OpenShift Container Platform 4.18

Installing on Azure

Installing OpenShift Container Platform on Azure

Last Updated: 2025-06-06

OpenShift Container Platform 4.18 Installing on Azure

Installing OpenShift Container Platform on Azure

Legal Notice

Abstract

This document describes how to install OpenShift Container Platform on Azure.

Table of Contents

CHAPTER 1. INSTALLATION METHODS 9
1.1. INSTALLING A CLUSTER ON INSTALLER-PROVISIONED INFRASTRUCTURE 9
1.2. INSTALLING A CLUSTER ON USER-PROVISIONED INFRASTRUCTURE 9
1.3. NEXT STEPS 10

CHAPTER 2. CONFIGURING AN AZURE ACCOUNT 11
2.1. AZURE ACCOUNT LIMITS 11
2.2. CONFIGURING A PUBLIC DNS ZONE IN AZURE 14
2.3. RECORDING THE SUBSCRIPTION AND TENANT IDS 14
2.4. SUPPORTED IDENTITIES TO ACCESS AZURE RESOURCES 16
2.4.1. Required Azure roles 17
2.4.1.1. Required Azure permissions for installer-provisioned infrastructure 17
2.4.2. Using Azure managed identities 25
2.4.3. Creating a service principal 25
2.5. SUPPORTED AZURE MARKETPLACE REGIONS 26
2.6. SUPPORTED AZURE REGIONS 27
Supported Azure public regions 27
Supported Azure Government regions 28
2.7. NEXT STEPS 28

CHAPTER 3. INSTALLER-PROVISIONED INFRASTRUCTURE 30
3.1. PREPARING TO INSTALL A CLUSTER ON AZURE 30
3.1.1. Internet access for OpenShift Container Platform 30
3.1.2. Generating a key pair for cluster node SSH access 30
3.1.3. Obtaining the installation program 32
3.1.4. Installing the OpenShift CLI 33
Installing the OpenShift CLI on Linux 33
Installing the OpenShift CLI on Windows 34
Installing the OpenShift CLI on macOS 34
3.1.5. Telemetry access for OpenShift Container Platform 35
3.1.6. Preparing an Azure Disk Encryption Set 35
3.2. INSTALLING A CLUSTER ON AZURE 37
3.2.1. Deploying the cluster 37
3.2.2. Logging in to the cluster by using the CLI 40
3.2.3. Next steps 41
3.3. INSTALLING A CLUSTER ON AZURE WITH CUSTOMIZATIONS 41
3.3.1. Using the Azure Marketplace offering 41
3.3.2. Creating the installation configuration file 43
3.3.2.1. Minimum resource requirements for cluster installation 46
3.3.2.2. Tested instance types for Azure 47
3.3.2.3. Tested instance types for Azure on 64-bit ARM infrastructures 50
3.3.2.4. Enabling trusted launch for Azure VMs 50
3.3.2.5. Enabling confidential VMs 52
3.3.2.6. Sample customized install-config.yaml file for Azure 53
3.3.2.7. Configuring the cluster-wide proxy during installation 56
3.3.3. Configuring user-defined tags for Azure 58
3.3.3.1. Creating user-defined tags for Azure 58
3.3.3.2. User-defined tags requirements 59
3.3.4. Alternatives to storing administrator-level secrets in the kube-system project 60
3.3.4.1. Manually creating long-term credentials 60
3.3.4.2. Configuring an Azure cluster to use short-term credentials 62

3.3.4.2.1. Configuring the Cloud Credential Operator utility 62
3.3.4.2.2. Creating Azure resources with the Cloud Credential Operator utility 65
3.3.4.2.3. Incorporating the Cloud Credential Operator utility manifests 67
3.3.5. Deploying the cluster 68
3.3.6. Logging in to the cluster by using the CLI 69
3.3.7. Next steps 70
3.4. INSTALLING A CLUSTER ON AZURE WITH NETWORK CUSTOMIZATIONS 70
3.4.1. Creating the installation configuration file 70
3.4.1.1. Minimum resource requirements for cluster installation 72
3.4.1.2. Tested instance types for Azure 74
3.4.1.3. Tested instance types for Azure on 64-bit ARM infrastructures 76
3.4.1.4. Enabling trusted launch for Azure VMs 77
3.4.1.5. Enabling confidential VMs 78
3.4.1.6. Sample customized install-config.yaml file for Azure 80
3.4.1.7. Configuring the cluster-wide proxy during installation 83
3.4.2. Network configuration phases 84
3.4.3. Specifying advanced network configuration 85
3.4.4. Cluster Network Operator configuration 86
3.4.4.1. Cluster Network Operator configuration object 87
defaultNetwork object configuration 87
Configuration for the OVN-Kubernetes network plugin 88
3.4.5. Configuring hybrid networking with OVN-Kubernetes 93
3.4.6. Alternatives to storing administrator-level secrets in the kube-system project 95
3.4.6.1. Manually creating long-term credentials 95
3.4.6.2. Configuring an Azure cluster to use short-term credentials 97
3.4.6.2.1. Configuring the Cloud Credential Operator utility 97
3.4.6.2.2. Creating Azure resources with the Cloud Credential Operator utility 99
3.4.6.2.3. Incorporating the Cloud Credential Operator utility manifests 102
3.4.7. Deploying the cluster 103
3.4.8. Logging in to the cluster by using the CLI 104
3.4.9. Next steps 105
3.5. INSTALLING A CLUSTER ON AZURE IN A DISCONNECTED ENVIRONMENT 105
3.5.1. Prerequisites 105
3.5.2. About installations in restricted networks 105
3.5.2.1. Additional limits 106
3.5.2.2. User-defined outbound routing 106
Restricted cluster with Azure Firewall 106
3.5.3. About reusing a VNet for your OpenShift Container Platform cluster 107
3.5.3.1. Requirements for using your VNet 107
3.5.3.1.1. Network security group requirements 108
3.5.3.2. Division of permissions 110
3.5.3.3. Isolation between clusters 110
3.5.4. Creating the installation configuration file 111
3.5.4.1. Minimum resource requirements for cluster installation 114
3.5.4.2. Tested instance types for Azure 115
3.5.4.3. Tested instance types for Azure on 64-bit ARM infrastructures 118
3.5.4.4. Enabling trusted launch for Azure VMs 119
3.5.4.5. Enabling confidential VMs 120
3.5.4.6. Sample customized install-config.yaml file for Azure 121
3.5.4.7. Configuring the cluster-wide proxy during installation 125
3.5.5. Alternatives to storing administrator-level secrets in the kube-system project 127
3.5.5.1. Manually creating long-term credentials 127
3.5.5.2. Configuring an Azure cluster to use short-term credentials 129

3.5.5.2.1. Configuring the Cloud Credential Operator utility 129
3.5.5.2.2. Creating Azure resources with the Cloud Credential Operator utility 131
3.5.5.2.3. Incorporating the Cloud Credential Operator utility manifests 133
3.5.6. Deploying the cluster 135
3.5.7. Logging in to the cluster by using the CLI 136
3.5.8. Next steps 137
3.6. INSTALLING A CLUSTER ON AZURE INTO AN EXISTING VNET 137
3.6.1. About reusing a VNet for your OpenShift Container Platform cluster 137
3.6.1.1. Requirements for using your VNet 137
3.6.1.1.1. Network security group requirements 138
3.6.1.2. Division of permissions 140
3.6.1.3. Isolation between clusters 140
3.6.2. Creating the installation configuration file 140
3.6.2.1. Minimum resource requirements for cluster installation 143
3.6.2.2. Tested instance types for Azure 144
3.6.2.3. Tested instance types for Azure on 64-bit ARM infrastructures 147
3.6.2.4. Enabling trusted launch for Azure VMs 147
3.6.2.5. Enabling confidential VMs 149
3.6.2.6. Sample customized install-config.yaml file for Azure 150
3.6.2.7. Configuring the cluster-wide proxy during installation 153
3.6.3. Alternatives to storing administrator-level secrets in the kube-system project 155
3.6.3.1. Manually creating long-term credentials 155
3.6.3.2. Configuring an Azure cluster to use short-term credentials 157
3.6.3.2.1. Configuring the Cloud Credential Operator utility 157
3.6.3.2.2. Creating Azure resources with the Cloud Credential Operator utility 160
3.6.3.2.3. Incorporating the Cloud Credential Operator utility manifests 162
3.6.4. Deploying the cluster 163
3.6.5. Next steps 164
3.7. INSTALLING A PRIVATE CLUSTER ON AZURE 165
3.7.1. Private clusters 165
3.7.1.1. Private clusters in Azure 165
3.7.1.1.1. Limitations 166
3.7.1.2. User-defined outbound routing 166
Private cluster with network address translation 166
Private cluster with Azure Firewall 167
Private cluster with a proxy configuration 167
Private cluster with no internet access 167
3.7.2. About reusing a VNet for your OpenShift Container Platform cluster 167
3.7.2.1. Requirements for using your VNet 167
3.7.2.1.1. Network security group requirements 169
3.7.2.2. Division of permissions 170
3.7.2.3. Isolation between clusters 171
3.7.3. Manually creating the installation configuration file 171
3.7.3.1. Minimum resource requirements for cluster installation 172
3.7.3.2. Tested instance types for Azure 173
3.7.3.3. Tested instance types for Azure on 64-bit ARM infrastructures 176
3.7.3.4. Enabling trusted launch for Azure VMs 176
3.7.3.5. Enabling confidential VMs 177
3.7.3.6. Sample customized install-config.yaml file for Azure 179
3.7.3.7. Configuring the cluster-wide proxy during installation 182
3.7.4. Alternatives to storing administrator-level secrets in the kube-system project 184
3.7.4.1. Manually creating long-term credentials 184
3.7.4.2. Configuring an Azure cluster to use short-term credentials 186

3.7.4.2.1. Configuring the Cloud Credential Operator utility 186
3.7.4.2.2. Creating Azure resources with the Cloud Credential Operator utility 188
3.7.4.2.3. Incorporating the Cloud Credential Operator utility manifests 191
3.7.5. Optional: Preparing a private Microsoft Azure cluster for a private image registry 192
3.7.6. Deploying the cluster 194
3.7.7. Logging in to the cluster by using the CLI 196
3.7.8. Next steps 196
3.8. INSTALLING A CLUSTER ON AZURE INTO A GOVERNMENT REGION 197
3.8.1. Azure government regions 197
3.8.2. Private clusters 197
3.8.2.1. Private clusters in Azure 198
3.8.2.1.1. Limitations 198
3.8.2.2. User-defined outbound routing 198
3.8.3. About reusing a VNet for your OpenShift Container Platform cluster 199
3.8.3.1. Requirements for using your VNet 199
3.8.3.1.1. Network security group requirements 200
3.8.3.2. Division of permissions 202
3.8.3.3. Isolation between clusters 202
3.8.4. Manually creating the installation configuration file 203
3.8.4.1. Minimum resource requirements for cluster installation 203
3.8.4.2. Tested instance types for Azure 205
3.8.4.3. Enabling trusted launch for Azure VMs 207
3.8.4.4. Enabling confidential VMs 209
3.8.4.5. Sample customized install-config.yaml file for Azure 210
3.8.4.6. Configuring the cluster-wide proxy during installation 213
3.8.5. Deploying the cluster 215
3.8.6. Logging in to the cluster by using the CLI 217
3.8.7. Next steps 218

CHAPTER 4. USER-PROVISIONED INFRASTRUCTURE 219
4.1. PREPARING TO INSTALL A CLUSTER ON AZURE 219
4.1.1. Internet access for OpenShift Container Platform 219
4.1.2. Generating a key pair for cluster node SSH access 219
4.1.3. Obtaining the installation program 221
4.1.4. Installing the OpenShift CLI 222
Installing the OpenShift CLI on Linux 222
Installing the OpenShift CLI on Windows 223
Installing the OpenShift CLI on macOS 223
4.1.5. Telemetry access for OpenShift Container Platform 224
4.2. INSTALLING A CLUSTER ON AZURE IN A DISCONNECTED ENVIRONMENT WITH USER-PROVISIONED INFRASTRUCTURE 224
4.2.1. About installations in restricted networks 225
4.2.1.1. Additional limits 225
4.2.1.2. Internet access for OpenShift Container Platform 226
4.2.2. Configuring your Azure project 226
4.2.2.1. Azure account limits 226
4.2.2.2. Configuring a public DNS zone in Azure 229
4.2.2.3. Certificate signing requests management 229
4.2.2.4. Required Azure roles 230
4.2.2.5. Required Azure permissions for user-provisioned infrastructure 230
4.2.2.6. Creating a service principal 236
4.2.2.7. Supported Azure regions 239
Supported Azure public regions 239

Supported Azure Government regions 240
4.2.3. Requirements for a cluster with user-provisioned infrastructure 241
4.2.3.1. Required machines for cluster installation 241
4.2.3.2. Minimum resource requirements for cluster installation 241
4.2.3.3. Tested instance types for Azure 242
4.2.3.4. Tested instance types for Azure on 64-bit ARM infrastructures 245
4.2.4. Using the Azure Marketplace offering 246
4.2.4.1. Obtaining the installation program 249
4.2.4.2. Generating a key pair for cluster node SSH access 250
4.2.5. Creating the installation files for Azure 251
4.2.5.1. Optional: Creating a separate /var partition 251
4.2.5.2. Creating the installation configuration file 254
4.2.5.3. Configuring the cluster-wide proxy during installation 257
4.2.5.4. Exporting common variables for ARM templates 259
4.2.5.5. Creating the Kubernetes manifest and Ignition config files 260
4.2.6. Creating the Azure resource group 263
4.2.7. Uploading the RHCOS cluster image and bootstrap Ignition config file 264
4.2.8. Example for creating DNS zones 265
4.2.9. Creating a VNet in Azure 266
4.2.9.1. ARM template for the VNet 266
4.2.10. Deploying the RHCOS cluster image for the Azure infrastructure 268
4.2.10.1. ARM template for image storage 269
4.2.11. Networking requirements for user-provisioned infrastructure 272
4.2.11.1. Network connectivity requirements 272
4.2.12. Creating networking and load balancing components in Azure 273
4.2.12.1. ARM template for the network and load balancers 274
4.2.13. Creating the bootstrap machine in Azure 279
4.2.13.1. ARM template for the bootstrap machine 280
4.2.14. Creating the control plane machines in Azure 284
4.2.14.1. ARM template for control plane machines 285
4.2.15. Wait for bootstrap completion and remove bootstrap resources in Azure 289
4.2.16. Creating additional worker machines in Azure 290
4.2.16.1. ARM template for worker machines 290
4.2.17. Installing the OpenShift CLI 294
Installing the OpenShift CLI on Linux 294
Installing the OpenShift CLI on Windows 295
Installing the OpenShift CLI on macOS 295
4.2.18. Logging in to the cluster by using the CLI 296
4.2.19. Approving the certificate signing requests for your machines 296
4.2.20. Adding the Ingress DNS records 299
4.2.21. Completing an Azure installation on user-provisioned infrastructure 300
4.2.22. Telemetry access for OpenShift Container Platform 301
4.3. INSTALLING A CLUSTER ON AZURE USING ARM TEMPLATES 301
4.3.1. Prerequisites 302
4.3.2. Internet access for OpenShift Container Platform 302
4.3.3. Configuring your Azure project 303
4.3.3.1. Azure account limits 303
4.3.3.2. Configuring a public DNS zone in Azure 306
4.3.3.3. Certificate signing requests management 306
4.3.3.4. Recording the subscription and tenant IDs 307
4.3.3.5. Supported identities to access Azure resources 308
4.3.3.6. Required Azure permissions for user-provisioned infrastructure 309
4.3.3.7. Using Azure managed identities 315

4.3.3.8. Creating a service principal 315
4.3.3.9. Supported Azure regions 316
Supported Azure public regions 316
Supported Azure Government regions 318
4.3.4. Requirements for a cluster with user-provisioned infrastructure 318
4.3.4.1. Required machines for cluster installation 318
4.3.4.2. Minimum resource requirements for cluster installation 319
4.3.4.3. Tested instance types for Azure 320
4.3.4.4. Tested instance types for Azure on 64-bit ARM infrastructures 323
4.3.5. Using the Azure Marketplace offering 324
4.3.6. Obtaining the installation program 326
4.3.7. Generating a key pair for cluster node SSH access 327
4.3.8. Creating the installation files for Azure 329
4.3.8.1. Optional: Creating a separate /var partition 329
4.3.8.2. Creating the installation configuration file 332
4.3.8.3. Configuring the cluster-wide proxy during installation 334
4.3.8.4. Exporting common variables for ARM templates 335
4.3.8.5. Creating the Kubernetes manifest and Ignition config files 337
4.3.9. Creating the Azure resource group 340
4.3.10. Uploading the RHCOS cluster image and bootstrap Ignition config file 340
4.3.11. Example for creating DNS zones 342
4.3.12. Creating a VNet in Azure 342
4.3.12.1. ARM template for the VNet 343
4.3.13. Deploying the RHCOS cluster image for the Azure infrastructure 345
4.3.13.1. ARM template for image storage 346
4.3.14. Networking requirements for user-provisioned infrastructure 349
4.3.14.1. Network connectivity requirements 349
4.3.15. Creating networking and load balancing components in Azure 350
4.3.15.1. ARM template for the network and load balancers 351
4.3.16. Creating the bootstrap machine in Azure 356
4.3.16.1. ARM template for the bootstrap machine 357
4.3.17. Creating the control plane machines in Azure 361
4.3.17.1. ARM template for control plane machines 362
4.3.18. Wait for bootstrap completion and remove bootstrap resources in Azure 366
4.3.19. Creating additional worker machines in Azure 366
4.3.19.1. ARM template for worker machines 367
4.3.20. Installing the OpenShift CLI 371
Installing the OpenShift CLI on Linux 371
Installing the OpenShift CLI on Windows 372
Installing the OpenShift CLI on macOS 372
4.3.21. Logging in to the cluster by using the CLI 373
4.3.22. Approving the certificate signing requests for your machines 373
4.3.23. Adding the Ingress DNS records 376
4.3.24. Completing an Azure installation on user-provisioned infrastructure 377
4.3.25. Telemetry access for OpenShift Container Platform 378

CHAPTER 5. INSTALLING A THREE-NODE CLUSTER ON AZURE **379**
5.1. CONFIGURING A THREE-NODE CLUSTER 379
5.2. NEXT STEPS 380

CHAPTER 6. UNINSTALLING A CLUSTER ON AZURE **381**
6.1. REMOVING A CLUSTER THAT USES INSTALLER-PROVISIONED INFRASTRUCTURE 381
6.2. DELETING MICROSOFT AZURE RESOURCES WITH THE CLOUD CREDENTIAL OPERATOR UTILITY 381

CHAPTER 7. INSTALLATION CONFIGURATION PARAMETERS FOR AZURE 383
7.1. AVAILABLE INSTALLATION CONFIGURATION PARAMETERS FOR AZURE 383
7.1.1. Required configuration parameters 383
7.1.2. Network configuration parameters 384
7.1.3. Optional configuration parameters 386
7.1.4. Additional Azure configuration parameters 393

CHAPTER 1. INSTALLATION METHODS

You can install OpenShift Container Platform on installer-provisioned or user-provisioned infrastructure. The default installation type uses installer-provisioned infrastructure, where the installation program provisions the underlying infrastructure for the cluster. You can also install OpenShift Container Platform on infrastructure that you provision. If you do not use infrastructure that the installation program provisions, you must manage and maintain the cluster resources yourself.

See Installation process for more information about installer-provisioned and user-provisioned installation processes.

1.1. INSTALLING A CLUSTER ON INSTALLER-PROVISIONED INFRASTRUCTURE

You can install a cluster on Microsoft Azure infrastructure that is provisioned by the OpenShift Container Platform installation program, by using one of the following methods:

- **Installing a cluster quickly on Azure**: You can install OpenShift Container Platform on Azure infrastructure that is provisioned by the OpenShift Container Platform installation program. You can install a cluster quickly by using the default configuration options.

- **Installing a customized cluster on Azure**: You can install a customized cluster on Azure infrastructure that the installation program provisions. The installation program allows for some customization to be applied at the installation stage. Many other customization options are available post-installation.

- **Installing a cluster on Azure with network customizations**: You can customize your OpenShift Container Platform network configuration during installation, so that your cluster can coexist with your existing IP address allocations and adhere to your network requirements.

- **Installing a cluster on Azure in a restricted network**: You can install a cluster on Azure in a restricted network by creating an internal mirror of the installation release content on an existing Azure Virtual Network (VNet).

