. Choose **Contracts for ROSA with HCP**.
4. Choose **Purchase contract**.
5. Enter the number of vCPUs to purchase. Specify in multiples of 4.
6. Enter the number of control planes to purchase.
7. Choose **Review contract**.
8. Review the contract details and choose **Purchase contract**.

> **Note**
>
> ROSA 12-month contracts cannot be downgraded or cancelled after creation using the console. If you need to downgrade or cancel the contract during the active contract duration, go to AWS Support Center and open a support case.

Upgrading a ROSA with HCP 12-month contract

You can upgrade your active ROSA with HCP 12-month contract at any time with additional worker node vCPUs and control planes. When you upgrade your ROSA with HCP 12-month contract, you make an upfront prorated payment for the added resources. Prorated amounts are calculated based on the number of days remaining on the contract. The contract covers only the worker node vCPUs and control planes that you specify during contract creation. Contract upgrades do not impact the fees charged for AWS infrastructure.

Upon upgrade, the added vCPUs and control planes are converted to a monthly usage quota using the same formulas as the original contract purchase. Hourly on-demand usage rates apply for vCPU and control plane usage that exceeds the monthly quota. For more information, see the section called “Monthly usage quota”.

To upgrade a ROSA with HCP 12-month contract

1. Go to the ROSA console.
2. On the left navigation pane, choose **Contracts**.
3. Choose **Contracts for ROSA with HCP**.
4. Choose **Upgrade**.
5. Enter the number of vCPUs to add. Specify in multiples of 4.
6. Enter the number of control planes to add to the contract.
7. Choose **Review upgrade**.
8. Review the contract details and choose **Purchase upgrade**.

> ⓘ **Note**
>
> ROSA classic 12-month contracts cannot be upgraded. Additional 12-month ROSA classic contracts can be purchased at any time using the ROSA console.

Obtaining a private offer

You can request an AWS Marketplace private offer for ROSA with HCP or ROSA classic to receive product pricing and end user licensing agreement (EULA) terms negotiated with Red Hat. For more information, see Private offers in the *AWS Marketplace Buyer Guide*.

To obtain a ROSA private offer

> **Note**
>
> If you are an AWS Organizations user and received a private offer that was issued to your payer and member accounts, follow the procedure below to subscribe to ROSA directly on each account in your organization.
>
> If you receive a ROSA classic private offer that was only issued to the AWS Organizations payer account, you will need to share the subscription with members accounts in your organization. For more information, see Sharing subscriptions in an organization in the *AWS Marketplace Buyer Guide*.

1. Once a private offer has been issued, sign in to the AWS Marketplace console.
2. Open the email with a ROSA private offer link.
3. Follow the link to directly access the private offer.

   > **Note**
   >
   > Following this link before logging in to the correct account will result in a **Page note found** (404) error.

4. Review the terms and conditions.
5. Choose **Accept terms**.

   > **Note**
   >
   > If an AWS Marketplace private offer is not accepted, the ROSA service fees from AWS Marketplace will continue to be billed at the public hourly rate.

6. To verify the offer details, select **Show details** in the product listing.
7. To get started using ROSA, choose **Continue to configuration**. You will be redirected to the ROSA console.

Private Marketplace

Private Marketplace enables administrators to build customized digital catalogs of approved products from AWS Marketplace. Administrators can create unique sets of vetted software available in AWS Marketplace for AWS organizational units or different AWS accounts within their organization to purchase.

If your organization uses a private marketplace, an administrator must add the AWS Marketplace listings for ROSA to the private marketplace before users can enable the service. For more information, see Getting started with private marketplace in the *AWS Marketplace Buyer Guide*.

Troubleshooting

The following documentation covers how to troubleshoot issues that might occur when enabling ROSA and provisioning ROSA clusters.

Topics

- Support for ROSA

Support for ROSA

With ROSA, you can receive troubleshooting support from AWS Support and the Red Hat support teams. Support cases can be opened with either organization, and are routed to the correct team to resolve your issue.

AWS Support

An AWS Developer Support plan is required to open ROSA technical cases, but an AWS Business or Enterprise On-Ramp Support plan is recommended for continuous access to ROSA technical support and architectural guidance. Red Hat uses the AWS Support API to open cases for customers when necessary. AWS Business Support and AWS Enterprise On-Ramp enable continuous phone, web, and chat access to support engineers. For more information about AWS Support plans, see AWS Support.