- **Installing a cluster on Azure into an existing VNet**: You can install OpenShift Container Platform on an existing Azure Virtual Network (VNet) on Azure. You can use this installation method if you have constraints set by the guidelines of your company, such as limits when creating new accounts or infrastructure.

- **Installing a private cluster on Azure**: You can install a private cluster into an existing Azure Virtual Network (VNet) on Azure. You can use this method to deploy OpenShift Container Platform on an internal network that is not visible to the internet.

- **Installing a cluster on Azure into a government region**: OpenShift Container Platform can be deployed into Microsoft Azure Government (MAG) regions that are specifically designed for US government agencies at the federal, state, and local level, as well as contractors, educational institutions, and other US customers that must run sensitive workloads on Azure.

1.2. INSTALLING A CLUSTER ON USER-PROVISIONED INFRASTRUCTURE

You can install a cluster on Azure infrastructure that you provision, by using one of the following methods:

- **Installing a cluster on Azure in a restricted network with user-provisioned infrastructure** You can perform an installation on Azure that does not require an active connection to the internet to obtain software components.

- **Installing a cluster on Azure using ARM templates** You can install OpenShift Container Platform on Azure by using infrastructure that you provide. You can use the provided Azure Resource Manager (ARM) templates to assist with an installation.

1.3. NEXT STEPS

- Configuring an Azure account

CHAPTER 2. CONFIGURING AN AZURE ACCOUNT

Before you can install OpenShift Container Platform, you must configure a Microsoft Azure account to meet installation requirements.

IMPORTANT

All Azure resources that are available through public endpoints are subject to resource name restrictions. For a list of terms that Azure restricts for resource names, see Resolve reserved resource name errors in the Azure documentation.

2.1. AZURE ACCOUNT LIMITS

The OpenShift Container Platform cluster uses a number of Microsoft Azure components, and the default Azure subscription and service limits, quotas, and constraints affect your ability to install OpenShift Container Platform clusters.

IMPORTANT

Default limits vary by offer category types, such as Free Trial and Pay-As-You-Go, and by series, such as Dv2, F, and G. For example, the default for Enterprise Agreement subscriptions is 350 cores.

Check the limits for your subscription type and if necessary, increase quota limits for your account before you install a default cluster on Azure.

The following table summarizes the Azure components whose limits can impact your ability to install and run OpenShift Container Platform clusters.

Component	Number of components required by default	Default Azure limit	Description

Component	Number of components required by default	Default Azure limit	Description
vCPU	44	20 per region	A default cluster requires 44 vCPUs, so you must increase the account limit. By default, each cluster creates the following instances: ● One bootstrap machine, which is removed after installation ● Three control plane machines ● Three compute machines Because the bootstrap and control plane machines use **Standard_D8s_v3** virtual machines, which use 8 vCPUs, and the compute machines use **Standard_D4s_v3** virtual machines, which use 4 vCPUs, a default cluster requires 44 vCPUs. The bootstrap node VM, which uses 8 vCPUs, is used only during installation. To deploy more worker nodes, enable autoscaling, deploy large workloads, or use a different instance type, you must further increase the vCPU limit for your account to ensure that your cluster can deploy the machines that you require.
OS Disk	7		Each cluster machine must have a minimum of 100 GB of storage and 300 IOPS. NOTE Faster storage is recommended for production clusters and clusters with intensive workloads. For more information about optimizing storage for performance, see the page titled "Optimizing storage" in the "Scalability and performance" section.
VNet	1	1000 per region	Each default cluster requires one Virtual Network (VNet), which contains two subnets.
Network interfaces	7	65,536 per region	Each default cluster requires seven network interfaces. If you create more machines or your deployed workloads create load balancers, your cluster uses more network interfaces.

<table>
<tr><th>Component</th><th>Number of components required by default</th><th>Default Azure limit</th><th>Description</th></tr>
<tr><td>Network security groups</td><td>2</td><td>5000</td><td>Each cluster creates network security groups for each subnet in the VNet. The default cluster creates network security groups for the control plane and for the compute node subnets:
<table>
<tr><td><b>controlplane</b></td><td>Allows the control plane machines to be reached on port 6443 from anywhere</td></tr>
<tr><td><b>node</b></td><td>Allows worker nodes to be reached from the internet on ports 80 and 443</td></tr>
</table></td></tr>
<tr><td>Network load balancers</td><td>3</td><td>1000 per region</td><td>Each cluster creates the following load balancers:
<table>
<tr><td><b>default</b></td><td>Public IP address that load balances requests to ports 80 and 443 across worker machines</td></tr>
<tr><td><b>internal</b></td><td>Private IP address that load balances requests to ports 6443 and 22623 across control plane machines</td></tr>
<tr><td><b>external</b></td><td>Public IP address that load balances requests to port 6443 across control plane machines</td></tr>
</table>
If your applications create more Kubernetes <b>LoadBalancer</b> service objects, your cluster uses more load balancers.</td></tr>
<tr><td>Public IP addresses</td><td>3</td><td></td><td>Each of the two public load balancers uses a public IP address. The bootstrap machine also uses a public IP address so that you can SSH into the machine to troubleshoot issues during installation. The IP address for the bootstrap node is used only during installation.</td></tr>
<tr><td>Private IP addresses</td><td>7</td><td></td><td>The internal load balancer, each of the three control plane machines, and each of the three worker machines each use a private IP address.</td></tr>
</table>

Component	Number of components required by default	Default Azure limit	Description
Spot VM vCPUs (optional)	0 If you configure spot VMs, your cluster must have two spot VM vCPUs for every compute node.	20 per region	This is an optional component. To use spot VMs, you must increase the Azure default limit to at least twice the number of compute nodes in your cluster. **NOTE** Using spot VMs for control plane nodes is not recommended.

To increase an account limit, file a support request on the Azure portal. For more information, see Request a quota limit increase for Azure Deployment Environments resources .

Additional resources

- Optimizing storage.

2.2. CONFIGURING A PUBLIC DNS ZONE IN AZURE

To install OpenShift Container Platform, the Microsoft Azure account you use must have a dedicated public hosted DNS zone in your account. This zone must be authoritative for the domain. This service provides cluster DNS resolution and name lookup for external connections to the cluster.

Procedure

1. Identify your domain, or subdomain, and registrar. You can transfer an existing domain and registrar or obtain a new one through Azure or another source.
   - To purchase a new domain through Azure, see Buy a custom domain name for Azure App Service.
   - If you are using an existing domain and registrar, migrate its DNS to Azure. For more information, see Migrate an active DNS name to Azure App Service in the Azure documentation.
2. Configure DNS for your domain, which includes creating a public hosted zone for your domain or subdomain, extracting the new authoritative name servers, and updating the registrar records for the name servers that your domain uses. For more information, see Tutorial: Host your domain in Azure DNS.
   Use an appropriate root domain, such as **openshiftcorp.com**, or subdomain, such as **clusters.openshiftcorp.com**.
3. If you use a subdomain, follow your organization’s procedures to add its delegation records to the parent domain.

2.3. RECORDING THE SUBSCRIPTION AND TENANT IDS

The installation program requires the subscription and tenant IDs that are associated with your Azure account. You can use the Azure CLI to gather this information.

Prerequisites

- You have installed or updated the Azure CLI.

Procedure

1. Log in to the Azure CLI by running the following command:

```
$ az login
```

2. Ensure that you are using the right subscription:

   a. View a list of available subscriptions by running the following command:

```
$ az account list --refresh
```

Example output

```
[
  {
    "cloudName": "AzureCloud",
    "id": "8xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "isDefault": true,
    "name": "Subscription Name 1",
    "state": "Enabled",
    "tenantId": "6xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "user": {
      "name": "you@example.com",
      "type": "user"
    }
  },
  {
    "cloudName": "AzureCloud",
    "id": "9xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "isDefault": false,
    "name": "Subscription Name 2",
    "state": "Enabled",
    "tenantId": "7xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "user": {
      "name": "you2@example.com",
      "type": "user"
    }
  }
]
```

   b. View the details of the active account, and confirm that this is the subscription you want to use, by running the following command:

```
$ az account show
```

Example output

```
{
  "environmentName": "AzureCloud",
  "id": "8xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
  "isDefault": true,
  "name": "Subscription Name 1",
  "state": "Enabled",
  "tenantId": "6xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

3. If you are not using the right subscription:

   a. Change the active subscription by running the following command:

   ```
   $ az account set -s <subscription_id>
   ```

   b. Verify that you are using the subscription you need by running the following command:

   ```
   $ az account show
   ```

   Example output

   ```
   {
     "environmentName": "AzureCloud",
     "id": "9xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
     "isDefault": true,
     "name": "Subscription Name 2",
     "state": "Enabled",
     "tenantId": "7xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
     "user": {
       "name": "you2@example.com",
       "type": "user"
     }
   }
   ```

4. Record the **id** and **tenantId** parameter values from the output. You require these values to install an OpenShift Container Platform cluster.

2.4. SUPPORTED IDENTITIES TO ACCESS AZURE RESOURCES

An OpenShift Container Platform cluster requires an Azure identity to create and manage Azure resources. You need one of the following types of identities to complete the installation:

- A service principal
- A system-assigned managed identity
- A user-assigned managed identity

For more information on Azure identities, see Managed identity types.

2.4.1. Required Azure roles

Before you create the identity, verify that your environment meets the following requirements based on the identity:

- The Azure account that you use to create the identity is assigned the **User Access Administrator** and **Contributor** roles. These roles are required when:
    - Creating a service principal or user-assigned managed identity.
    - Enabling a system-assigned managed identity on a virtual machine.
- If you are going to use a service principal to complete the installation, verify that the Azure account that you use to create the identity is assigned the **microsoft.directory/servicePrincipals/createAsOwner** permission in Microsoft Entra ID.

To set roles on the Azure portal, see Assign Azure roles using the Azure portal in the Azure documentation.

2.4.1.1. Required Azure permissions for installer-provisioned infrastructure

The installation program requires access to an Azure service principal or managed identity with the necessary permissions to deploy the cluster and to maintain its daily operation. These permissions must be granted to the Azure subscription that is associated with the identity.

The following options are available to you:

- You can assign the identity the **Contributor** and **User Access Administrator** roles, which grant all of the required permissions.
  For more information about assigning roles, see the Azure documentation for managing access to Azure resources using the Azure portal.
- If the security policies of your organization require a more restrictive set of permissions, you can create a custom role with the necessary permissions.

The following permissions are required for creating an OpenShift Container Platform cluster on Microsoft Azure.

> Example 2.1. Required permissions for creating authorization resources
>
> - **Microsoft.Authorization/policies/audit/action**
> - **Microsoft.Authorization/policies/auditIfNotExists/action**
> - **Microsoft.Authorization/roleAssignments/read**
> - **Microsoft.Authorization/roleAssignments/write**

> Example 2.2. Required permissions for creating compute resources
>
> - **Microsoft.Compute/availabilitySets/read**
> - **Microsoft.Compute/availabilitySets/write**
> - **Microsoft.Compute/disks/beginGetAccess/action**

- **Microsoft.Compute/disks/delete**
- **Microsoft.Compute/disks/read**
- **Microsoft.Compute/disks/write**
- **Microsoft.Compute/galleries/images/read**
- **Microsoft.Compute/galleries/images/versions/read**
- **Microsoft.Compute/galleries/images/versions/write**
- **Microsoft.Compute/galleries/images/write**
- **Microsoft.Compute/galleries/read**
- **Microsoft.Compute/galleries/write**
- **Microsoft.Compute/snapshots/read**
- **Microsoft.Compute/snapshots/write**
- **Microsoft.Compute/snapshots/delete**
- **Microsoft.Compute/virtualMachines/delete**
- **Microsoft.Compute/virtualMachines/powerOff/action**
- **Microsoft.Compute/virtualMachines/read**
- **Microsoft.Compute/virtualMachines/write**

Example 2.3. Required permissions for creating identity management resources

- **Microsoft.ManagedIdentity/userAssignedIdentities/assign/action**
- **Microsoft.ManagedIdentity/userAssignedIdentities/read**
- **Microsoft.ManagedIdentity/userAssignedIdentities/write**

Example 2.4. Required permissions for creating network resources

- **Microsoft.Network/dnsZones/A/write**
- **Microsoft.Network/dnsZones/CNAME/write**
- **Microsoft.Network/dnszones/CNAME/read**
- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/loadBalancers/backendAddressPools/join/action**
- **Microsoft.Network/loadBalancers/backendAddressPools/read**
- **Microsoft.Network/loadBalancers/backendAddressPools/write**

- **Microsoft.Network/loadBalancers/read**
- **Microsoft.Network/loadBalancers/write**
- **Microsoft.Network/loadBalancers/inboundNatRules/read**
- **Microsoft.Network/loadBalancers/inboundNatRules/write**
- **Microsoft.Network/loadBalancers/inboundNatRules/join/action**
- **Microsoft.Network/loadBalancers/inboundNatRules/delete**
- **Microsoft.Network/routeTables/read**
- **Microsoft.Network/routeTables/write**
- **Microsoft.Network/routeTables/join/action**
- **Microsoft.Network/networkInterfaces/delete**
- **Microsoft.Network/networkInterfaces/join/action**
- **Microsoft.Network/networkInterfaces/read**
- **Microsoft.Network/networkInterfaces/write**
- **Microsoft.Network/networkSecurityGroups/join/action**
- **Microsoft.Network/networkSecurityGroups/read**
- **Microsoft.Network/networkSecurityGroups/securityRules/delete**
- **Microsoft.Network/networkSecurityGroups/securityRules/read**
- **Microsoft.Network/networkSecurityGroups/securityRules/write**
- **Microsoft.Network/networkSecurityGroups/write**
- **Microsoft.Network/privateDnsZones/A/read**
- **Microsoft.Network/privateDnsZones/A/write**
- **Microsoft.Network/privateDnsZones/A/delete**
- **Microsoft.Network/privateDnsZones/SOA/read**
- **Microsoft.Network/privateDnsZones/read**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/read**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/write**
- **Microsoft.Network/privateDnsZones/write**
- **Microsoft.Network/publicIPAddresses/delete**
- **Microsoft.Network/publicIPAddresses/join/action**

- **Microsoft.Network/publicIPAddresses/read**
- **Microsoft.Network/publicIPAddresses/write**
- **Microsoft.Network/virtualNetworks/join/action**
- **Microsoft.Network/virtualNetworks/read**
- **Microsoft.Network/virtualNetworks/subnets/join/action**
- **Microsoft.Network/virtualNetworks/subnets/read**
- **Microsoft.Network/virtualNetworks/subnets/write**
- **Microsoft.Network/virtualNetworks/write**

NOTE

The following permissions are not required to create the private OpenShift Container Platform cluster on Azure.

- **Microsoft.Network/dnsZones/A/write**
- **Microsoft.Network/dnsZones/CNAME/write**
- **Microsoft.Network/dnszones/CNAME/read**
- **Microsoft.Network/dnszones/read**

Example 2.5. Required permissions for checking the health of resources

- **Microsoft.Resourcehealth/healthevent/Activated/action**
- **Microsoft.Resourcehealth/healthevent/InProgress/action**
- **Microsoft.Resourcehealth/healthevent/Pending/action**
- **Microsoft.Resourcehealth/healthevent/Resolved/action**
- **Microsoft.Resourcehealth/healthevent/Updated/action**

Example 2.6. Required permissions for creating a resource group

- **Microsoft.Resources/subscriptions/resourceGroups/read**
- **Microsoft.Resources/subscriptions/resourcegroups/write**

Example 2.7. Required permissions for creating resource tags

- **Microsoft.Resources/tags/write**

Example 2.8. Required permissions for creating storage resources

- **Microsoft.Storage/storageAccounts/blobServices/read**
- **Microsoft.Storage/storageAccounts/blobServices/containers/write**
- **Microsoft.Storage/storageAccounts/fileServices/read**
- **Microsoft.Storage/storageAccounts/fileServices/shares/read**
- **Microsoft.Storage/storageAccounts/fileServices/shares/write**
- **Microsoft.Storage/storageAccounts/fileServices/shares/delete**
- **Microsoft.Storage/storageAccounts/listKeys/action**
- **Microsoft.Storage/storageAccounts/read**
- **Microsoft.Storage/storageAccounts/write**

Example 2.9. Optional permissions for creating a private storage endpoint for the image registry

- **Microsoft.Network/privateEndpoints/write**
- **Microsoft.Network/privateEndpoints/read**
- **Microsoft.Network/privateEndpoints/privateDnsZoneGroups/write**
- **Microsoft.Network/privateEndpoints/privateDnsZoneGroups/read**
- **Microsoft.Network/privateDnsZones/join/action**
- **Microsoft.Storage/storageAccounts/PrivateEndpointConnectionsApproval/action**

Example 2.10. Optional permissions for creating marketplace virtual machine resources

- **Microsoft.MarketplaceOrdering/offertypes/publishers/offers/plans/agreements/read**
- **Microsoft.MarketplaceOrdering/offertypes/publishers/offers/plans/agreements/write**

Example 2.11. Optional permissions for creating compute resources

- **Microsoft.Compute/availabilitySets/delete**
- **Microsoft.Compute/images/read**
- **Microsoft.Compute/images/write**
- **Microsoft.Compute/images/delete**

Example 2.12. Optional permissions for enabling user-managed encryption

- **Microsoft.Compute/diskEncryptionSets/read**
- **Microsoft.Compute/diskEncryptionSets/write**
- **Microsoft.Compute/diskEncryptionSets/delete**
- **Microsoft.KeyVault/vaults/read**
- **Microsoft.KeyVault/vaults/write**
- **Microsoft.KeyVault/vaults/delete**
- **Microsoft.KeyVault/vaults/deploy/action**
- **Microsoft.KeyVault/vaults/keys/read**
- **Microsoft.KeyVault/vaults/keys/write**
- **Microsoft.Features/providers/features/register/action**

Example 2.13. Optional permissions for installing a cluster using the **NatGateway** outbound type

- **Microsoft.Network/natGateways/read**
- **Microsoft.Network/natGateways/write**

Example 2.14. Optional permissions for installing a private cluster with Azure Network Address Translation (NAT)

- **Microsoft.Network/natGateways/join/action**
- **Microsoft.Network/natGateways/read**
- **Microsoft.Network/natGateways/write**

Example 2.15. Optional permissions for installing a private cluster with Azure firewall

- **Microsoft.Network/azureFirewalls/applicationRuleCollections/write**
- **Microsoft.Network/azureFirewalls/read**
- **Microsoft.Network/azureFirewalls/write**
- **Microsoft.Network/routeTables/join/action**
- **Microsoft.Network/routeTables/read**
- **Microsoft.Network/routeTables/routes/read**
- **Microsoft.Network/routeTables/routes/write**
- **Microsoft.Network/routeTables/write**

- **Microsoft.Network/virtualNetworks/peer/action**
- **Microsoft.Network/virtualNetworks/virtualNetworkPeerings/read**
- **Microsoft.Network/virtualNetworks/virtualNetworkPeerings/write**

Example 2.16. Optional permission for running gather bootstrap

- **Microsoft.Compute/virtualMachines/retrieveBootDiagnosticsData/action**

The following permissions are required for deleting an OpenShift Container Platform cluster on Microsoft Azure. You can use the same permissions to delete a private OpenShift Container Platform cluster on Azure.

Example 2.17. Required permissions for deleting authorization resources

- **Microsoft.Authorization/roleAssignments/delete**

Example 2.18. Required permissions for deleting compute resources

- **Microsoft.Compute/disks/delete**
- **Microsoft.Compute/galleries/delete**
- **Microsoft.Compute/galleries/images/delete**
- **Microsoft.Compute/galleries/images/versions/delete**
- **Microsoft.Compute/virtualMachines/delete**

Example 2.19. Required permissions for deleting identity management resources

- **Microsoft.ManagedIdentity/userAssignedIdentities/delete**

Example 2.20. Required permissions for deleting network resources

- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/dnsZones/A/read**
- **Microsoft.Network/dnsZones/A/delete**
- **Microsoft.Network/dnsZones/CNAME/read**
- **Microsoft.Network/dnsZones/CNAME/delete**
- **Microsoft.Network/loadBalancers/delete**
- **Microsoft.Network/networkInterfaces/delete**

- **Microsoft.Network/networkSecurityGroups/delete**
- **Microsoft.Network/privateDnsZones/read**
- **Microsoft.Network/privateDnsZones/A/read**
- **Microsoft.Network/privateDnsZones/delete**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/delete**
- **Microsoft.Network/publicIPAddresses/delete**
- **Microsoft.Network/virtualNetworks/delete**

NOTE

The following permissions are not required to delete a private OpenShift Container Platform cluster on Azure.

- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/dnsZones/A/read**
- **Microsoft.Network/dnsZones/A/delete**
- **Microsoft.Network/dnsZones/CNAME/read**
- **Microsoft.Network/dnsZones/CNAME/delete**

Example 2.21. Required permissions for checking the health of resources

- **Microsoft.Resourcehealth/healthevent/Activated/action**
- **Microsoft.Resourcehealth/healthevent/Resolved/action**
- **Microsoft.Resourcehealth/healthevent/Updated/action**

Example 2.22. Required permissions for deleting a resource group

- **Microsoft.Resources/subscriptions/resourcegroups/delete**

Example 2.23. Required permissions for deleting storage resources

- **Microsoft.Storage/storageAccounts/delete**
- **Microsoft.Storage/storageAccounts/listKeys/action**

NOTE

To install OpenShift Container Platform on Azure, you must scope the permissions to your subscription. You can re-scope these permissions to the resource group created by installation program. If the public DNS zone is present in a different resource group, then the network DNS zone related permissions must always be applied to your subscription. By default, the OpenShift Container Platform installation program assigns the Azure identity the **Contributor** role.

You can scope all the permissions to your subscription when deleting an OpenShift Container Platform cluster.

2.4.2. Using Azure managed identities

The installation program requires an Azure identity to complete the installation. You can use either a system-assigned or user-assigned managed identity.

If you are unable to use a managed identity, you can use a service principal.

Procedure

1. If you are using a system-assigned managed identity, enable it on the virtual machine that you will run the installation program from.
2. If you are using a user-assigned managed identity:
   a. Assign it to the virtual machine that you will run the installation program from.
   b. Record its client ID. You require this value when installing the cluster. For more information about viewing the details of a user-assigned managed identity, see List user-assigned managed identities in the Azure documentation.
3. Verify that the required permissions are assigned to the managed identity.

2.4.3. Creating a service principal

The installation program requires an Azure identity to complete the installation. You can use a service principal.

If you are unable to use a service principal, you can use a managed identity.

Prerequisites

- You have installed or updated the Azure CLI.
- You have an Azure subscription ID.
- If you are not assigning the **Contributor** and **User Administrator Access** roles to the service principal, you have created a custom role with the required Azure permissions.

Procedure

1. Create the service principal for your account by running the following command:

```
$ az ad sp create-for-rbac --role <role_name> \ 1
    --name <service_principal> \ 2
    --scopes /subscriptions/<subscription_id> 3
```

1 Defines the role name. You can use the **Contributor** role, or you can specify a custom role which contains the necessary permissions.

2 Defines the service principal name.

3 Specifies the subscription ID.

Example output

```
Creating 'Contributor' role assignment under scope '/subscriptions/<subscription_id>'
The output includes credentials that you must protect. Be sure that you do not
include these credentials in your code or check the credentials into your source
control. For more information, see https://aka.ms/azadsp-cli
{
  "appId": "axxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
  "displayName": <service_principal>",
  "password": "00000000-0000-0000-0000-000000000000",
  "tenantId": "8xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
}
```

Record the values of the **appId** and **password** parameters from the output. You require these values when installing the cluster.

2. If you assigned the **Contributor** role to your service principal, assign the **User Administrator Access** role by running the following command:

```
$ az role assignment create --role "User Access Administrator" \
  --assignee-object-id $(az ad sp show --id <appId> --query id -o tsv) 1
  --scope /subscriptions/<subscription_id> 2
```

1 Specify the **appId** parameter value for your service principal.

2 Specifies the subscription ID.

Additional resources

- About the Cloud Credential Operator

2.5. SUPPORTED AZURE MARKETPLACE REGIONS

Installing a cluster using the Azure Marketplace image is available to customers who purchase the offer in North America and EMEA.

While the offer must be purchased in North America or EMEA, you can deploy the cluster to any of the Azure public partitions that OpenShift Container Platform supports.

NOTE

Deploying a cluster using the Azure Marketplace image is not supported for the Azure Government regions.

2.6. SUPPORTED AZURE REGIONS

The installation program dynamically generates the list of available Microsoft Azure regions based on your subscription.

Supported Azure public regions

- **australiacentral** (Australia Central)
- **australiaeast** (Australia East)
- **australiasoutheast** (Australia South East)
- **brazilsouth** (Brazil South)
- **canadacentral** (Canada Central)
- **canadaeast** (Canada East)
- **centralindia** (Central India)
- **centralus** (Central US)
- **chilecentral** (Chile Central)
- **eastasia** (East Asia)
- **eastus** (East US)
- **eastus2** (East US 2)
- **francecentral** (France Central)
- **germanywestcentral** (Germany West Central)
- **indonesiacentral** (Indonesia Central)
- **israelcentral** (Israel Central)
- **italynorth** (Italy North)
- **japaneast** (Japan East)
- **japanwest** (Japan West)
- **koreacentral** (Korea Central)
- **koreasouth** (Korea South)
- **malaysiawest** (Malaysia West)
- **mexicocentral** (Mexico Central)

- **newzealandnorth** (New Zealand North)
- **northcentralus** (North Central US)
- **northeurope** (North Europe)
- **norwayeast** (Norway East)
- **polandcentral** (Poland Central)
- **qatarcentral** (Qatar Central)
- **southafricanorth** (South Africa North)
- **southcentralus** (South Central US)
- **southeastasia** (Southeast Asia)
- **southindia** (South India)
- **spaincentral** (Spain Central)
- **swedencentral** (Sweden Central)
- **switzerlandnorth** (Switzerland North)
- **uaenorth** (UAE North)
- **uksouth** (UK South)
- **ukwest** (UK West)
- **westcentralus** (West Central US)
- **westeurope** (West Europe)
- **westindia** (West India)
- **westus** (West US)
- **westus2** (West US 2)
- **westus3** (West US 3)

Supported Azure Government regions

Support for the following Microsoft Azure Government (MAG) regions was added in OpenShift Container Platform version 4.6:

- **usgovtexas** (US Gov Texas)
- **usgovvirginia** (US Gov Virginia)

You can reference all available MAG regions in the Azure documentation. Other provided MAG regions are expected to work with OpenShift Container Platform, but have not been tested.

2.7. NEXT STEPS

- Install an OpenShift Container Platform cluster on Azure. You can install a customized cluster or quickly install a cluster with default options.

CHAPTER 3. INSTALLER-PROVISIONED INFRASTRUCTURE

3.1. PREPARING TO INSTALL A CLUSTER ON AZURE

To prepare for installation of an OpenShift Container Platform cluster on Azure, complete the following steps:

- You have selected a cluster installation method.
- You configured an Azure account to host the cluster and determined the tested and validated region to deploy the cluster to.
- If you use a firewall, you have configured it to allow the sites that your cluster requires access to.

3.1.1. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.18, you require access to the internet to install your cluster.

You must have internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

3.1.2. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the **~/.ssh/authorized_keys** list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The **./openshift-install gather** command also requires the SSH public key to be in place on the cluster nodes.

IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

NOTE

You must use a local key, not one that you configured with platform-specific approaches.

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
   ```

   1 Specify the path and file name, such as **~/.ssh/id_ed25519**, of the new SSH key. If you have an existing key pair, ensure your public key is in the your **~/.ssh** directory.

   NOTE

   If you plan to install an OpenShift Container Platform cluster that uses the RHEL cryptographic libraries that have been submitted to NIST for FIPS 140-2/140-3 Validation on only the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

   ```
   $ cat <path>/<file_name>.pub
   ```

   For example, run the following to view the **~/.ssh/id_ed25519.pub** public key:

   ```
   $ cat ~/.ssh/id_ed25519.pub
   ```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the **./openshift-install gather** command.

   NOTE

   On some distributions, default SSH private key identities such as **~/.ssh/id_rsa** and **~/.ssh/id_dsa** are managed automatically.

   a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

      ```
      $ eval "$(ssh-agent -s)"
      ```

      Example output

```
Agent pid 31874
```

NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

   ```
   $ ssh-add <path>/<file_name> 1
   ```

   1 Specify the path and file name for your SSH private key, such as **~/.ssh/id_ed25519**

   Example output

   ```
   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
   ```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

3.1.3. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Go to the Cluster Type page on the Red Hat Hybrid Cloud Console. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

   TIP

   You can also download the binaries for a specific OpenShift Container Platform release .

2. Select your infrastructure provider from the **Run it yourself** section of the page.

3. Select your host operating system and architecture from the dropdown menus under **OpenShift Installer** and click **Download Installer**.

4. Place the downloaded file in the directory where you want to store the installation configuration files.

IMPORTANT

- The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both of the files are required to delete the cluster.
- Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

5. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

6. Download your installation pull secret from Red Hat OpenShift Cluster Manager . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

TIP

Alternatively, you can retrieve the installation program from the Red Hat Customer Portal, where you can specify a version of the installation program to download. However, you must have an active subscription to access this page.

3.1.4. Installing the OpenShift CLI

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.

IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.18. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux
You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the OpenShift Container Platform downloads page on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.18 Linux Clients** entry and save the file.
5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
   To check your **PATH**, execute the following command:

   ```
   $ echo $PATH
   ```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

  ```
  $ oc <command>
  ```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the OpenShift Container Platform downloads page on the Red Hat Customer Portal.

2. Select the appropriate version from the **Version** drop-down list.

3. Click **Download Now** next to the **OpenShift v4.18 Windows Client** entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the **oc** binary to a directory that is on your **PATH**.
   To check your **PATH**, open the command prompt and execute the following command:

   ```
   C:\> path
   ```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

  ```
  C:\> oc <command>
  ```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the OpenShift Container Platform downloads page on the Red Hat Customer Portal.

2. Select the appropriate version from the **Version** drop-down list.

3. Click **Download Now** next to the **OpenShift v4.18 macOS Clients** entry and save the file.

NOTE

For macOS arm64, choose the **OpenShift v4.18 macOS arm64 Client**entry.

4. Unpack and unzip the archive.

5. Move the **oc** binary to a directory on your PATH.
   To check your **PATH**, open a terminal and execute the following command:

   ```
   $ echo $PATH
   ```

Verification

- Verify your installation by using an **oc** command:

  ```
  $ oc <command>
  ```

3.1.5. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.18, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- For more information about the Telemetry service, see About remote health monitoring

3.1.6. Preparing an Azure Disk Encryption Set

The OpenShift Container Platform installer can use an existing Disk Encryption Set with a user-managed key. To enable this feature, you can create a Disk Encryption Set in Azure and provide the key to the installer.

Procedure

1. Set the environment variables for the Azure resource group by running the following command:

   ```
   $ export RESOURCEGROUP="<resource_group>" \ 1
     LOCATION="<location>" 2
   ```

   1 Specifies the name of the Azure resource group where the Disk Encryption Set and encryption key are to be created. To prevent losing access to your keys when you destroy the cluster, create the Disk Encryption Set in a separate resource group from the one where you install the cluster.

   2 Specifies the Azure location where the resource group is to be created.

2. Set the environment variables for the Azure Key Vault and Disk Encryption Set by running the following command:

   ```
   $ export KEYVAULT_NAME="<keyvault_name>" \ 1
       KEYVAULT_KEY_NAME="<keyvault_key_name>" \ 2
       DISK_ENCRYPTION_SET_NAME="<disk_encryption_set_name>" 3
   ```

   1 Specifies the name of the Azure Key Vault to be created.

   2 Specifies the name of the encryption key to be created.

   3 Specifies the name of the disk encryption set to be created.

3. Set the environment variable for the ID of your Azure service principal by running the following command:

   ```
   $ export CLUSTER_SP_ID="<service_principal_id>" 1
   ```

   1 Specifies the ID of the service principal to be used for installation.

4. Enable host-level encryption in Azure by running the following command:

   ```
   $ az feature register --namespace "Microsoft.Compute" --name "EncryptionAtHost"
   ```

   ```
   $ az feature show --namespace Microsoft.Compute --name EncryptionAtHost
   ```

   ```
   $ az provider register -n Microsoft.Compute
   ```

5. Create an Azure resource group to hold the disk encryption set and associated resources by running the following command:

   ```
   $ az group create --name $RESOURCEGROUP --location $LOCATION
   ```

6. Create an Azure Key Vault by running the following command:

   ```
   $ az keyvault create -n $KEYVAULT_NAME -g $RESOURCEGROUP -l $LOCATION \
       --enable-purge-protection true
   ```

7. Create an encryption key in the key vault by running the following command:

   ```
   $ az keyvault key create --vault-name $KEYVAULT_NAME -n $KEYVAULT_KEY_NAME \
       --protection software
   ```

8. Capture the ID of the key vault by running the following command:

   ```
   $ KEYVAULT_ID=$(az keyvault show --name $KEYVAULT_NAME --query "[id]" -o tsv)
   ```

9. Capture the key URL in the key vault by running the following command:

```
$ KEYVAULT_KEY_URL=$(az keyvault key show --vault-name $KEYVAULT_NAME --name
\
    $KEYVAULT_KEY_NAME --query "[key.kid]" -o tsv)
```

10. Create a disk encryption set by running the following command:

```
$ az disk-encryption-set create -n $DISK_ENCRYPTION_SET_NAME -l $LOCATION -g \
    $RESOURCEGROUP --source-vault $KEYVAULT_ID --key-url $KEYVAULT_KEY_URL
```

11. Grant the **DiskEncryptionSet** resource access to the key vault by running the following commands:

```
$ DES_IDENTITY=$(az disk-encryption-set show -n $DISK_ENCRYPTION_SET_NAME -g \
    $RESOURCEGROUP --query "[identity.principalId]" -o tsv)
```

```
$ az keyvault set-policy -n $KEYVAULT_NAME -g $RESOURCEGROUP --object-id \
    $DES_IDENTITY --key-permissions wrapkey unwrapkey get
```

12. Grant the Azure service principal permission to read the Disk Encryption Set by running the following commands:

```
$ DES_RESOURCE_ID=$(az disk-encryption-set show -n
$DISK_ENCRYPTION_SET_NAME -g \
    $RESOURCEGROUP --query "[id]" -o tsv)
```

```
$ az role assignment create --assignee $CLUSTER_SP_ID --role "<reader_role>" \ 1
    --scope $DES_RESOURCE_ID -o jsonc
```

1 Specifies an Azure role with read permissions to the disk encryption set. You can use the **Owner** role or a custom role with the necessary permissions.

Next steps

- Install an OpenShift Container Platform cluster:
  - Install a cluster with customizations on installer-provisioned infrastructure
  - Install a cluster with network customizations on installer-provisioned infrastructure
  - Install a cluster into an existing VNet on installer-provisioned infrastructure
  - Install a private cluster on installer-provisioned infrastructure
  - Install a cluster into an government region on installer-provisioned infrastructure

3.2. INSTALLING A CLUSTER ON AZURE

You can install a cluster on Microsoft Azure that uses the default configuration options.

3.2.1. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- You have configured an account with the cloud platform that hosts your cluster.
- You have the OpenShift Container Platform installation program and the pull secret for your cluster.
- You have an Azure subscription ID and tenant ID.
- You have the application ID and password of a service principal.

Procedure

1. Optional: If you have run the installation program on this computer before, and want to use an alternative service principal, go to the **~/.azure/** directory and delete the **osServicePrincipal.json** configuration file.
   Deleting this file prevents the installation program from automatically reusing subscription and authentication values from a previous installation.

2. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```
   $ ./openshift-install create cluster --dir <installation_directory> \ 1
       --log-level=info 2
   ```

   1 For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

   2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

   When specifying the directory:

   - Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
   - Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

3. Provide values at the prompts:

   a. Optional: Select an SSH key to use to access your cluster machines.

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

b. Select **azure** as the platform to target.
If the installation program cannot locate the **osServicePrincipal.json** configuration file from a previous installation, you are prompted for Azure subscription and authentication values.

c. Specify the following Azure parameter values for your subscription and service principal:

- **azure subscription id** Enter the subscription ID to use for the cluster.
- **azure tenant id** Enter the tenant ID.
- **azure service principal client id** Enter its application ID.
- **azure service principal client secret** Enter its password.

d. Select the region to deploy the cluster to.

e. Select the base domain to deploy the cluster to. The base domain corresponds to the Azure DNS Zone that you created for your cluster.

f. Enter a descriptive name for your cluster.

IMPORTANT

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see Resolve reserved resource name errors in the Azure documentation.

g. Paste the pull secret from Red Hat OpenShift Cluster Manager .

If previously not detected, the installation program creates an **osServicePrincipal.json** configuration file and stores this file in the ~/**.azure**/ directory on your computer. This ensures that the installation program can load the profile when it is creating an OpenShift Container Platform cluster on the target platform.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/.openshift_install.log**.

IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

3.2.2. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.

- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
   ```

   1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```

Example output

```
system:admin
```

Additional resources

- For more information about accessing and understanding the OpenShift Container Platform web console, see Accessing the web console.

3.2.3. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting .

3.3. INSTALLING A CLUSTER ON AZURE WITH CUSTOMIZATIONS

You can install a customized cluster on infrastructure that the installation program provisions on Microsoft Azure. To customize the installation, modify parameters in the **install-config.yaml** file before you install the cluster.

3.3.1. Using the Azure Marketplace offering

Using the Azure Marketplace offering lets you deploy an OpenShift Container Platform cluster, which is billed on pay-per-use basis (hourly, per core) through Azure, while still being supported directly by Red Hat.

To deploy an OpenShift Container Platform cluster using the Azure Marketplace offering, you must first obtain the Azure Marketplace image. The installation program uses this image to deploy worker or control plane nodes. When obtaining your image, consider the following:

- While the images are the same, the Azure Marketplace publisher is different depending on your region. If you are located in North America, specify **redhat** as the publisher. If you are located in EMEA, specify **redhat-limited** as the publisher.
- The offer includes a **rh-ocp-worker** SKU and a **rh-ocp-worker-gen1** SKU. The **rh-ocp-worker** SKU represents a Hyper-V generation version 2 VM image. The default instance types used in OpenShift Container Platform are version 2 compatible. If you plan to use an instance type that is only version 1 compatible, use the image associated with the **rh-ocp-worker-gen1** SKU. The **rh-ocp-worker-gen1** SKU represents a Hyper-V version 1 VM image.

IMPORTANT

Installing images with the Azure marketplace is not supported on clusters with 64-bit ARM instances.

You should only modify the RHCOS image for compute machines to use an Azure Marketplace image. Control plane machines and infrastructure nodes do not require an OpenShift Container Platform subscription and use the public RHCOS default image by default, which does not incur subscription costs on your Azure bill. Therefore, you should not modify the cluster default boot image or the control plane boot images. Applying the Azure Marketplace image to them will incur additional licensing costs that cannot be recovered.

Prerequisites

- You have installed the Azure CLI client **(az)**.
- Your Azure account is entitled for the offer and you have logged into this account with the Azure CLI client.

Procedure

1. Display all of the available OpenShift Container Platform images by running one of the following commands:
   - North America:

     ```
     $  az vm image list --all --offer rh-ocp-worker --publisher redhat -o table
     ```

     Example output

     ```
     Offer         Publisher       Sku                 Urn                                                          Version
     -------------  --------------  ------------------  --------------------------------------------------------------
     ----  -----------------
     rh-ocp-worker  RedHat          rh-ocp-worker       RedHat:rh-ocp-worker:rh-ocp-
     worker:4.15.2024072409            4.15.2024072409
     rh-ocp-worker  RedHat          rh-ocp-worker-gen1  RedHat:rh-ocp-worker:rh-ocp-worker-
     gen1:4.15.2024072409         4.15.2024072409
     ```

   - EMEA:

     ```
     $  az vm image list --all --offer rh-ocp-worker --publisher redhat-limited -o table
     ```

     Example output

     ```
     Offer         Publisher       Sku                 Urn
     Version
     -------------  --------------  ------------------  --------------------------------------------------------------
     ----           -----------------
     rh-ocp-worker  redhat-limited  rh-ocp-worker       redhat-limited:rh-ocp-worker:rh-ocp-
     worker:4.15.2024072409            4.15.2024072409
     rh-ocp-worker  redhat-limited  rh-ocp-worker-gen1  redhat-limited:rh-ocp-worker:rh-ocp-
     worker-gen1:4.15.2024072409         4.15.2024072409
     ```

   NOTE

   Use the latest image that is available for compute and control plane nodes. If required, your VMs are automatically upgraded as part of the installation process.