For steps to enable an AWS Support plan, see How do I sign up for an AWS Support plan?

For information about creating an AWS Support case, see Creating support cases and case management.

Red Hat Support

ROSA includes Red Hat Premium Support. To receive Red Hat Premium Support, navigate to the Red Hat Customer Portal and use the support case tool to create a support ticket. For more information, see How to engage with Red Hat support.

Troubleshoot ROSA cluster creation issues

This section contains solutions to issues you may have when creating ROSA clusters.

With ROSA, you can also receive troubleshooting support from AWS Support and the Red Hat support teams. For more information, see the section called "Support for ROSA".

Topics

- Access ROSA cluster debug logs
- ROSA cluster fails AWS service quota check during cluster creation
- Troubleshoot ROSA CLI expired offline access tokens

Access ROSA cluster debug logs

To begin to troubleshoot issues with your application, first review the debug logs. The ROSA CLI debug logs provide details on the error messages that are produced when a cluster fails to create.

To display cluster debug information, run the following ROSA CLI command. In the command, replace `<cluster_name>` with the name of your cluster.

```
rosa describe cluster -c <cluster_name> --debug
```

ROSA cluster fails AWS service quota check during cluster creation

Description

To use ROSA, the service quotas for your account may need increased. For more information, see *Service quotas*.

Solution

1. Run the following command to identify your account's quotas.

   ```
   rosa verify quota
   ```

   > **Note**
   >
   > Quotas are different in different AWS Regions. Make sure to verify each of the quotas for your Regions.

2. If you need to increase your quota, navigate to the Service Quotas console.
3. On the navigation pane, choose **AWS services**.

4. Choose the service that needs a quota increase.
5. Select the quota that needs to be increased and choose **Request quota increase**.
6. For **Request quota increase**, enter the total amount that you want the quota to be and choose **Request**.

Troubleshoot ROSA CLI expired offline access tokens

Description

If you use the ROSA CLI and your api.openshift.com offline access token expires, an error message appears. This happens when sso.redhat.com invalidates the token.

Solution

1. Navigate to the OpenShift Cluster Manager API Token page and choose **Load Token**.
2. Copy and paste the following authentication command in the terminal.

```
rosa login --token="<api_token>"
```

Troubleshoot non-STS ROSA cluster issues

This section covers how to troubleshoot issues that you might encounter when provisioning non-STS ROSA clusters.

We recommend that you provision ROSA clusters using AWS Security Token Service (STS) short-lived credentials for better security protection. For more information about provisioning ROSA STS clusters, see the section called "Using ROSA classic with the ROSA CLI in auto mode".

With ROSA, you can also receive troubleshooting support from AWS Support or the Red Hat support teams. For more information, see the section called "Support for ROSA".

Failed to create a cluster with an osdCcsAdmin error

> **Note**
>
> This error occurs only when you use the non-STS method of provisioning ROSA clusters. To avoid this issue, provision your ROSA clusters using AWS STS. For more information, see the section called "Using ROSA classic with the ROSA CLI in auto mode".

Description

If your cluster fails to create, you might receive the following error message:

```
Failed to create cluster: Unable to create cluster spec: Failed to get access keys for
 user 'osdCcsAdmin': NoSuchEntity: The user with name osdCcsAdmin cannot be found.
```

Solution

1. Delete the stack.

   ```
   rosa init --delete-stack
   ```

2. Reinitialize your account.

   ```
   rosa init
   ```

Document history for the ROSA User Guide

The following table describes the important changes to the documentation. For notification about updates to this documentation, you can subscribe to an RSS feed.