2. Inspect the image for your offer by running one of the following commands:
   - North America:

     ```
     $ az vm image show --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
     ```

   - EMEA:

```
$ az vm image show --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
```

3. Review the terms of the offer by running one of the following commands:
   - North America:

     ```
     $ az vm image terms show --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
     ```

   - EMEA:

     ```
     $ az vm image terms show --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
     ```

4. Accept the terms of the offering by running one of the following commands:
   - North America:

     ```
     $ az vm image terms accept --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
     ```

   - EMEA:

     ```
     $ az vm image terms accept --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
     ```

5. Record the image details of your offer. You must update the **compute** section in the **install-config.yaml** file with values for **publisher**, **offer**, **sku**, and **version** before deploying the cluster. You may also update the **controlPlane** section to deploy control plane machines with the specified image details, or the **defaultMachinePlatform** section to deploy both control plane and compute machines with the specified image details. Use the latest available image for control plane and compute nodes.

Sample **install-config.yaml** file with the Azure Marketplace compute nodes

```
apiVersion: v1
baseDomain: example.com
compute:
- hyperthreading: Enabled
  name: worker
  platform:
    azure:
      type: Standard_D4s_v5
      osImage:
        publisher: redhat
        offer: rh-ocp-worker
        sku: rh-ocp-worker
        version: 413.92.2023101700
  replicas: 3
```

3.3.2. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Microsoft Azure.

Prerequisites

- You have the OpenShift Container Platform installation program and the pull secret for your cluster.
- You have an Azure subscription ID and tenant ID.
- If you are installing the cluster using a service principal, you have its application ID and password.
- If you are installing the cluster using a system-assigned managed identity, you have enabled it on the virtual machine that you will run the installation program from.
- If you are installing the cluster using a user-assigned managed identity, you have met these prerequisites:
  - You have its client ID.
  - You have assigned it to the virtual machine that you will run the installation program from.

Procedure

1. Optional: If you have run the installation program on this computer before, and want to use an alternative service principal or managed identity, go to the ~/**.azure**/ directory and delete the **osServicePrincipal.json** configuration file.
   Deleting this file prevents the installation program from automatically reusing subscription and authentication values from a previous installation.

2. Create the **install-config.yaml** file.

   a. Change to the directory that contains the installation program and run the following command:

      ```
      $ ./openshift-install create install-config --dir <installation_directory> ①
      ```

      ① For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

      When specifying the directory:

      - Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
      - Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

   b. At the prompts, provide the configuration details for your cloud:

      i. Optional: Select an SSH key to use to access your cluster machines.

         NOTE

         For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

    ii. Select **azure** as the platform to target.
    If the installation program cannot locate the **osServicePrincipal.json** configuration file from a previous installation, you are prompted for Azure subscription and authentication values.

    iii. Enter the following Azure parameter values for your subscription:

    - **azure subscription id** Enter the subscription ID to use for the cluster.
    - **azure tenant id** Enter the tenant ID.

    iv. Depending on the Azure identity you are using to deploy the cluster, do one of the following when prompted for the **azure service principal client id**

    - If you are using a service principal, enter its application ID.
    - If you are using a system-assigned managed identity, leave this value blank.
    - If you are using a user-assigned managed identity, specify its client ID.

    v. Depending on the Azure identity you are using to deploy the cluster, do one of the following when prompted for the **azure service principal client secret**

    - If you are using a service principal, enter its password.
    - If you are using a system-assigned managed identity, leave this value blank.
    - If you are using a user-assigned managed identity, leave this value blank.

    vi. Select the region to deploy the cluster to.

    vii. Select the base domain to deploy the cluster to. The base domain corresponds to the Azure DNS Zone that you created for your cluster.

    viii. Enter a descriptive name for your cluster.

    IMPORTANT

    All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see Resolve reserved resource name errors in the Azure documentation.

3. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.

   NOTE

   If you are installing a three-node cluster, be sure to set the **compute.replicas** parameter to **0**. This ensures that the cluster’s control planes are schedulable. For more information, see "Installing a three-node cluster on Azure".

4. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

If previously not detected, the installation program creates an **osServicePrincipal.json** configuration file and stores this file in the ~/**.azure**/ directory on your computer. This ensures that the installation program can load the profile when it is creating an OpenShift Container Platform cluster on the target platform.

Additional resources

- Installation configuration parameters for Azure

3.3.2.1. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 3.1. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or Hyper-Threading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

NOTE

For OpenShift Container Platform version 4.18, RHCOS is based on RHEL version 9.4, which updates the micro-architecture requirements. The following list contains the minimum instruction set architectures (ISA) that each architecture requires:

- x86-64 architecture requires x86-64-v2 ISA
- ARM64 architecture requires ARMv8.0-A ISA
- IBM Power architecture requires Power 9 ISA
- s390x architecture requires z14 ISA

For more information, see Architectures (RHEL documentation).

IMPORTANT

You are required to use Azure virtual machines that have the **premiumIO** parameter set to **true**.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- Optimizing storage

3.3.2.2. Tested instance types for Azure

The following Microsoft Azure instance types have been tested with OpenShift Container Platform.

Example 3.1. Machine types based on 64-bit x86 architecture

- **standardBasv2Family**
- **standardBSFamily**
- **standardBsv2Family**
- **standardDADSv5Family**
- **standardDASv4Family**
- **standardDASv5Family**
- **standardDCACCV5Family**
- **standardDCADCCV5Family**
- **standardDCADSv5Family**
- **standardDCASv5Family**
- **standardDCSv3Family**

- **standardDCSv2Family**
- **standardDDCSv3Family**
- **standardDDSv4Family**
- **standardDDSv5Family**
- **standardDLDSv5Family**
- **standardDLSv5Family**
- **standardDSFamily**
- **standardDSv2Family**
- **standardDSv2PromoFamily**
- **standardDSv3Family**
- **standardDSv4Family**
- **standardDSv5Family**
- **standardEADSv5Family**
- **standardEASv4Family**
- **standardEASv5Family**
- **standardEBDSv5Family**
- **standardEBSv5Family**
- **standardECACCV5Family**
- **standardECADCCV5Family**
- **standardECADSv5Family**
- **standardECASv5Family**
- **standardEDSv4Family**
- **standardEDSv5Family**
- **standardEIADSv5Family**
- **standardEIASv4Family**
- **standardEIASv5Family**
- **standardEIBDSv5Family**
- **standardEIBSv5Family**
- **standardEIDSv5Family**

- **standardEISv3Family**
- **standardEISv5Family**
- **standardESv3Family**
- **standardESv4Family**
- **standardESv5Family**
- **standardFXMDVSFamily**
- **standardFSFamily**
- **standardFSv2Family**
- **standardGSFamily**
- **standardHBrsv2Family**
- **standardHBSFamily**
- **standardHBv4Family**
- **standardHCSFamily**
- **standardHXFamily**
- **standardLASv3Family**
- **standardLSFamily**
- **standardLSv2Family**
- **standardLSv3Family**
- **standardMDSHighMemoryv3Family**
- **standardMDSMediumMemoryv2Family**
- **standardMDSMediumMemoryv3Family**
- **standardMIDSHighMemoryv3Family**
- **standardMIDSMediumMemoryv2Family**
- **standardMISHighMemoryv3Family**
- **standardMISMediumMemoryv2Family**
- **standardMSFamily**
- **standardMSHighMemoryv3Family**
- **standardMSMediumMemoryv2Family**
- **standardMSMediumMemoryv3Family**

- StandardNCADSA100v4Family
- Standard NCASv3_T4 Family
- standardNCSv3Family
- standardNDSv2Family
- StandardNGADSV620v1Family
- standardNPSFamily
- StandardNVADSA10v5Family
- standardNVSv3Family
- standardXEISv4Family

3.3.2.3. Tested instance types for Azure on 64-bit ARM infrastructures

The following Microsoft Azure ARM64 instance types have been tested with OpenShift Container Platform.

Example 3.2. Machine types based on 64-bit ARM architecture

- standardBpsv2Family
- standardDPSv5Family
- standardDPDSv5Family
- standardDPLDSv5Family
- standardDPLSv5Family
- standardEPSv5Family
- standardEPDSv5Family
- StandardDpdsv6Family
- StandardDpldsv6Famil
- StandardDplsv6Family
- StandardDpsv6Family
- StandardEpdsv6Family
- StandardEpsv6Family

3.3.2.4. Enabling trusted launch for Azure VMs

You can enable two trusted launch features when installing your cluster on Azure: secure boot and virtualized Trusted Platform Modules.

For more information about the sizes of virtual machines that support the trusted launch features, see Virtual machine sizes.

IMPORTANT

Trusted launch is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see Technology Preview Features Support Scope .

Prerequisites

- You have created an **install-config.yaml** file.

Procedure

- Edit the **install-config.yaml** file before deploying your cluster:
  - Enable trusted launch only on control plane by adding the following stanza:

    ```
    controlPlane:
      platform:
        azure:
          settings:
            securityType: TrustedLaunch
            trustedLaunch:
              uefiSettings:
                secureBoot: Enabled
                virtualizedTrustedPlatformModule: Enabled
    ```

  - Enable trusted launch only on compute node by adding the following stanza:

    ```
    compute:
      platform:
        azure:
          settings:
            securityType: TrustedLaunch
            trustedLaunch:
              uefiSettings:
                secureBoot: Enabled
                virtualizedTrustedPlatformModule: Enabled
    ```

  - Enable trusted launch on all nodes by adding the following stanza:

    ```
    platform:
      azure:
        settings:
          securityType: TrustedLaunch
    ```

```
      trustedLaunch:
        uefiSettings:
          secureBoot: Enabled
          virtualizedTrustedPlatformModule: Enabled
```

3.3.2.5. Enabling confidential VMs

You can enable confidential VMs when installing your cluster. You can enable confidential VMs for compute nodes, control plane nodes, or all nodes.

IMPORTANT

Using confidential VMs is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see Technology Preview Features Support Scope .

You can use confidential VMs with the following VM sizes:

- DCasv5-series
- DCadsv5-series
- ECasv5-series
- ECadsv5-series

IMPORTANT

Confidential VMs are currently not supported on 64-bit ARM architectures.

Prerequisites

- You have created an **install-config.yaml** file.

Procedure

- Edit the **install-config.yaml** file before deploying your cluster:
  - Enable confidential VMs only on control plane by adding the following stanza:

```
controlPlane:
  platform:
    azure:
      settings:
        securityType: ConfidentialVM
        confidentialVM:
          uefiSettings:
            secureBoot: Enabled
            virtualizedTrustedPlatformModule: Enabled
      osDisk:
        securityProfile:
          securityEncryptionType: VMGuestStateOnly
```

- Enable confidential VMs only on compute nodes by adding the following stanza:

```
compute:
  platform:
    azure:
      settings:
        securityType: ConfidentialVM
        confidentialVM:
          uefiSettings:
            secureBoot: Enabled
            virtualizedTrustedPlatformModule: Enabled
      osDisk:
        securityProfile:
          securityEncryptionType: VMGuestStateOnly
```

- Enable confidential VMs on all nodes by adding the following stanza:

```
platform:
  azure:
    settings:
      securityType: ConfidentialVM
      confidentialVM:
        uefiSettings:
          secureBoot: Enabled
          virtualizedTrustedPlatformModule: Enabled
    osDisk:
      securityProfile:
        securityEncryptionType: VMGuestStateOnly
```

3.3.2.6. Sample customized install-config.yaml file for Azure

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

> **IMPORTANT**
>
> This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```
apiVersion: v1
baseDomain: example.com 1
controlPlane: 2
  hyperthreading: Enabled 3 4
  name: master
  platform:
    azure:
      encryptionAtHost: true
      ultraSSDCapability: Enabled
      osDisk:
        diskSizeGB: 1024 5
        diskType: Premium_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      type: Standard_D8s_v3
  replicas: 3
compute: 6
- hyperthreading: Enabled 7 8
  name: worker
  platform:
    azure:
      ultraSSDCapability: Enabled
      type: Standard_D2s_v3
      encryptionAtHost: true
      osDisk:
        diskSizeGB: 512 9
        diskType: Standard_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      zones: 10
      - "1"
      - "2"
      - "3"
  replicas: 5
metadata:
  name: test-cluster 11
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 12
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    defaultMachinePlatform:
      osImage: 13
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      ultraSSDCapability: Enabled
    baseDomainResourceGroupName: resource_group 14
    region: centralus 15
    resourceGroupName: existing_resource_group 16
    outboundType: Loadbalancer
    cloudName: AzurePublicCloud
pullSecret: '{"auths": ...}' 17
fips: false 18
sshKey: ssh-ed25519 AAAA... 19
```

1 11 15 17 Required. The installation program prompts you for this value.

2 6 If you do not provide these parameters and values, the installation program provides the default value.

3 7 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, **-**, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

4 8 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger virtual machine types, such as **Standard_D8s_v3**, for your machines if you disable simultaneous multithreading.

5 9 You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes is 1024 GB.

10 Specify a list of zones to deploy your machines to. For high availability, specify at least two zones.

12 The cluster network plugin to install. The default value **OVNKubernetes** is the only supported value.

13 Optional: A custom Red Hat Enterprise Linux CoreOS (RHCOS) image that should be used to boot control plane and compute machines. The **publisher**, **offer**, **sku**, and **version** parameters under **platform.azure.defaultMachinePlatform.osImage** apply to both control plane and compute machines. If the parameters under **controlPlane.platform.azure.osImage** or **compute.platform.azure.osImage** are set, they override the **platform.azure.defaultMachinePlatform.osImage** parameters.

14 Specify the name of the resource group that contains the DNS zone for your base domain.

16 Specify the name of an already existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.

18 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is

IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Switching RHEL to FIPS mode.

When running Red Hat Enterprise Linux (RHEL) or Red Hat Enterprise Linux CoreOS (RHCOS) booted in FIPS mode, OpenShift Container Platform core components use the RHEL cryptographic libraries that have been submitted to NIST for FIPS 140-2/140-3 Validation on only the x86_64, ppc64le, and s390x architectures.

19 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

3.3.2.7. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.

NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
```

```
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.

2. A proxy URL to use for creating HTTPS connections outside the cluster.

3. A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use * to bypass the proxy for all destinations.

4. If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

5. Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

NOTE

The installation program does not support the proxy **readinessEndpoints** field.

NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

Additional resources

- For more details about Accelerated Networking, see Accelerated Networking for Microsoft Azure VMs.

3.3.3. Configuring user-defined tags for Azure

In OpenShift Container Platform, you can use tags for grouping resources and for managing resource access and cost. Tags are applied only to the resources created by the OpenShift Container Platform installation program and its core Operators such as Machine API Operator, Cluster Ingress Operator, Cluster Image Registry Operator. The OpenShift Container Platform consists of the following types of tags:

OpenShift Container Platform tags

By default, OpenShift Container Platform installation program attaches the OpenShift Container Platform tags to the Azure resources. These OpenShift Container Platform tags are not accessible to the users. The format of the OpenShift Container Platform tags is **kubernetes.io_cluster.<cluster_id>:owned**, where **<cluster_id>** is the value of **.status.infrastructureName** in the infrastructure resource for the cluster.

User-defined tags

User-defined tags are manually created in **install-config.yaml** file during installation. When creating the user-defined tags, you must consider the following points:

- User-defined tags on Azure resources can only be defined during OpenShift Container Platform cluster creation, and cannot be modified after the cluster is created.
- Support for user-defined tags is available only for the resources created in the Azure Public Cloud.
- User-defined tags are not supported for the OpenShift Container Platform clusters upgraded to OpenShift Container Platform 4.18.

3.3.3.1. Creating user-defined tags for Azure

To define the list of user-defined tags, edit the **.platform.azure.userTags** field in the **install-config.yaml** file.

Procedure

- Specify the **.platform.azure.userTags** field as shown in the following **install-config.yaml** file:

```
apiVersion: v1
baseDomain: example.com
#...
platform:
  azure:
    userTags: ①
      <key>: <value> ②
#...
```

① Defines the additional keys and values that the installation program adds as tags to all Azure resources that it creates.

② Specify the key and value. You can configure a maximum of 10 tags for resource group and resources. Tag keys are case-insensitive. For more information on requirements for specifying user-defined tags, see "User-defined tags requirements" section.

Example install-config.yaml file

```
apiVersion: v1
baseDomain: example.com
#...
platform:
  azure:
    userTags:
      createdBy: user
      environment: dev
#...
```

Verification

- Access the list of created user-defined tags for the Azure resources by running the following command:

```
$ oc get infrastructures.config.openshift.io cluster -o=jsonpath-as-
json='{.status.platformStatus.azure.resourceTags}'
```

Example output

```
[
    [
        {
            "key": "createdBy",
            "value": "user"
        },
        {
            "key": "environment",
            "value": "dev"
        }
    ]
]
```

3.3.3.2. User-defined tags requirements

The user-defined tags have the following requirements:

- A tag key must have a maximum of 128 characters.
- A tag key must begin with a letter.

- A tag key must end with a letter, number or underscore.
- A tag key must contain only letters, numbers, underscores(**_**), periods(**.**), and hyphens(**-**).
- A tag key must not be specified as **name**.
- A tag key must not have the following prefixes:
  - **kubernetes.io**
  - **openshift.io**
  - **microsoft**
  - **azure**
  - **windows**
- A tag value must have a maximum of 256 characters.

For more information about Azure tags, see Azure user-defined tags.

3.3.4. Alternatives to storing administrator-level secrets in the kube-system project

By default, administrator secrets are stored in the **kube-system** project. If you configured the **credentialsMode** parameter in the **install-config.yaml** file to **Manual**, you must use one of the following alternatives:

- To manage long-term cloud credentials manually, follow the procedure in Manually creating long-term credentials.
- To implement short-term credentials that are managed outside the cluster for individual components, follow the procedures in Configuring an Azure cluster to use short-term credentials.

3.3.4.1. Manually creating long-term credentials

The Cloud Credential Operator (CCO) can be put into manual mode prior to installation in environments where the cloud identity and access management (IAM) APIs are not reachable, or the administrator prefers not to store an administrator-level credential secret in the cluster **kube-system** namespace.

Procedure

1. If you did not set the **credentialsMode** parameter in the **install-config.yaml** configuration file to **Manual**, modify the value as shown:

   Sample configuration file snippet

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. If you have not previously created installation manifest files, do so by running the following command:

```
$ openshift-install create manifests --dir <installation_directory>
```

where **<installation_directory>** is the directory in which the installation program creates files.

3. Set a **$RELEASE_IMAGE** variable with the release image from your installation file by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

4. Extract the list of **CredentialsRequest** custom resources (CRs) from the OpenShift Container Platform release image by running the following command:

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1 The **--included** parameter includes only the manifests that your specific cluster configuration requires.

2 Specify the location of the **install-config.yaml** file.

3 Specify the path to the directory where you want to store the **CredentialsRequest** objects. If the specified directory does not exist, this command creates it.

This command creates a YAML file for each **CredentialsRequest** object.

Sample CredentialsRequest object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
  ...
```

5. Create YAML files for secrets in the **openshift-install** manifests directory that you generated previously. The secrets must be stored using the namespace and secret name defined in the **spec.secretRef** for each **CredentialsRequest** object.

Sample CredentialsRequest object with secrets

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
```

```
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Sample Secret object

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  azure_subscription_id: <base64_encoded_azure_subscription_id>
  azure_client_id: <base64_encoded_azure_client_id>
  azure_client_secret: <base64_encoded_azure_client_secret>
  azure_tenant_id: <base64_encoded_azure_tenant_id>
  azure_resource_prefix: <base64_encoded_azure_resource_prefix>
  azure_resourcegroup: <base64_encoded_azure_resourcegroup>
  azure_region: <base64_encoded_azure_region>
```

IMPORTANT

Before upgrading a cluster that uses manually maintained credentials, you must ensure that the CCO is in an upgradeable state.

3.3.4.2. Configuring an Azure cluster to use short-term credentials

To install a cluster that uses Microsoft Entra Workload ID, you must configure the Cloud Credential Operator utility and create the required Azure resources for your cluster.

3.3.4.2.1. Configuring the Cloud Credential Operator utility

To create and manage cloud credentials from outside of the cluster when the Cloud Credential Operator (CCO) is operating in manual mode, extract and prepare the CCO utility (**ccoctl**) binary.

NOTE

The **ccoctl** utility is a Linux binary that must run in a Linux environment.

Prerequisites

- You have access to an OpenShift Container Platform account with cluster administrator access.

- You have installed the OpenShift CLI (**oc**).
- You have created a global Azure account for the **ccoctl** utility to use with the following permissions:
  - **Microsoft.Resources/subscriptions/resourceGroups/read**
  - **Microsoft.Resources/subscriptions/resourceGroups/write**
  - **Microsoft.Resources/subscriptions/resourceGroups/delete**
  - **Microsoft.Authorization/roleAssignments/read**
  - **Microsoft.Authorization/roleAssignments/delete**
  - **Microsoft.Authorization/roleAssignments/write**
  - **Microsoft.Authorization/roleDefinitions/read**
  - **Microsoft.Authorization/roleDefinitions/write**
  - **Microsoft.Authorization/roleDefinitions/delete**
  - **Microsoft.Storage/storageAccounts/listkeys/action**
  - **Microsoft.Storage/storageAccounts/delete**
  - **Microsoft.Storage/storageAccounts/read**
  - **Microsoft.Storage/storageAccounts/write**
  - **Microsoft.Storage/storageAccounts/blobServices/containers/delete**
  - **Microsoft.Storage/storageAccounts/blobServices/containers/read**
  - **Microsoft.Storage/storageAccounts/blobServices/containers/write**
  - **Microsoft.ManagedIdentity/userAssignedIdentities/delete**
  - **Microsoft.ManagedIdentity/userAssignedIdentities/read**
  - **Microsoft.ManagedIdentity/userAssignedIdentities/write**
  - **Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/read**
  - **Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/write**
  - **Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/delete**
  - **Microsoft.Storage/register/action**
  - **Microsoft.ManagedIdentity/register/action**

Procedure

1. Set a variable for the OpenShift Container Platform release image by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. Obtain the CCO container image from the OpenShift Container Platform release image by running the following command:

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

NOTE

Ensure that the architecture of the **$RELEASE_IMAGE** matches the architecture of the environment in which you will use the **ccoctl** tool.

3. Extract the **ccoctl** binary from the CCO container image within the OpenShift Container Platform release image by running the following command:

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 For **<rhel_version>**, specify the value that corresponds to the version of Red Hat Enterprise Linux (RHEL) that the host uses. If no value is specified, **ccoctl.rhel8** is used by default. The following values are valid:

- **rhel8**: Specify this value for hosts that use RHEL 8.
- **rhel9**: Specify this value for hosts that use RHEL 9.

4. Change the permissions to make **ccoctl** executable by running the following command:

```
$ chmod 775 ccoctl.<rhel_version>
```

Verification

- To verify that **ccoctl** is ready to use, display the help file. Use a relative file name when you run the command, for example:

```
$ ./ccoctl.rhel9
```

Example output

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws          Manage credentials objects for AWS cloud
  azure        Manage credentials objects for Azure
  gcp          Manage credentials objects for Google cloud
  help         Help about any command
  ibmcloud     Manage credentials objects for {ibm-cloud-title}
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

3.3.4.2.2. Creating Azure resources with the Cloud Credential Operator utility

You can use the **ccoctl azure create-all** command to automate the creation of Azure resources.

> NOTE
>
> By default, **ccoctl** creates objects in the directory in which the commands are run. To create the objects in a different directory, use the **--output-dir** flag. This procedure uses **<path_to_ccoctl_output_dir>** to refer to this directory.

Prerequisites

You must have:

- Extracted and prepared the **ccoctl** binary.
- Access to your Microsoft Azure account by using the Azure CLI.

Procedure

1. Set a **$RELEASE_IMAGE** variable with the release image from your installation file by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. Extract the list of **CredentialsRequest** objects from the OpenShift Container Platform release image by running the following command:

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1 The **--included** parameter includes only the manifests that your specific cluster configuration requires.

2 Specify the location of the **install-config.yaml** file.

3 Specify the path to the directory where you want to store the **CredentialsRequest** objects. If the specified directory does not exist, this command creates it.

> NOTE
>
> This command might take a few moments to run.

3. To enable the **ccoctl** utility to detect your Azure credentials automatically, log in to the Azure CLI by running the following command:

    ```
    $ az login
    ```

4. Use the **ccoctl** tool to process all **CredentialsRequest** objects by running the following command:

    ```
    $ ccoctl azure create-all \
      --name=<azure_infra_name> \ 1
      --output-dir=<ccoctl_output_dir> \ 2
      --region=<azure_region> \ 3
      --subscription-id=<azure_subscription_id> \ 4
      --credentials-requests-dir=<path_to_credentials_requests_directory> \ 5
      --dnszone-resource-group-name=<azure_dns_zone_resource_group_name> \ 6
      --tenant-id=<azure_tenant_id> 7
    ```

    1 Specify the user-defined name for all created Azure resources used for tracking.

    2 Optional: Specify the directory in which you want the **ccoctl** utility to create objects. By default, the utility creates objects in the directory in which the commands are run.

    3 Specify the Azure region in which cloud resources will be created.

    4 Specify the Azure subscription ID to use.

    5 Specify the directory containing the files for the component **CredentialsRequest** objects.

    6 Specify the name of the resource group containing the cluster's base domain Azure DNS zone.

    7 Specify the Azure tenant ID to use.

    NOTE

    If your cluster uses Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set, you must include the **--enable-tech-preview** parameter.

    To see additional optional parameters and explanations of how to use them, run the **azure create-all --help** command.

Verification

- To verify that the OpenShift Container Platform secrets are created, list the files in the **<path_to_ccoctl_output_dir>/manifests** directory:

    ```
    $ ls <path_to_ccoctl_output_dir>/manifests
    ```

    Example output

    ```
    azure-ad-pod-identity-webhook-config.yaml
    cluster-authentication-02-config.yaml
    ```

```
openshift-cloud-controller-manager-azure-cloud-credentials-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capz-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-azure-disk-credentials-credentials.yaml
openshift-cluster-csi-drivers-azure-file-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-azure-cloud-credentials-credentials.yaml
```

You can verify that the Microsoft Entra ID service accounts are created by querying Azure. For more information, refer to Azure documentation on listing Entra ID service accounts.

3.3.4.2.3. Incorporating the Cloud Credential Operator utility manifests

To implement short-term security credentials managed outside the cluster for individual components, you must move the manifest files that the Cloud Credential Operator utility (**ccoctl**) created to the correct directories for the installation program.

Prerequisites

- You have configured an account with the cloud platform that hosts your cluster.
- You have configured the Cloud Credential Operator utility (**ccoctl**).
- You have created the cloud provider resources that are required for your cluster with the **ccoctl** utility.

Procedure

1. If you did not set the **credentialsMode** parameter in the **install-config.yaml** configuration file to **Manual**, modify the value as shown:

   Sample configuration file snippet

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. If you used the **ccoctl** utility to create a new Azure resource group instead of using an existing resource group, modify the **resourceGroupName** parameter in the **install-config.yaml** as shown:

   Sample configuration file snippet

   ```
   apiVersion: v1
   baseDomain: example.com
   # ...
   platform:
     azure:
       resourceGroupName: <azure_infra_name> 1
   # ...
   ```

1 This value must match the user-defined name for Azure resources that was specified with the **--name** argument of the **ccoctl azure create-all** command.

3. If you have not previously created installation manifest files, do so by running the following command:

```
$ openshift-install create manifests --dir <installation_directory>
```

   where **<installation_directory>** is the directory in which the installation program creates files.

4. Copy the manifests that the **ccoctl** utility generated to the **manifests** directory that the installation program created by running the following command:

```
$ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
```

5. Copy the **tls** directory that contains the private key to the installation directory:

```
$ cp -a /<path_to_ccoctl_output_dir>/tls .
```

3.3.5. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- You have configured an account with the cloud platform that hosts your cluster.
- You have the OpenShift Container Platform installation program and the pull secret for your cluster.
- You have an Azure subscription ID and tenant ID.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.

2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.

- Credential information also outputs to **<installation_directory>/.openshift_install.log**.

IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

3.3.6. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.

- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
   ```

   1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```

   Example output

   ```
   system:admin
   ```

Additional resources

- See Accessing the web console for more details about accessing and understanding the OpenShift Container Platform web console.

3.3.7. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting .

3.4. INSTALLING A CLUSTER ON AZURE WITH NETWORK CUSTOMIZATIONS

In OpenShift Container Platform version 4.18, you can install a cluster with a customized network configuration on infrastructure that the installation program provisions on Microsoft Azure. By customizing your network configuration, your cluster can coexist with existing IP address allocations in your environment and integrate with existing MTU and VXLAN configurations.

You must set most of the network configuration parameters during installation, and you can modify only **kubeProxy** configuration parameters in a running cluster.

3.4.1. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Microsoft Azure.

Prerequisites

- You have the OpenShift Container Platform installation program and the pull secret for your cluster.
- You have an Azure subscription ID and tenant ID.
- If you are installing the cluster using a service principal, you have its application ID and password.
- If you are installing the cluster using a system-assigned managed identity, you have enabled it on the virtual machine that you will run the installation program from.

- If you are installing the cluster using a user-assigned managed identity, you have met these prerequisites:
  - You have its client ID.
  - You have assigned it to the virtual machine that you will run the installation program from.

Procedure

1. Optional: If you have run the installation program on this computer before, and want to use an alternative service principal or managed identity, go to the **~/.azure/** directory and delete the **osServicePrincipal.json** configuration file.
   Deleting this file prevents the installation program from automatically reusing subscription and authentication values from a previous installation.

2. Create the **install-config.yaml** file.

   a. Change to the directory that contains the installation program and run the following command:

   ```
   $ ./openshift-install create install-config --dir <installation_directory> 1
   ```

   1 For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

   When specifying the directory:

   - Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
   - Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

   b. At the prompts, provide the configuration details for your cloud:

      i. Optional: Select an SSH key to use to access your cluster machines.

      > **NOTE**
      >
      > For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

      ii. Select **azure** as the platform to target.
      If the installation program cannot locate the **osServicePrincipal.json** configuration file from a previous installation, you are prompted for Azure subscription and authentication values.

      iii. Enter the following Azure parameter values for your subscription:

      - **azure subscription id** Enter the subscription ID to use for the cluster.

- **azure tenant id** Enter the tenant ID.

iv. Depending on the Azure identity you are using to deploy the cluster, do one of the following when prompted for the **azure service principal client id**

- If you are using a service principal, enter its application ID.
- If you are using a system-assigned managed identity, leave this value blank.
- If you are using a user-assigned managed identity, specify its client ID.

v. Depending on the Azure identity you are using to deploy the cluster, do one of the following when prompted for the **azure service principal client secret**

- If you are using a service principal, enter its password.
- If you are using a system-assigned managed identity, leave this value blank.
- If you are using a user-assigned managed identity, leave this value blank.

vi. Select the region to deploy the cluster to.

vii. Select the base domain to deploy the cluster to. The base domain corresponds to the Azure DNS Zone that you created for your cluster.

viii. Enter a descriptive name for your cluster.

> IMPORTANT
>
> All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see Resolve reserved resource name errors in the Azure documentation.

3. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.

4. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

> IMPORTANT
>
> The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

If previously not detected, the installation program creates an **osServicePrincipal.json** configuration file and stores this file in the **~/.azure/** directory on your computer. This ensures that the installation program can load the profile when it is creating an OpenShift Container Platform cluster on the target platform.

Additional resources

- Installation configuration parameters for Azure

3.4.1.1. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 3.2. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or Hyper-Threading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

NOTE

For OpenShift Container Platform version 4.18, RHCOS is based on RHEL version 9.4, which updates the micro-architecture requirements. The following list contains the minimum instruction set architectures (ISA) that each architecture requires:

- x86-64 architecture requires x86-64-v2 ISA
- ARM64 architecture requires ARMv8.0-A ISA
- IBM Power architecture requires Power 9 ISA
- s390x architecture requires z14 ISA

For more information, see Architectures (RHEL documentation).

IMPORTANT

You are required to use Azure virtual machines that have the **premiumIO** parameter set to **true**.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- Optimizing storage

3.4.1.2. Tested instance types for Azure

The following Microsoft Azure instance types have been tested with OpenShift Container Platform.

Example 3.3. Machine types based on 64-bit x86 architecture

- **standardBasv2Family**
- **standardBSFamily**
- **standardBsv2Family**
- **standardDADSv5Family**
- **standardDASv4Family**
- **standardDASv5Family**
- **standardDCACCV5Family**
- **standardDCADCCV5Family**
- **standardDCADSv5Family**
- **standardDCASv5Family**
- **standardDCSv3Family**
- **standardDCSv2Family**
- **standardDDCSv3Family**
- **standardDDSv4Family**
- **standardDDSv5Family**
- **standardDLDSv5Family**
- **standardDLSv5Family**
- **standardDSFamily**
- **standardDSv2Family**
- **standardDSv2PromoFamily**
- **standardDSv3Family**
- **standardDSv4Family**

- **standardDSv5Family**
- **standardEADSv5Family**
- **standardEASv4Family**
- **standardEASv5Family**
- **standardEBDSv5Family**
- **standardEBSv5Family**
- **standardECACCV5Family**
- **standardECADCCV5Family**
- **standardECADSv5Family**
- **standardECASv5Family**
- **standardEDSv4Family**
- **standardEDSv5Family**
- **standardEIADSv5Family**
- **standardEIASv4Family**
- **standardEIASv5Family**
- **standardEIBDSv5Family**
- **standardEIBSv5Family**
- **standardEIDSv5Family**
- **standardEISv3Family**
- **standardEISv5Family**
- **standardESv3Family**
- **standardESv4Family**
- **standardESv5Family**
- **standardFXMDVSFamily**
- **standardFSFamily**
- **standardFSv2Family**
- **standardGSFamily**
- **standardHBrsv2Family**
- **standardHBSFamily**

- **standardHBv4Family**
- **standardHCSFamily**
- **standardHXFamily**
- **standardLASv3Family**
- **standardLSFamily**
- **standardLSv2Family**
- **standardLSv3Family**
- **standardMDSHighMemoryv3Family**
- **standardMDSMediumMemoryv2Family**
- **standardMDSMediumMemoryv3Family**
- **standardMIDSHighMemoryv3Family**
- **standardMIDSMediumMemoryv2Family**
- **standardMISHighMemoryv3Family**
- **standardMISMediumMemoryv2Family**
- **standardMSFamily**
- **standardMSHighMemoryv3Family**
- **standardMSMediumMemoryv2Family**
- **standardMSMediumMemoryv3Family**
- **StandardNCADSA100v4Family**
- **Standard NCASv3_T4 Family**
- **standardNCSv3Family**
- **standardNDSv2Family**
- **StandardNGADSV620v1Family**
- **standardNPSFamily**
- **StandardNVADSA10v5Family**
- **standardNVSv3Family**
- **standardXEISv4Family**

3.4.1.3. Tested instance types for Azure on 64-bit ARM infrastructures

The following Microsoft Azure ARM64 instance types have been tested with OpenShift Container Platform.

Example 3.4. Machine types based on 64-bit ARM architecture

- **standardBpsv2Family**
- **standardDPSv5Family**
- **standardDPDSv5Family**
- **standardDPLDSv5Family**
- **standardDPLSv5Family**
- **standardEPSv5Family**
- **standardEPDSv5Family**
- **StandardDpdsv6Family**
- **StandardDpldsv6Famil**
- **StandardDplsv6Family**
- **StandardDpsv6Family**
- **StandardEpdsv6Family**
- **StandardEpsv6Family**

3.4.1.4. Enabling trusted launch for Azure VMs

You can enable two trusted launch features when installing your cluster on Azure: secure boot and virtualized Trusted Platform Modules.

For more information about the sizes of virtual machines that support the trusted launch features, see Virtual machine sizes.

IMPORTANT

Trusted launch is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see Technology Preview Features Support Scope .

Prerequisites

- You have created an **install-config.yaml** file.

Procedure

- Edit the **install-config.yaml** file before deploying your cluster:
  - Enable trusted launch only on control plane by adding the following stanza:

    ```
    controlPlane:
      platform:
        azure:
          settings:
            securityType: TrustedLaunch
            trustedLaunch:
              uefiSettings:
                secureBoot: Enabled
                virtualizedTrustedPlatformModule: Enabled
    ```

  - Enable trusted launch only on compute node by adding the following stanza:

    ```
    compute:
      platform:
        azure:
          settings:
            securityType: TrustedLaunch
            trustedLaunch:
              uefiSettings:
                secureBoot: Enabled
                virtualizedTrustedPlatformModule: Enabled
    ```

  - Enable trusted launch on all nodes by adding the following stanza:

    ```
    platform:
      azure:
        settings:
          securityType: TrustedLaunch
          trustedLaunch:
            uefiSettings:
              secureBoot: Enabled
              virtualizedTrustedPlatformModule: Enabled
    ```

3.4.1.5. Enabling confidential VMs

You can enable confidential VMs when installing your cluster. You can enable confidential VMs for compute nodes, control plane nodes, or all nodes.

> **IMPORTANT**
>
> Using confidential VMs is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.
>
> For more information about the support scope of Red Hat Technology Preview features, see Technology Preview Features Support Scope .

You can use confidential VMs with the following VM sizes:

- DCasv5-series
- DCadsv5-series
- ECasv5-series
- ECadsv5-series

IMPORTANT

Confidential VMs are currently not supported on 64-bit ARM architectures.

Prerequisites

- You have created an **install-config.yaml** file.

Procedure

- Edit the **install-config.yaml** file before deploying your cluster:
  - Enable confidential VMs only on control plane by adding the following stanza:

```
controlPlane:
  platform:
    azure:
      settings:
        securityType: ConfidentialVM
        confidentialVM:
          uefiSettings:
            secureBoot: Enabled
            virtualizedTrustedPlatformModule: Enabled
      osDisk:
        securityProfile:
          securityEncryptionType: VMGuestStateOnly
```

  - Enable confidential VMs only on compute nodes by adding the following stanza:

```
compute:
  platform:
    azure:
      settings:
        securityType: ConfidentialVM
        confidentialVM:
          uefiSettings:
            secureBoot: Enabled
            virtualizedTrustedPlatformModule: Enabled
      osDisk:
        securityProfile:
          securityEncryptionType: VMGuestStateOnly
```

  - Enable confidential VMs on all nodes by adding the following stanza:

```
platform:
```

```
azure:
  settings:
    securityType: ConfidentialVM
    confidentialVM:
      uefiSettings:
        secureBoot: Enabled
        virtualizedTrustedPlatformModule: Enabled
  osDisk:
    securityProfile:
      securityEncryptionType: VMGuestStateOnly
```

3.4.1.6. Sample customized install-config.yaml file for Azure

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```
apiVersion: v1
baseDomain: example.com ❶
controlPlane: ❷
  hyperthreading: Enabled ❸ ❹
  name: master
  platform:
    azure:
      encryptionAtHost: true
      ultraSSDCapability: Enabled
      osDisk:
        diskSizeGB: 1024 ❺
        diskType: Premium_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      type: Standard_D8s_v3
  replicas: 3
compute: ❻
- hyperthreading: Enabled ❼ ❽
  name: worker
  platform:
    azure:
      ultraSSDCapability: Enabled
      type: Standard_D2s_v3
      encryptionAtHost: true
      osDisk:
        diskSizeGB: 512 9
        diskType: Standard_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      zones: 10
      - "1"
      - "2"
      - "3"
  replicas: 5
metadata:
  name: test-cluster 11
networking: 12
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 13
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    defaultMachinePlatform:
      osImage: 14
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      ultraSSDCapability: Enabled
    baseDomainResourceGroupName: resource_group 15
    region: centralus 16
    resourceGroupName: existing_resource_group 17
    outboundType: Loadbalancer
    cloudName: AzurePublicCloud
pullSecret: '{"auths": ...}' 18
fips: false 19
sshKey: ssh-ed25519 AAAA... 20
```

1 11 16 18 Required. The installation program prompts you for this value.

2 6 12 If you do not provide these parameters and values, the installation program provides the default value.

3 7 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, **-**, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

4 8 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger virtual machine types, such as **Standard_D8s_v3**, for your machines if you disable simultaneous multithreading.

5 9 You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes is 1024 GB.

10 Specify a list of zones to deploy your machines to. For high availability, specify at least two zones.

13 The cluster network plugin to install. The default value **OVNKubernetes** is the only supported value.

14 Optional: A custom Red Hat Enterprise Linux CoreOS (RHCOS) image that should be used to boot control plane and compute machines. The **publisher**, **offer**, **sku**, and **version** parameters under **platform.azure.defaultMachinePlatform.osImage** apply to both control plane and compute machines. If the parameters under **controlPlane.platform.azure.osImage** or **compute.platform.azure.osImage** are set, they override the **platform.azure.defaultMachinePlatform.osImage** parameters.

15 Specify the name of the resource group that contains the DNS zone for your base domain.

17 Specify the name of an already existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.

19 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Switching RHEL to FIPS mode.

When running Red Hat Enterprise Linux (RHEL) or Red Hat Enterprise Linux CoreOS (RHCOS) booted in FIPS mode, OpenShift Container Platform core components use the RHEL cryptographic libraries that have been submitted to NIST for FIPS 140-2/140-3 Validation on only the x86_64, ppc64le, and s390x architectures.

20 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

3.4.1.7. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.

NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> ①
  httpsProxy: https://<username>:<pswd>@<ip>:<port> ②
  noProxy: example.com ③
additionalTrustBundle: | ④
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> ⑤
```

① A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.