Change	Description	Date
ROSA with HCP AWS Region expansion	ROSA with hosted control planes (HCP) is now available in the Middle East (UAE) AWS Region.	May 13, 2024
ROSA with HCP AWS Region expansion	ROSA with hosted control planes (HCP) is now available in the Europe (Paris) AWS Region.	May 6, 2024
Updated ROSANodePoolManagementPolicy	Updated the AWS managed policy ROSANodePoolManagementPolicy.	May 2, 2024
ROSA with HCP AWS Region expansion	ROSA with hosted control planes (HCP) is now available in the Europe (Spain) AWS Region.	April 29, 2024
Updated ROSAInstallerPolicy	Updated the AWS managed policy ROSAInstallerPolicy.	April 24, 2024
ROSA with HCP AWS Region expansion	ROSA with hosted control planes (HCP) is now available in the Europe (Zurich) AWS Region.	April 19, 2024
ROSA with HCP AWS Region expansion	ROSA with hosted control planes (HCP) is now available in the Asia Pacific (Osaka) AWS Region.	April 17, 2024

Updated ROSAInstallerPolicy and ROSASRESupportPolicy	Updated the AWS managed policies ROSAInstallerPolicy and ROSASRESupportPolicy.	April 10, 2024
ROSA with HCP AWS Region expansion	ROSA with hosted control planes (HCP) is now available in the Asia Pacific (Hong Kong) AWS Region.	April 8, 2024
ROSA with HCP AWS Region expansion	ROSA with hosted control planes (HCP) is now available in the South America (São Paulo) AWS Region.	April 1, 2024
ROSA with HCP AWS Region expansion	ROSA with hosted control planes (HCP) is now available in the Middle East (Bahrain) AWS Region.	March 25, 2024
ROSA with HCP AWS Region expansion	ROSA with hosted control planes (HCP) is now available in the Asia Pacific (Seoul) AWS Region.	March 14, 2024
ROSA with HCP AWS Region expansion	ROSA with hosted control planes (HCP) is now available in the Africa (Cape Town) AWS Region.	March 5, 2024
Updated ROSAInstallerPolicy	Updated the AWS managed policy ROSAInstallerPolicy.	January 26, 2024
Updated ROSASRESupportPolicy	Updated the AWS managed policy ROSASRESupportPolicy.	January 22, 2024

Updated ROSAImageRegistryOperatorPolicy	Updated the AWS managed policy ROSAImageRegistryOperatorPolicy.	December 12, 2023
Updated ROSAKubeControllerPolicy	Updated the AWS managed policy ROSAKubeControllerPolicy.	October 16, 2023
Updated ROSAManageSubscription	Updated the AWS managed policy ROSAManageSubscription.	August 1, 2023
Updated ROSAKubeControllerPolicy	Updated the AWS managed policy ROSAKubeControllerPolicy.	July 13, 2023
Added ROSA security pages	Resilience in ROSA, Infrastructure security in ROSA, and Data protection in ROSA pages were added.	June 30, 2023
Added the deployment options page	Deployment options page was added.	June 9, 2023
Added new AWS managed policy ROSANodePoolManagementPolicy	New AWS managed policy ROSANodePoolManagementPolicy was added.	June 8, 2023
Added new AWS managed policy ROSAInstallerPolicy	New AWS managed policy ROSAInstallerPolicy was added.	June 6, 2023
Added new AWS managed policy ROSASRESupportPolicy	New AWS managed policy ROSASRESupportPolicy was added.	June 1, 2023
Added Overview of responsibilities for ROSA	Added Overview of responsibilities for ROSA page.	May 26, 2023

Updated What is Red Hat OpenShift Service on AWS?	Updated the What is Red Hat OpenShift Service on AWS page.	May 24, 2023
Added new AWS managed policies for ROSA operator roles	New AWS managed policies ROSAImageRegistryOperatorPolicy, ROSAKubeControllerPolicy, and ROSAKMSProviderPolicy were added.	April 27, 2023
Added new AWS managed policy ROSAControlPlaneOperatorPolicy	New AWS managed policy ROSAControlPlaneOperatorPolicy was added.	April 24, 2023
Added new AWS managed policies for ROSA account roles	New AWS managed policy pages for ROSA account and operator roles page were added.	April 20, 2023
Added the ROSA service quotas page	The ROSA service quotas page was added.	December 22, 2022
Added troubleshooting pages	Troubleshooting pages were added.	November 1, 2022
Added getting started pages	Getting started pages were added.	August 12, 2022
Added new AWS managed policy ROSAManageSubscription	New AWS managed policy ROSAManageSubscription was added.	April 11, 2022
Initial release	The initial release of the Red Hat OpenShift Service on AWS User Guide.	March 24, 2021