② A proxy URL to use for creating HTTPS connections outside the cluster.

3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For

4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

NOTE

The installation program does not support the proxy **readinessEndpoints** field.

NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

3.4.2. Network configuration phases

There are two phases prior to OpenShift Container Platform installation where you can customize the network configuration.

Phase 1

You can customize the following network-related fields in the **install-config.yaml** file before you create the manifest files:

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**

- **networking.machineNetwork**
- **nodeNetworking**
  For more information, see "Installation configuration parameters".

NOTE

Set the **networking.machineNetwork** to match the Classless Inter-Domain Routing (CIDR) where the preferred subnet is located.

IMPORTANT

The CIDR range **172.17.0.0/16** is reserved by **libVirt**. You cannot use any other CIDR range that overlaps with the **172.17.0.0/16** CIDR range for networks in your cluster.

Phase 2

After creating the manifest files by running **openshift-install create manifests**, you can define a customized Cluster Network Operator manifest with only the fields you want to modify. You can use the manifest to specify an advanced network configuration.

During phase 2, you cannot override the values that you specified in phase 1 in the **install-config.yaml** file. However, you can customize the network plugin during phase 2.

3.4.3. Specifying advanced network configuration

You can use advanced network configuration for your network plugin to integrate your cluster into your existing network environment.

You can specify advanced network configuration only before you install the cluster.

IMPORTANT

Customizing your network configuration by modifying the OpenShift Container Platform manifest files created by the installation program is not supported. Applying a manifest file that you create, as in the following procedure, is supported.

Prerequisites

- You have created the **install-config.yaml** file and completed any modifications to it.

Procedure

1. Change to the directory that contains the installation program and create the manifests:

   ```
   $ ./openshift-install create manifests --dir <installation_directory> 1
   ```

   1 **<installation_directory>** specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yml** in the **<installation_directory>/manifests/** directory:

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

3. Specify the advanced network configuration for your cluster in the **cluster-network-03-config.yml** file, such as in the following example:

Enable IPsec for the OVN-Kubernetes network provider

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig:
        mode: Full
```

4. Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program consumes the **manifests/** directory when you create the Ignition config files.

5. Remove the Kubernetes manifest files that define the control plane machines and compute **MachineSets**:

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

Because you create and manage these resources yourself, you do not have to initialize them.

- You can preserve the **MachineSet** files to create compute machines by using the machine API, but you must update references to them to match your environment.

3.4.4. Cluster Network Operator configuration

The configuration for the cluster network is specified as part of the Cluster Network Operator (CNO) configuration and stored in a custom resource (CR) object that is named **cluster**. The CR specifies the fields for the **Network** API in the **operator.openshift.io** API group.

The CNO configuration inherits the following fields during cluster installation from the **Network** API in the **Network.config.openshift.io** API group:

clusterNetwork

IP address pools from which pod IP addresses are allocated.

serviceNetwork

IP address pool for services.

defaultNetwork.type

Cluster network plugin. **OVNKubernetes** is the only supported plugin during installation.

You can specify the cluster network plugin configuration for your cluster by setting the fields for the **defaultNetwork** object in the CNO object named **cluster**.

3.4.4.1. Cluster Network Operator configuration object

The fields for the Cluster Network Operator (CNO) are described in the following table:

Table 3.3. Cluster Network Operator configuration object

Field	Type	Description
metadata.name	**string**	The name of the CNO object. This name is always **cluster**.

spec.clusterNetwork	**array**	A list specifying the blocks of IP addresses from which pod IP addresses are allocated and the subnet prefix length assigned to each individual node in the cluster. For example:
		`spec:`
		`  clusterNetwork:`
		`  - cidr: 10.128.0.0/19`
		`    hostPrefix: 23`
		`  - cidr: 10.128.32.0/19`
		`    hostPrefix: 23`

spec.serviceNetwork	**array**	A block of IP addresses for services. The OVN-Kubernetes network plugin supports only a single IP address block for the service network. For example:
		`spec:`
		`  serviceNetwork:`
		`  - 172.30.0.0/14`
		You can customize this field only in the **install-config.yaml** file before you create the manifests. The value is read-only in the manifest file.

spec.defaultNetwork	**object**	Configures the network plugin for the cluster network.
spec.kubeProxyConfig	**object**	The fields for this object specify the kube-proxy configuration. If you are using the OVN-Kubernetes cluster network plugin, the kube-proxy configuration has no effect.

IMPORTANT

For a cluster that needs to deploy objects across multiple networks, ensure that you specify the same value for the **clusterNetwork.hostPrefix** parameter for each network type that is defined in the **install-config.yaml** file. Setting a different value for each **clusterNetwork.hostPrefix** parameter can impact the OVN-Kubernetes network plugin, where the plugin cannot effectively route object traffic among different nodes.

defaultNetwork object configuration

The values for the **defaultNetwork** object are defined in the following table:

Table 3.4. **defaultNetwork** object

Field	Type	Description
type	**string**	**OVNKubernetes**. The Red Hat OpenShift Networking network plugin is selected during installation. This value cannot be changed after cluster installation. NOTE OpenShift Container Platform uses the OVN-Kubernetes network plugin by default.
ovnKubernetesConfig	**object**	This object is only valid for the OVN-Kubernetes network plugin.

Configuration for the OVN-Kubernetes network plugin

The following table describes the configuration fields for the OVN-Kubernetes network plugin:

Table 3.5. **ovnKubernetesConfig** object

Field	Type	Description
mtu	**integer**	The maximum transmission unit (MTU) for the Geneve (Generic Network Virtualization Encapsulation) overlay network. This is detected automatically based on the MTU of the primary network interface. You do not normally need to override the detected MTU. If the auto-detected value is not what you expect it to be, confirm that the MTU on the primary network interface on your nodes is correct. You cannot use this option to change the MTU value of the primary network interface on the nodes. If your cluster requires different MTU values for different nodes, you must set this value to **100** less than the lowest MTU value in your cluster. For example, if some nodes in your cluster have an MTU of **9001**, and some have an MTU of**1500**, you must set this value to **1400**.
genevePort	**integer**	The port to use for all Geneve packets. The default value is **6081**. This value cannot be changed after cluster installation.
ipsecConfig	**object**	Specify a configuration object for customizing the IPsec configuration.
ipv4	**object**	Specifies a configuration object for IPv4 settings.
ipv6	**object**	Specifies a configuration object for IPv6 settings.

Field	Type	Description
policyAuditConfig	**object**	Specify a configuration object for customizing network policy audit logging. If unset, the defaults audit log settings are used.
gatewayConfig	**object**	Optional: Specify a configuration object for customizing how egress traffic is sent to the node gateway. Valid values are **Shared** and **Local**. The default value is **Shared**. In the default setting, the Open vSwitch (OVS) outputs traffic directly to the node IP interface. In the **Local** setting, it traverses the host network; consequently, it gets applied to the routing table of the host. NOTE While migrating egress traffic, you can expect some disruption to workloads and service traffic until the Cluster Network Operator (CNO) successfully rolls out the changes.

Table 3.6. **ovnKubernetesConfig.ipv4** object

Field	Type	Description
internalTransitSwitchSubnet	string	If your existing network infrastructure overlaps with the **100.88.0.0/16** IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. The subnet for the distributed transit switch that enables east-west traffic. This subnet cannot overlap with any other subnets used by OVN-Kubernetes or on the host itself. It must be large enough to accommodate one IP address per node in your cluster. The default value is **100.88.0.0/16**.
internalJoinSubnet	string	If your existing network infrastructure overlaps with the **100.64.0.0/16** IPv4 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. For example, if the **clusterNetwork.cidr** value is **10.128.0.0/14** and the **clusterNetwork.hostPrefix** value is **/23**, then the maximum number of nodes is **2^(23-14)=512**. The default value is **100.64.0.0/16**.

Table 3.7. **ovnKubernetesConfig.ipv6** object

Field	Type	Description
internalTransitSwitchSubnet	string	If your existing network infrastructure overlaps with the **fd97::/64** IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. The subnet for the distributed transit switch that enables east-west traffic. This subnet cannot overlap with any other subnets used by OVN-Kubernetes or on the host itself. It must be large enough to accommodate one IP address per node in your cluster. The default value is **fd97::/64**.
internalJoinSubnet	string	If your existing network infrastructure overlaps with the **fd98::/64** IPv6 subnet, you can specify a different IP address range for internal use by OVN-Kubernetes. You must ensure that the IP address range does not overlap with any other subnet used by your OpenShift Container Platform installation. The IP address range must be larger than the maximum number of nodes that can be added to the cluster. The default value is **fd98::/64**.

Table 3.8. **policyAuditConfig** object

Field	Type	Description
rateLimit	integer	The maximum number of messages to generate every second per node. The default value is **20** messages per second.
maxFileSize	integer	The maximum size for the audit log in bytes. The default value is **50000000** or 50 MB.
maxLogFiles	integer	The maximum number of log files that are retained.
destination	string	One of the following additional audit log targets: **libc** The libc **syslog()** function of the journald process on the host. **udp:<host>:<port>** A syslog server. Replace **<host>:<port>** with the host and port of the syslog server. **unix:<file>** A Unix Domain Socket file specified by **<file>**. **null** Do not send the audit logs to any additional target.
syslogFacility	string	The syslog facility, such as **kern**, as defined by RFC5424. The default value is **local0**.

Table 3.9. **gatewayConfig** object

Field	Type	Description
routingViaHost	**boolean**	Set this field to **true** to send egress traffic from pods to the host networking stack. For highly-specialized installations and applications that rely on manually configured routes in the kernel routing table, you might want to route egress traffic to the host networking stack. By default, egress traffic is processed in OVN to exit the cluster and is not affected by specialized routes in the kernel routing table. The default value is **false**. This field has an interaction with the Open vSwitch hardware offloading feature. If you set this field to **true**, you do not receive the performance benefits of the offloading because egress traffic is processed by the host networking stack.
ipForwarding	**object**	You can control IP forwarding for all traffic on OVN-Kubernetes managed interfaces by using the **ipForwarding** specification in the **Network** resource. Specify **Restricted** to only allow IP forwarding for Kubernetes related traffic. Specify **Global** to allow forwarding of all IP traffic. For new installations, the default is **Restricted**. For updates to OpenShift Container Platform 4.14 or later, the default is **Global**. NOTE The default value of **Restricted** sets the IP forwarding to drop.
ipv4	**object**	Optional: Specify an object to configure the internal OVN-Kubernetes masquerade address for host to service traffic for IPv4 addresses.
ipv6	**object**	Optional: Specify an object to configure the internal OVN-Kubernetes masquerade address for host to service traffic for IPv6 addresses.

Table 3.10. **gatewayConfig.ipv4** object

Field	Type	Description

Field	Type	Description
internalMasqueradeSubnet	**string**	The masquerade IPv4 addresses that are used internally to enable host to service traffic. The host is configured with these IP addresses as well as the shared gateway bridge interface. The default value is **169.254.169.0/29**. IMPORTANT For OpenShift Container Platform 4.17 and later versions, clusters use **169.254.0.0/17** as the default masquerade subnet. For upgraded clusters, there is no change to the default masquerade subnet.

Table 3.11. **gatewayConfig.ipv6** object

Field	Type	Description
internalMasqueradeSubnet	**string**	The masquerade IPv6 addresses that are used internally to enable host to service traffic. The host is configured with these IP addresses as well as the shared gateway bridge interface. The default value is **fd69::/125**. IMPORTANT For OpenShift Container Platform 4.17 and later versions, clusters use **fd69::/112** as the default masquerade subnet. For upgraded clusters, there is no change to the default masquerade subnet.

Table 3.12. **ipsecConfig** object

Field	Type	Description
mode	**string**	Specifies the behavior of the IPsec implementation. Must be one of the following values: - **Disabled**: IPsec is not enabled on cluster nodes. - **External**: IPsec is enabled for network traffic with external hosts. - **Full**: IPsec is enabled for pod traffic and network traffic with external hosts.

Field	Type	Description

Example OVN-Kubernetes configuration with IPSec enabled

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
    ipsecConfig:
      mode: Full
```

3.4.5. Configuring hybrid networking with OVN-Kubernetes

You can configure your cluster to use hybrid networking with the OVN-Kubernetes network plugin. This allows a hybrid cluster that supports different node networking configurations.

> **NOTE**
>
> This configuration is necessary to run both Linux and Windows nodes in the same cluster.

Prerequisites

- You defined **OVNKubernetes** for the **networking.networkType** parameter in the **install-config.yaml** file. See the installation documentation for configuring OpenShift Container Platform network customizations on your chosen cloud provider for more information.

Procedure

1. Change to the directory that contains the installation program and create the manifests:

   ```
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   where:

   <installation_directory>

   Specifies the name of the directory that contains the **install-config.yaml** file for your cluster.

2. Create a stub manifest file for the advanced network configuration that is named **cluster-network-03-config.yml** in the **<installation_directory>/manifests/** directory:

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
EOF
```

where:

<installation_directory>

Specifies the directory name that contains the **manifests/** directory for your cluster.

3. Open the **cluster-network-03-config.yml** file in an editor and configure OVN-Kubernetes with hybrid networking, as in the following example:

Specify a hybrid networking configuration

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      hybridOverlayConfig:
        hybridClusterNetwork: ①
        - cidr: 10.132.0.0/14
          hostPrefix: 23
        hybridOverlayVXLANPort: 9898 ②
```

① Specify the CIDR configuration used for nodes on the additional overlay network. The **hybridClusterNetwork** CIDR must not overlap with the **clusterNetwork** CIDR.

② Specify a custom VXLAN port for the additional overlay network. This is required for running Windows nodes in a cluster installed on vSphere, and must not be configured for any other cloud provider. The custom port can be any open port excluding the default **4789** port. For more information on this requirement, see the Microsoft documentation on Pod-to-pod connectivity between hosts is broken.

NOTE

Windows Server Long-Term Servicing Channel (LTSC): Windows Server 2019 is not supported on clusters with a custom **hybridOverlayVXLANPort** value because this Windows server version does not support selecting a custom VXLAN port.

4. Save the **cluster-network-03-config.yml** file and quit the text editor.

5. Optional: Back up the **manifests/cluster-network-03-config.yml** file. The installation program deletes the **manifests/** directory when creating the cluster.

NOTE

For more information about using Linux and Windows nodes in the same cluster, see Understanding Windows container workloads.

Additional resources

- For more details about Accelerated Networking, see Accelerated Networking for Microsoft Azure VMs.

3.4.6. Alternatives to storing administrator-level secrets in the kube-system project

By default, administrator secrets are stored in the **kube-system** project. If you configured the **credentialsMode** parameter in the **install-config.yaml** file to **Manual**, you must use one of the following alternatives:

- To manage long-term cloud credentials manually, follow the procedure in Manually creating long-term credentials.
- To implement short-term credentials that are managed outside the cluster for individual components, follow the procedures in Configuring an Azure cluster to use short-term credentials.

3.4.6.1. Manually creating long-term credentials

The Cloud Credential Operator (CCO) can be put into manual mode prior to installation in environments where the cloud identity and access management (IAM) APIs are not reachable, or the administrator prefers not to store an administrator-level credential secret in the cluster **kube-system** namespace.

Procedure

1. If you did not set the **credentialsMode** parameter in the **install-config.yaml** configuration file to **Manual**, modify the value as shown:

   Sample configuration file snippet

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. If you have not previously created installation manifest files, do so by running the following command:

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   where **<installation_directory>** is the directory in which the installation program creates files.

3. Set a **$RELEASE_IMAGE** variable with the release image from your installation file by running the following command:

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

4. Extract the list of **CredentialsRequest** custom resources (CRs) from the OpenShift Container Platform release image by running the following command:

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1 The **--included** parameter includes only the manifests that your specific cluster configuration requires.

2 Specify the location of the **install-config.yaml** file.

3 Specify the path to the directory where you want to store the **CredentialsRequest** objects. If the specified directory does not exist, this command creates it.

This command creates a YAML file for each **CredentialsRequest** object.

Sample CredentialsRequest object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
  ...
```

5. Create YAML files for secrets in the **openshift-install** manifests directory that you generated previously. The secrets must be stored using the namespace and secret name defined in the **spec.secretRef** for each **CredentialsRequest** object.

Sample CredentialsRequest object with secrets

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
    ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Sample Secret object

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  azure_subscription_id: <base64_encoded_azure_subscription_id>
  azure_client_id: <base64_encoded_azure_client_id>
  azure_client_secret: <base64_encoded_azure_client_secret>
  azure_tenant_id: <base64_encoded_azure_tenant_id>
  azure_resource_prefix: <base64_encoded_azure_resource_prefix>
  azure_resourcegroup: <base64_encoded_azure_resourcegroup>
  azure_region: <base64_encoded_azure_region>
```

IMPORTANT

Before upgrading a cluster that uses manually maintained credentials, you must ensure that the CCO is in an upgradeable state.

3.4.6.2. Configuring an Azure cluster to use short-term credentials

To install a cluster that uses Microsoft Entra Workload ID, you must configure the Cloud Credential Operator utility and create the required Azure resources for your cluster.

3.4.6.2.1. Configuring the Cloud Credential Operator utility

To create and manage cloud credentials from outside of the cluster when the Cloud Credential Operator (CCO) is operating in manual mode, extract and prepare the CCO utility (**ccoctl**) binary.

NOTE

The **ccoctl** utility is a Linux binary that must run in a Linux environment.

Prerequisites

- You have access to an OpenShift Container Platform account with cluster administrator access.
- You have installed the OpenShift CLI (**oc**).
- You have created a global Azure account for the **ccoctl** utility to use with the following permissions:
  - **Microsoft.Resources/subscriptions/resourceGroups/read**
  - **Microsoft.Resources/subscriptions/resourceGroups/write**

- **Microsoft.Resources/subscriptions/resourceGroups/delete**
- **Microsoft.Authorization/roleAssignments/read**
- **Microsoft.Authorization/roleAssignments/delete**
- **Microsoft.Authorization/roleAssignments/write**
- **Microsoft.Authorization/roleDefinitions/read**
- **Microsoft.Authorization/roleDefinitions/write**
- **Microsoft.Authorization/roleDefinitions/delete**
- **Microsoft.Storage/storageAccounts/listkeys/action**
- **Microsoft.Storage/storageAccounts/delete**
- **Microsoft.Storage/storageAccounts/read**
- **Microsoft.Storage/storageAccounts/write**
- **Microsoft.Storage/storageAccounts/blobServices/containers/delete**
- **Microsoft.Storage/storageAccounts/blobServices/containers/read**
- **Microsoft.Storage/storageAccounts/blobServices/containers/write**
- **Microsoft.ManagedIdentity/userAssignedIdentities/delete**
- **Microsoft.ManagedIdentity/userAssignedIdentities/read**
- **Microsoft.ManagedIdentity/userAssignedIdentities/write**
- **Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/read**
- **Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/write**
- **Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/delete**
- **Microsoft.Storage/register/action**
- **Microsoft.ManagedIdentity/register/action**

Procedure

1. Set a variable for the OpenShift Container Platform release image by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. Obtain the CCO container image from the OpenShift Container Platform release image by running the following command:

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

NOTE

Ensure that the architecture of the **$RELEASE_IMAGE** matches the architecture of the environment in which you will use the **ccoctl** tool.

3. Extract the **ccoctl** binary from the CCO container image within the OpenShift Container Platform release image by running the following command:

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 For **<rhel_version>**, specify the value that corresponds to the version of Red Hat Enterprise Linux (RHEL) that the host uses. If no value is specified, **ccoctl.rhel8** is used by default. The following values are valid:

- **rhel8**: Specify this value for hosts that use RHEL 8.
- **rhel9**: Specify this value for hosts that use RHEL 9.

4. Change the permissions to make **ccoctl** executable by running the following command:

```
$ chmod 775 ccoctl.<rhel_version>
```

Verification

- To verify that **ccoctl** is ready to use, display the help file. Use a relative file name when you run the command, for example:

```
$ ./ccoctl.rhel9
```

Example output

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws          Manage credentials objects for AWS cloud
  azure        Manage credentials objects for Azure
  gcp          Manage credentials objects for Google cloud
  help         Help about any command
  ibmcloud     Manage credentials objects for {ibm-cloud-title}
  nutanix      Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

3.4.6.2.2. Creating Azure resources with the Cloud Credential Operator utility

You can use the **ccoctl azure create-all** command to automate the creation of Azure resources.

NOTE

By default, **ccoctl** creates objects in the directory in which the commands are run. To create the objects in a different directory, use the **--output-dir** flag. This procedure uses **<path_to_ccoctl_output_dir>** to refer to this directory.

Prerequisites

You must have:

- Extracted and prepared the **ccoctl** binary.
- Access to your Microsoft Azure account by using the Azure CLI.

Procedure

1. Set a **$RELEASE_IMAGE** variable with the release image from your installation file by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. Extract the list of **CredentialsRequest** objects from the OpenShift Container Platform release image by running the following command:

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1 The **--included** parameter includes only the manifests that your specific cluster configuration requires.

2 Specify the location of the **install-config.yaml** file.

3 Specify the path to the directory where you want to store the **CredentialsRequest** objects. If the specified directory does not exist, this command creates it.

NOTE

This command might take a few moments to run.

3. To enable the **ccoctl** utility to detect your Azure credentials automatically, log in to the Azure CLI by running the following command:

```
$ az login
```

4. Use the **ccoctl** tool to process all **CredentialsRequest** objects by running the following command:

```
$ ccoctl azure create-all \
  --name=<azure_infra_name> \ 1
  --output-dir=<ccoctl_output_dir> \ 2
  --region=<azure_region> \ 3
  --subscription-id=<azure_subscription_id> \ 4
  --credentials-requests-dir=<path_to_credentials_requests_directory> \ 5
  --dnszone-resource-group-name=<azure_dns_zone_resource_group_name> \ 6
  --tenant-id=<azure_tenant_id> 7
```

1. Specify the user-defined name for all created Azure resources used for tracking.
2. Optional: Specify the directory in which you want the **ccoctl** utility to create objects. By default, the utility creates objects in the directory in which the commands are run.
3. Specify the Azure region in which cloud resources will be created.
4. Specify the Azure subscription ID to use.
5. Specify the directory containing the files for the component **CredentialsRequest** objects.
6. Specify the name of the resource group containing the cluster's base domain Azure DNS zone.
7. Specify the Azure tenant ID to use.

NOTE

If your cluster uses Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set, you must include the **--enable-tech-preview** parameter.

To see additional optional parameters and explanations of how to use them, run the **azure create-all --help** command.

Verification

- To verify that the OpenShift Container Platform secrets are created, list the files in the **<path_to_ccoctl_output_dir>/manifests** directory:

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

Example output

```
azure-ad-pod-identity-webhook-config.yaml
cluster-authentication-02-config.yaml
openshift-cloud-controller-manager-azure-cloud-credentials-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capz-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-azure-disk-credentials-credentials.yaml
openshift-cluster-csi-drivers-azure-file-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-azure-cloud-credentials-credentials.yaml
```

You can verify that the Microsoft Entra ID service accounts are created by querying Azure. For more information, refer to Azure documentation on listing Entra ID service accounts.

3.4.6.2.3. Incorporating the Cloud Credential Operator utility manifests

To implement short-term security credentials managed outside the cluster for individual components, you must move the manifest files that the Cloud Credential Operator utility (**ccoctl**) created to the correct directories for the installation program.

Prerequisites

- You have configured an account with the cloud platform that hosts your cluster.
- You have configured the Cloud Credential Operator utility (**ccoctl**).
- You have created the cloud provider resources that are required for your cluster with the **ccoctl** utility.

Procedure

1. If you did not set the **credentialsMode** parameter in the **install-config.yaml** configuration file to **Manual**, modify the value as shown:

   Sample configuration file snippet

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. If you used the **ccoctl** utility to create a new Azure resource group instead of using an existing resource group, modify the **resourceGroupName** parameter in the **install-config.yaml** as shown:

   Sample configuration file snippet

   ```
   apiVersion: v1
   baseDomain: example.com
   # ...
   platform:
     azure:
       resourceGroupName: <azure_infra_name> 1
   # ...
   ```

   1 This value must match the user-defined name for Azure resources that was specified with the **--name** argument of the **ccoctl azure create-all** command.

3. If you have not previously created installation manifest files, do so by running the following command:

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   where **<installation_directory>** is the directory in which the installation program creates files.

4. Copy the manifests that the **ccoctl** utility generated to the **manifests** directory that the installation program created by running the following command:

```
$ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
```

5. Copy the **tls** directory that contains the private key to the installation directory:

```
$ cp -a /<path_to_ccoctl_output_dir>/tls .
```

3.4.7. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- You have configured an account with the cloud platform that hosts your cluster.
- You have the OpenShift Container Platform installation program and the pull secret for your cluster.
- You have an Azure subscription ID and tenant ID.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

```
$ ./openshift-install create cluster --dir <installation_directory> \ ①
    --log-level=info ②
```

① For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.

② To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/.openshift_install.log**.

IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

> **IMPORTANT**
>
> - The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
>
> - It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

3.4.8. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.

- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
   ```

   1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```

Example output

```
system:admin
```

Additional resources

- See Accessing the web console for more details about accessing and understanding the OpenShift Container Platform web console.

3.4.9. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting .

3.5. INSTALLING A CLUSTER ON AZURE IN A DISCONNECTED ENVIRONMENT

In OpenShift Container Platform version 4.18, you can install a cluster on Microsoft Azure in a restricted network by creating an internal mirror of the installation release content on an existing Azure Virtual Network (VNet).

IMPORTANT

You can install an OpenShift Container Platform cluster by using mirrored installation release content, but your cluster requires internet access to use the Azure APIs.

3.5.1. Prerequisites

- You mirrored the images for a disconnected installation to your registry and obtained the **imageContentSources** data for your version of OpenShift Container Platform.

  IMPORTANT

  Because the installation media is on the mirror host, you can use that computer to complete all installation steps.

- You have an existing VNet in Azure. While installing a cluster in a restricted network that uses installer-provisioned infrastructure, you cannot use the installer-provisioned VNet. You must use a user-provisioned VNet that satisfies one of the following requirements:
  - The VNet contains the mirror registry.
  - The VNet has firewall rules or a peering connection to access the mirror registry hosted elsewhere.

3.5.2. About installations in restricted networks

In OpenShift Container Platform 4.18, you can perform an installation that does not require an active connection to the internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service's Route 53 DNS and IAM services, require internet access. Depending on your network, you might require less internet access for an installation on bare metal hardware, Nutanix, or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift image registry and contains the installation media. You can create this registry on a mirror host, which can access both the internet and your closed network, or by using other methods that meet your restrictions.

3.5.2.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.
- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

3.5.2.2. User-defined outbound routing

In OpenShift Container Platform, you can choose your own outbound routing for a cluster to connect to the internet. This allows you to skip the creation of public IP addresses and the public load balancer.

You can configure user-defined routing by modifying parameters in the **install-config.yaml** file before installing your cluster. A pre-existing VNet is required to use outbound routing when installing a cluster; the installation program is not responsible for configuring this.

When configuring a cluster to use user-defined routing, the installation program does not create the following resources:

- Outbound rules for access to the internet.
- Public IPs for the public load balancer.
- Kubernetes Service object to add the cluster machines to the public load balancer for outbound requests.

You must ensure the following items are available before setting user-defined routing:

- Egress to the internet is possible to pull container images, unless using an OpenShift image registry mirror.
- The cluster can access Azure APIs.
- Various allowlist endpoints are configured. You can reference these endpoints in the *Configuring your firewall* section.

There are several pre-existing networking setups that are supported for internet access using user-defined routing.

Restricted cluster with Azure Firewall
You can use Azure Firewall to restrict the outbound routing for the Virtual Network (VNet) that is used to install the OpenShift Container Platform cluster. For more information, see providing user-defined routing with Azure Firewall. You can create a OpenShift Container Platform cluster in a restricted network by using VNet with Azure Firewall and configuring the user-defined routing.

IMPORTANT

If you are using Azure Firewall for restricting internet access, you must set the **publish** field to **Internal** in the **install-config.yaml** file. This is because Azure Firewall does not work properly with Azure public load balancers.

3.5.3. About reusing a VNet for your OpenShift Container Platform cluster

In OpenShift Container Platform 4.18, you can deploy a cluster into an existing Azure Virtual Network (VNet) in Microsoft Azure. If you do, you must also use existing subnets within the VNet and routing rules.

By deploying OpenShift Container Platform into an existing Azure VNet, you might be able to avoid service limit constraints in new accounts or more easily abide by the operational constraints that your company's guidelines set. This is a good option to use if you cannot obtain the infrastructure creation permissions that are required to create the VNet.

3.5.3.1. Requirements for using your VNet

When you deploy a cluster by using an existing VNet, you must perform additional network configuration before you install the cluster. In installer-provisioned infrastructure clusters, the installer usually creates the following components, but it does not create them when you install into an existing VNet:

- Subnets
- Route tables
- VNets
- Network Security Groups

NOTE

The installation program requires that you use the cloud-provided DNS server. Using a custom DNS server is not supported and causes the installation to fail.

If you use a custom VNet, you must correctly configure it and its subnets for the installation program and the cluster to use. The installation program cannot subdivide network ranges for the cluster to use, set route tables for the subnets, or set VNet options like DHCP, so you must do so before you install the cluster.

The cluster must be able to access the resource group that contains the existing VNet and subnets. While all of the resources that the cluster creates are placed in a separate resource group that it creates, some network resources are used from a separate group. Some cluster Operators must be able to access resources in both resource groups. For example, the Machine API controller attaches NICS for the virtual machines that it creates to subnets from the networking resource group.

Your VNet must meet the following characteristics:

- The VNet's CIDR block must contain the **Networking.MachineCIDR** range, which is the IP address pool for cluster machines.
- The VNet and its subnets must belong to the same resource group, and the subnets must be configured to use Azure-assigned DHCP IP addresses instead of static IP addresses.

You must provide two subnets within your VNet, one for the control plane machines and one for the compute machines. Because Azure distributes machines in different availability zones within the region that you specify, your cluster will have high availability by default.

NOTE

By default, if you specify availability zones in the **install-config.yaml** file, the installation program distributes the control plane machines and the compute machines across these availability zones within a region. To ensure high availability for your cluster, select a region with at least three availability zones. If your region contains fewer than three availability zones, the installation program places more than one control plane machine in the available zones.

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the specified subnets exist.
- There are two private subnets, one for the control plane machines and one for the compute machines.
- The subnet CIDRs belong to the machine CIDR that you specified. Machines are not provisioned in availability zones that you do not provide private subnets for. If required, the installation program creates public load balancers that manage the control plane and worker nodes, and Azure allocates a public IP address to them.

NOTE

If you destroy a cluster that uses an existing VNet, the VNet is not deleted.

3.5.3.1.1. Network security group requirements

The network security groups for the subnets that host the compute and control plane machines require specific access to ensure that the cluster communication is correct. You must create rules to allow access to the required cluster communication ports.

IMPORTANT

The network security group rules must be in place before you install the cluster. If you attempt to install a cluster without the required access, the installation program cannot reach the Azure APIs, and installation fails.

Table 3.13. Required ports

Port	Description	Control plane	Compute
80	Allows HTTP traffic		x
443	Allows HTTPS traffic		x
6443	Allows communication to the control plane machines	x	

Port	Description	Control plane	Compute
22623	Allows internal communication to the machine config server for provisioning machines	x	
*	Allows connections to Azure APIs. You must set a Destination Service Tag to **AzureCloud**. [1]	x	x
*	Denies connections to the internet. You must set a Destination Service Tag to **Internet**. [1]	x	x

1. If you are using Azure Firewall to restrict the internet access, then you can configure Azure Firewall to allow the Azure APIs. A network security group rule is not needed. For more information, see "Configuring your firewall" in "Additional resources".

IMPORTANT

Currently, there is no supported way to block or restrict the machine config server endpoint. The machine config server must be exposed to the network so that newly-provisioned machines, which have no existing configuration or state, are able to fetch their configuration. In this model, the root of trust is the certificate signing requests (CSR) endpoint, which is where the kubelet sends its certificate signing request for approval to join the cluster. Because of this, machine configs should not be used to distribute sensitive information, such as secrets and certificates.

To ensure that the machine config server endpoints, ports 22623 and 22624, are secured in bare metal scenarios, customers must configure proper network policies.

Because cluster components do not modify the user-provided network security groups, which the Kubernetes controllers update, a pseudo-network security group is created for the Kubernetes controller to modify without impacting the rest of the environment.

Table 3.14. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	**1936**	Metrics
	9000-**9999**	Host level services, including the node exporter on ports **9100**-**9101** and the Cluster Version Operator on port**9099**.
	10250-**10259**	The default ports that Kubernetes reserves
UDP	**4789**	VXLAN
	6081	Geneve

Protocol	Port	Description
	9000-9999	Host level services, including the node exporter on ports **9100-9101**.
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port **123** If you configure an external NTP time server, you must open UDP port **123**.
TCP/UDP	**30000-32767**	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 3.15. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	**2379-2380**	etcd server and peer ports

3.5.3.2. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resources in your clouds than others. For example, you might be able to create application-specific items, like instances, storage, and load balancers, but not networking-related components such as VNets, subnet, or ingress rules.

The Azure credentials that you use when you create your cluster do not need the networking permissions that are required to make VNets and core networking components within the VNet, such as subnets, routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as load balancers, security groups, storage accounts, and nodes.

3.5.3.3. Isolation between clusters

Because the cluster is unable to modify network security groups in an existing subnet, there is no way to isolate clusters from each other on the VNet.

Additional resources

- About the OVN-Kubernetes network plugin
- Configuring your firewall

3.5.4. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Microsoft Azure.

Prerequisites

- You have the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.
- You have the **imageContentSources** values that were generated during mirror registry creation.
- You have obtained the contents of the certificate for your mirror registry.
- You have retrieved a Red Hat Enterprise Linux CoreOS (RHCOS) image and uploaded it to an accessible location.
- You have an Azure subscription ID and tenant ID.
- If you are installing the cluster using a service principal, you have its application ID and password.
- If you are installing the cluster using a system-assigned managed identity, you have enabled it on the virtual machine that you will run the installation program from.
- If you are installing the cluster using a user-assigned managed identity, you have met these prerequisites:
  - You have its client ID.
  - You have assigned it to the virtual machine that you will run the installation program from.

Procedure

1. Optional: If you have run the installation program on this computer before, and want to use an alternative service principal or managed identity, go to the **~/.azure/** directory and delete the **osServicePrincipal.json** configuration file.
   Deleting this file prevents the installation program from automatically reusing subscription and authentication values from a previous installation.

2. Create the **install-config.yaml** file.

   a. Change to the directory that contains the installation program and run the following command:

   ```
   $ ./openshift-install create install-config --dir <installation_directory> 1
   ```

   1 For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

   When specifying the directory:

   - Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
   - Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If

you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

ii. Select **azure** as the platform to target.
If the installation program cannot locate the **osServicePrincipal.json** configuration file from a previous installation, you are prompted for Azure subscription and authentication values.

iii. Enter the following Azure parameter values for your subscription:

- **azure subscription id** Enter the subscription ID to use for the cluster.
- **azure tenant id** Enter the tenant ID.

iv. Depending on the Azure identity you are using to deploy the cluster, do one of the following when prompted for the **azure service principal client id**

- If you are using a service principal, enter its application ID.
- If you are using a system-assigned managed identity, leave this value blank.
- If you are using a user-assigned managed identity, specify its client ID.

v. Depending on the Azure identity you are using to deploy the cluster, do one of the following when prompted for the **azure service principal client secret**

- If you are using a service principal, enter its password.
- If you are using a system-assigned managed identity, leave this value blank.
- If you are using a user-assigned managed identity, leave this value blank.

vi. Select the region to deploy the cluster to.

vii. Select the base domain to deploy the cluster to. The base domain corresponds to the Azure DNS Zone that you created for your cluster.

viii. Enter a descriptive name for your cluster.

IMPORTANT

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see Resolve reserved resource name errors in the Azure documentation.

ix. Paste the pull secret from Red Hat OpenShift Cluster Manager .

3. Edit the **install-config.yaml** file to give the additional information that is required for an installation in a restricted network.

   a. Update the **pullSecret** value to contain the authentication information for your registry:

   ```
   pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email":
   "you@example.com"}}}'
   ```

   For **<mirror_host_name>**, specify the registry domain name that you specified in the certificate for your mirror registry, and for **<credentials>**, specify the base64-encoded user name and password for your mirror registry.

   b. Add the **additionalTrustBundle** parameter and value.

   ```
   additionalTrustBundle: |
     -----BEGIN CERTIFICATE-----
   
   ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
     -----END CERTIFICATE-----
   ```

   The value must be the contents of the certificate file that you used for your mirror registry. The certificate file can be an existing, trusted certificate authority, or the self-signed certificate that you generated for the mirror registry.

   c. Define the network and subnets for the VNet to install the cluster under the **platform.azure** field:

   ```
   networkResourceGroupName: <vnet_resource_group> 1
   virtualNetwork: <vnet> 2
   controlPlaneSubnet: <control_plane_subnet> 3
   computeSubnet: <compute_subnet> 4
   ```

   1 Replace **<vnet_resource_group>** with the resource group name that contains the existing virtual network (VNet).

   2 Replace **<vnet>** with the existing virtual network name.

   3 Replace **<control_plane_subnet>** with the existing subnet name to deploy the control plane machines.

   4 Replace **<compute_subnet>** with the existing subnet name to deploy compute machines.

   d. Add the image content resources, which resemble the following YAML excerpt:

   ```
   imageContentSources:
   - mirrors:
     - <mirror_host_name>:5000/<repo_name>/release
     source: quay.io/openshift-release-dev/ocp-release
   - mirrors:
     - <mirror_host_name>:5000/<repo_name>/release
     source: registry.redhat.io/ocp/release
   ```

For these values, use the **imageContentSources** that you recorded during mirror registry creation.

e. Optional: Set the publishing strategy to **Internal**:

```
publish: Internal
```

By setting this option, you create an internal Ingress Controller and a private load balancer.

IMPORTANT

Azure Firewall does not work seamlessly with Azure Public Load balancers. Thus, when using Azure Firewall for restricting internet access, the **publish** field in **install-config.yaml** should be set to **Internal**.

4. Make any other modifications to the **install-config.yaml** file that you require. For more information about the parameters, see "Installation configuration parameters".

5. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

If previously not detected, the installation program creates an **osServicePrincipal.json** configuration file and stores this file in the ~/**.azure**/ directory on your computer. This ensures that the installation program can load the profile when it is creating an OpenShift Container Platform cluster on the target platform.

Additional resources

- Installation configuration parameters for Azure

3.5.4.1. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 3.16. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or Hyper-Threading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

NOTE

For OpenShift Container Platform version 4.18, RHCOS is based on RHEL version 9.4, which updates the micro-architecture requirements. The following list contains the minimum instruction set architectures (ISA) that each architecture requires:

- x86-64 architecture requires x86-64-v2 ISA
- ARM64 architecture requires ARMv8.0-A ISA
- IBM Power architecture requires Power 9 ISA
- s390x architecture requires z14 ISA

For more information, see Architectures (RHEL documentation).

IMPORTANT

You are required to use Azure virtual machines that have the **premiumIO** parameter set to **true**.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

3.5.4.2. Tested instance types for Azure

The following Microsoft Azure instance types have been tested with OpenShift Container Platform.

Example 3.5. Machine types based on 64-bit x86 architecture

- **standardBasv2Family**
- **standardBSFamily**
- **standardBsv2Family**
- **standardDADSv5Family**
- **standardDASv4Family**

- standardDASv5Family
- standardDCACCV5Family
- standardDCADCCV5Family
- standardDCADSv5Family
- standardDCASv5Family
- standardDCSv3Family
- standardDCSv2Family
- standardDDCSv3Family
- standardDDSv4Family
- standardDDSv5Family
- standardDLDSv5Family
- standardDLSv5Family
- standardDSFamily
- standardDSv2Family
- standardDSv2PromoFamily
- standardDSv3Family
- standardDSv4Family
- standardDSv5Family
- standardEADSv5Family
- standardEASv4Family
- standardEASv5Family
- standardEBDSv5Family
- standardEBSv5Family
- standardECACCV5Family
- standardECADCCV5Family
- standardECADSv5Family
- standardECASv5Family
- standardEDSv4Family
- standardEDSv5Family

- **standardEIADSv5Family**
- **standardEIASv4Family**
- **standardEIASv5Family**
- **standardEIBDSv5Family**
- **standardEIBSv5Family**
- **standardEIDSv5Family**
- **standardEISv3Family**
- **standardEISv5Family**
- **standardESv3Family**
- **standardESv4Family**
- **standardESv5Family**
- **standardFXMDVSFamily**
- **standardFSFamily**
- **standardFSv2Family**
- **standardGSFamily**
- **standardHBrsv2Family**
- **standardHBSFamily**
- **standardHBv4Family**
- **standardHCSFamily**
- **standardHXFamily**
- **standardLASv3Family**
- **standardLSFamily**
- **standardLSv2Family**
- **standardLSv3Family**
- **standardMDSHighMemoryv3Family**
- **standardMDSMediumMemoryv2Family**
- **standardMDSMediumMemoryv3Family**
- **standardMIDSHighMemoryv3Family**
- **standardMIDSMediumMemoryv2Family**

- **standardMISHighMemoryv3Family**
- **standardMISMediumMemoryv2Family**
- **standardMSFamily**
- **standardMSHighMemoryv3Family**
- **standardMSMediumMemoryv2Family**
- **standardMSMediumMemoryv3Family**
- **StandardNCADSA100v4Family**
- **Standard NCASv3_T4 Family**
- **standardNCSv3Family**
- **standardNDSv2Family**
- **StandardNGADSV620v1Family**
- **standardNPSFamily**
- **StandardNVADSA10v5Family**
- **standardNVSv3Family**
- **standardXEISv4Family**

3.5.4.3. Tested instance types for Azure on 64-bit ARM infrastructures

The following Microsoft Azure ARM64 instance types have been tested with OpenShift Container Platform.

Example 3.6. Machine types based on 64-bit ARM architecture

- **standardBpsv2Family**
- **standardDPSv5Family**
- **standardDPDSv5Family**
- **standardDPLDSv5Family**
- **standardDPLSv5Family**
- **standardEPSv5Family**
- **standardEPDSv5Family**
- **StandardDpdsv6Family**
- **StandardDpldsv6Famil**
- **StandardDplsv6Family**

- **StandardDpsv6Family**
- **StandardEpdsv6Family**
- **StandardEpsv6Family**

3.5.4.4. Enabling trusted launch for Azure VMs

You can enable two trusted launch features when installing your cluster on Azure: secure boot and virtualized Trusted Platform Modules.

For more information about the sizes of virtual machines that support the trusted launch features, see Virtual machine sizes.

IMPORTANT

Trusted launch is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see Technology Preview Features Support Scope .

Prerequisites

- You have created an **install-config.yaml** file.

Procedure

- Edit the **install-config.yaml** file before deploying your cluster:
  - Enable trusted launch only on control plane by adding the following stanza:

```
controlPlane:
  platform:
    azure:
      settings:
        securityType: TrustedLaunch
        trustedLaunch:
          uefiSettings:
            secureBoot: Enabled
            virtualizedTrustedPlatformModule: Enabled
```

  - Enable trusted launch only on compute node by adding the following stanza:

```
compute:
  platform:
    azure:
      settings:
        securityType: TrustedLaunch
        trustedLaunch:
          uefiSettings:
            secureBoot: Enabled
            virtualizedTrustedPlatformModule: Enabled
```

- Enable trusted launch on all nodes by adding the following stanza:

```
platform:
  azure:
    settings:
      securityType: TrustedLaunch
      trustedLaunch:
        uefiSettings:
          secureBoot: Enabled
          virtualizedTrustedPlatformModule: Enabled
```

3.5.4.5. Enabling confidential VMs

You can enable confidential VMs when installing your cluster. You can enable confidential VMs for compute nodes, control plane nodes, or all nodes.

IMPORTANT

Using confidential VMs is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see Technology Preview Features Support Scope .

You can use confidential VMs with the following VM sizes:

- DCasv5-series
- DCadsv5-series
- ECasv5-series
- ECadsv5-series

IMPORTANT

Confidential VMs are currently not supported on 64-bit ARM architectures.

Prerequisites

- You have created an **install-config.yaml** file.

Procedure

- Edit the **install-config.yaml** file before deploying your cluster:
  - Enable confidential VMs only on control plane by adding the following stanza:

```
controlPlane:
  platform:
    azure:
      settings:
        securityType: ConfidentialVM
        confidentialVM:
          uefiSettings:
            secureBoot: Enabled
            virtualizedTrustedPlatformModule: Enabled
      osDisk:
        securityProfile:
          securityEncryptionType: VMGuestStateOnly
```

- Enable confidential VMs only on compute nodes by adding the following stanza:

```
compute:
  platform:
    azure:
      settings:
        securityType: ConfidentialVM
        confidentialVM:
          uefiSettings:
            secureBoot: Enabled
            virtualizedTrustedPlatformModule: Enabled
      osDisk:
        securityProfile:
          securityEncryptionType: VMGuestStateOnly
```

- Enable confidential VMs on all nodes by adding the following stanza:

```
platform:
  azure:
    settings:
      securityType: ConfidentialVM
      confidentialVM:
        uefiSettings:
          secureBoot: Enabled
          virtualizedTrustedPlatformModule: Enabled
    osDisk:
      securityProfile:
        securityEncryptionType: VMGuestStateOnly
```

3.5.4.6. Sample customized install-config.yaml file for Azure

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```
apiVersion: v1
```

```
baseDomain: example.com 1
controlPlane: 2
  hyperthreading: Enabled 3 4
  name: master
  platform:
    azure:
      encryptionAtHost: true
      ultraSSDCapability: Enabled
      osDisk:
        diskSizeGB: 1024 5
        diskType: Premium_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      type: Standard_D8s_v3
  replicas: 3
compute: 6
- hyperthreading: Enabled 7 8
  name: worker
  platform:
    azure:
      ultraSSDCapability: Enabled
      type: Standard_D2s_v3
      encryptionAtHost: true
      osDisk:
        diskSizeGB: 512 9
        diskType: Standard_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      zones: 10
      - "1"
      - "2"
      - "3"
  replicas: 5
metadata:
  name: test-cluster 11
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 12
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    defaultMachinePlatform:
      osImage: 13
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      ultraSSDCapability: Enabled
    baseDomainResourceGroupName: resource_group 14
    region: centralus 15
    resourceGroupName: existing_resource_group 16
    networkResourceGroupName: vnet_resource_group 17
    virtualNetwork: vnet 18
    controlPlaneSubnet: control_plane_subnet 19
    computeSubnet: compute_subnet 20
    outboundType: UserDefinedRouting 21
    cloudName: AzurePublicCloud
pullSecret: '{"auths": ...}' 22
fips: false 23
sshKey: ssh-ed25519 AAAA... 24
additionalTrustBundle: | 25
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
imageContentSources: 26
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
publish: Internal 27
```

1 11 15 22 Required. The installation program prompts you for this value.

2 6 If you do not provide these parameters and values, the installation program provides the default value.

3 7 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, **-**, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

4 8 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger virtual machine types, such as **Standard_D8s_v3**, for your machines if you disable simultaneous multithreading.

5 9 You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes is 1024 GB.

10 Specify a list of zones to deploy your machines to. For high availability, specify at least two zones.

12 The cluster network plugin to install. The default value **OVNKubernetes** is the only supported value.

13 Optional: A custom Red Hat Enterprise Linux CoreOS (RHCOS) image that should be used to boot control plane and compute machines. The **publisher**, **offer**, **sku**, and **version** parameters under **platform.azure.defaultMachinePlatform.osImage** apply to both control plane and compute machines. If the parameters under **controlPlane.platform.azure.osImage** or **compute.platform.azure.osImage** are set, they override the **platform.azure.defaultMachinePlatform.osImage** parameters.

14 Specify the name of the resource group that contains the DNS zone for your base domain.

16 Specify the name of an already existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.

17 If you use an existing VNet, specify the name of the resource group that contains it.

18 If you use an existing VNet, specify its name.

19 If you use an existing VNet, specify the name of the subnet to host the control plane machines.

20 If you use an existing VNet, specify the name of the subnet to host the compute machines.

21 When using Azure Firewall to restrict Internet access, you must configure outbound routing to send traffic through the Azure Firewall. Configuring user-defined routing prevents exposing external endpoints in your cluster.

23 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Switching RHEL to FIPS mode.

When running Red Hat Enterprise Linux (RHEL) or Red Hat Enterprise Linux CoreOS (RHCOS) booted in FIPS mode, OpenShift Container Platform core components use the RHEL cryptographic libraries that have been submitted to NIST for FIPS 140-2/140-3 Validation on only the x86_64, ppc64le, and s390x architectures.

24 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.

> NOTE
>
> For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

25 Provide the contents of the certificate file that you used for your mirror registry.

26 Provide the **imageContentSources** section from the output of the command to mirror the repository.

27 How to publish the user-facing endpoints of your cluster. When using Azure Firewall to restrict Internet access, set **publish** to **Internal** to deploy a private cluster. The user-facing endpoints then cannot be accessed from the internet. The default value is **External**.

3.5.4.7. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.

> NOTE
>
> The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.
>
> For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
```

```
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.

2. A proxy URL to use for creating HTTPS connections outside the cluster.

3. A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use * to bypass the proxy for all destinations.

4. If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

5. Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

NOTE

The installation program does not support the proxy **readinessEndpoints** field.

NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

3.5.5. Alternatives to storing administrator-level secrets in the kube-system project

By default, administrator secrets are stored in the **kube-system** project. If you configured the **credentialsMode** parameter in the **install-config.yaml** file to **Manual**, you must use one of the following alternatives:

- To manage long-term cloud credentials manually, follow the procedure in Manually creating long-term credentials.
- To implement short-term credentials that are managed outside the cluster for individual components, follow the procedures in Configuring an Azure cluster to use short-term credentials.

3.5.5.1. Manually creating long-term credentials

The Cloud Credential Operator (CCO) can be put into manual mode prior to installation in environments where the cloud identity and access management (IAM) APIs are not reachable, or the administrator prefers not to store an administrator-level credential secret in the cluster **kube-system** namespace.

Procedure

1. If you did not set the **credentialsMode** parameter in the **install-config.yaml** configuration file to **Manual**, modify the value as shown:

   Sample configuration file snippet

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. If you have not previously created installation manifest files, do so by running the following command:

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   where **<installation_directory>** is the directory in which the installation program creates files.

3. Set a **$RELEASE_IMAGE** variable with the release image from your installation file by running the following command:

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

4. Extract the list of **CredentialsRequest** custom resources (CRs) from the OpenShift Container Platform release image by running the following command:

   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
     --credentials-requests \
     --included \ ①
     --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ ②
     --to=<path_to_directory_for_credentials_requests> ③
   ```

1 The **--included** parameter includes only the manifests that your specific cluster configuration requires.

2 Specify the location of the **install-config.yaml** file.

3 Specify the path to the directory where you want to store the **CredentialsRequest** objects. If the specified directory does not exist, this command creates it.

This command creates a YAML file for each **CredentialsRequest** object.

Sample CredentialsRequest object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
  ...
```

5. Create YAML files for secrets in the **openshift-install** manifests directory that you generated previously. The secrets must be stored using the namespace and secret name defined in the **spec.secretRef** for each **CredentialsRequest** object.

Sample CredentialsRequest object with secrets

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Sample Secret object

```
apiVersion: v1
kind: Secret
```

```
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  azure_subscription_id: <base64_encoded_azure_subscription_id>
  azure_client_id: <base64_encoded_azure_client_id>
  azure_client_secret: <base64_encoded_azure_client_secret>
  azure_tenant_id: <base64_encoded_azure_tenant_id>
  azure_resource_prefix: <base64_encoded_azure_resource_prefix>
  azure_resourcegroup: <base64_encoded_azure_resourcegroup>
  azure_region: <base64_encoded_azure_region>
```

IMPORTANT

Before upgrading a cluster that uses manually maintained credentials, you must ensure that the CCO is in an upgradeable state.

3.5.5.2. Configuring an Azure cluster to use short-term credentials

To install a cluster that uses Microsoft Entra Workload ID, you must configure the Cloud Credential Operator utility and create the required Azure resources for your cluster.

3.5.5.2.1. Configuring the Cloud Credential Operator utility

To create and manage cloud credentials from outside of the cluster when the Cloud Credential Operator (CCO) is operating in manual mode, extract and prepare the CCO utility (**ccoctl**) binary.

NOTE

The **ccoctl** utility is a Linux binary that must run in a Linux environment.

Prerequisites

- You have access to an OpenShift Container Platform account with cluster administrator access.
- You have installed the OpenShift CLI (**oc**).
- You have created a global Azure account for the **ccoctl** utility to use with the following permissions:
  - **Microsoft.Resources/subscriptions/resourceGroups/read**
  - **Microsoft.Resources/subscriptions/resourceGroups/write**
  - **Microsoft.Resources/subscriptions/resourceGroups/delete**
  - **Microsoft.Authorization/roleAssignments/read**
  - **Microsoft.Authorization/roleAssignments/delete**
  - **Microsoft.Authorization/roleAssignments/write**
  - **Microsoft.Authorization/roleDefinitions/read**
  - **Microsoft.Authorization/roleDefinitions/write**

- **Microsoft.Authorization/roleDefinitions/delete**
- **Microsoft.Storage/storageAccounts/listkeys/action**
- **Microsoft.Storage/storageAccounts/delete**
- **Microsoft.Storage/storageAccounts/read**
- **Microsoft.Storage/storageAccounts/write**
- **Microsoft.Storage/storageAccounts/blobServices/containers/delete**
- **Microsoft.Storage/storageAccounts/blobServices/containers/read**
- **Microsoft.Storage/storageAccounts/blobServices/containers/write**
- **Microsoft.ManagedIdentity/userAssignedIdentities/delete**
- **Microsoft.ManagedIdentity/userAssignedIdentities/read**
- **Microsoft.ManagedIdentity/userAssignedIdentities/write**
- **Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/read**
- **Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/write**
- **Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/delete**
- **Microsoft.Storage/register/action**
- **Microsoft.ManagedIdentity/register/action**

Procedure

1. Set a variable for the OpenShift Container Platform release image by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. Obtain the CCO container image from the OpenShift Container Platform release image by running the following command:

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

> NOTE
>
> Ensure that the architecture of the **$RELEASE_IMAGE** matches the architecture of the environment in which you will use the **ccoctl** tool.

3. Extract the **ccoctl** binary from the CCO container image within the OpenShift Container Platform release image by running the following command:

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 For **<rhel_version>**, specify the value that corresponds to the version of Red Hat Enterprise Linux (RHEL) that the host uses. If no value is specified, **ccoctl.rhel8** is used by default. The following values are valid:

- **rhel8**: Specify this value for hosts that use RHEL 8.
- **rhel9**: Specify this value for hosts that use RHEL 9.

4. Change the permissions to make **ccoctl** executable by running the following command:

```
$ chmod 775 ccoctl.<rhel_version>
```

Verification

- To verify that **ccoctl** is ready to use, display the help file. Use a relative file name when you run the command, for example:

```
$ ./ccoctl.rhel9
```

Example output

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws          Manage credentials objects for AWS cloud
  azure        Manage credentials objects for Azure
  gcp          Manage credentials objects for Google cloud
  help         Help about any command
  ibmcloud     Manage credentials objects for {ibm-cloud-title}
  nutanix      Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

3.5.5.2.2. Creating Azure resources with the Cloud Credential Operator utility

You can use the **ccoctl azure create-all** command to automate the creation of Azure resources.

NOTE

By default, **ccoctl** creates objects in the directory in which the commands are run. To create the objects in a different directory, use the **--output-dir** flag. This procedure uses **<path_to_ccoctl_output_dir>** to refer to this directory.

Prerequisites

You must have:

- Extracted and prepared the **ccoctl** binary.
- Access to your Microsoft Azure account by using the Azure CLI.

Procedure

1. Set a **$RELEASE_IMAGE** variable with the release image from your installation file by running the following command:

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

2. Extract the list of **CredentialsRequest** objects from the OpenShift Container Platform release image by running the following command:

   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
     --credentials-requests \
     --included \ 1
     --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
     --to=<path_to_directory_for_credentials_requests> 3
   ```

   1 The **--included** parameter includes only the manifests that your specific cluster configuration requires.

   2 Specify the location of the **install-config.yaml** file.

   3 Specify the path to the directory where you want to store the **CredentialsRequest** objects. If the specified directory does not exist, this command creates it.

   NOTE

   This command might take a few moments to run.

3. To enable the **ccoctl** utility to detect your Azure credentials automatically, log in to the Azure CLI by running the following command:

   ```
   $ az login
   ```

4. Use the **ccoctl** tool to process all **CredentialsRequest** objects by running the following command:

   ```
   $ ccoctl azure create-all \
     --name=<azure_infra_name> \ 1
     --output-dir=<ccoctl_output_dir> \ 2
     --region=<azure_region> \ 3
     --subscription-id=<azure_subscription_id> \ 4
   ```

```
--credentials-requests-dir=<path_to_credentials_requests_directory> \ 5
--dnszone-resource-group-name=<azure_dns_zone_resource_group_name> \ 6
--tenant-id=<azure_tenant_id> 7
```

1 Specify the user-defined name for all created Azure resources used for tracking.

2 Optional: Specify the directory in which you want the **ccoctl** utility to create objects. By default, the utility creates objects in the directory in which the commands are run.

3 Specify the Azure region in which cloud resources will be created.

4 Specify the Azure subscription ID to use.

5 Specify the directory containing the files for the component **CredentialsRequest** objects.

6 Specify the name of the resource group containing the cluster's base domain Azure DNS zone.

7 Specify the Azure tenant ID to use.

NOTE

If your cluster uses Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set, you must include the **--enable-tech-preview** parameter.

To see additional optional parameters and explanations of how to use them, run the **azure create-all --help** command.

Verification

- To verify that the OpenShift Container Platform secrets are created, list the files in the **<path_to_ccoctl_output_dir>/manifests** directory:

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

Example output

```
azure-ad-pod-identity-webhook-config.yaml
cluster-authentication-02-config.yaml
openshift-cloud-controller-manager-azure-cloud-credentials-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capz-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-azure-disk-credentials-credentials.yaml
openshift-cluster-csi-drivers-azure-file-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-azure-cloud-credentials-credentials.yaml
```

You can verify that the Microsoft Entra ID service accounts are created by querying Azure. For more information, refer to Azure documentation on listing Entra ID service accounts.

3.5.5.2.3. Incorporating the Cloud Credential Operator utility manifests

To implement short-term security credentials managed outside the cluster for individual components, you must move the manifest files that the Cloud Credential Operator utility (**ccoctl**) created to the correct directories for the installation program.

Prerequisites

- You have configured an account with the cloud platform that hosts your cluster.
- You have configured the Cloud Credential Operator utility (**ccoctl**).
- You have created the cloud provider resources that are required for your cluster with the **ccoctl** utility.

Procedure

1. If you did not set the **credentialsMode** parameter in the **install-config.yaml** configuration file to **Manual**, modify the value as shown:

   #### Sample configuration file snippet

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. If you used the **ccoctl** utility to create a new Azure resource group instead of using an existing resource group, modify the **resourceGroupName** parameter in the **install-config.yaml** as shown:

   #### Sample configuration file snippet

   ```
   apiVersion: v1
   baseDomain: example.com
   # ...
   platform:
     azure:
       resourceGroupName: <azure_infra_name> 1
   # ...
   ```

   1 This value must match the user-defined name for Azure resources that was specified with the **--name** argument of the **ccoctl azure create-all** command.

3. If you have not previously created installation manifest files, do so by running the following command:

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   where **<installation_directory>** is the directory in which the installation program creates files.

4. Copy the manifests that the **ccoctl** utility generated to the **manifests** directory that the installation program created by running the following command:

   ```
   $ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
   ```

5. Copy the **tls** directory that contains the private key to the installation directory:

   ```
   $ cp -a /<path_to_ccoctl_output_dir>/tls .
   ```

3.5.6. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

> IMPORTANT
>
> You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- You have configured an account with the cloud platform that hosts your cluster.
- You have the OpenShift Container Platform installation program and the pull secret for your cluster.
- You have an Azure subscription ID and tenant ID.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

  ```
  $ ./openshift-install create cluster --dir <installation_directory> \ ①
      --log-level=info ②
  ```

  ① For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.

  ② To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/.openshift_install.log**.

> IMPORTANT
>
> Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
```

```
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

3.5.7. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.

- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

   ```
   $ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
   ```

   1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

   ```
   $ oc whoami
   ```

   Example output

   ```
   system:admin
   ```

3.5.8. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting .

3.6. INSTALLING A CLUSTER ON AZURE INTO AN EXISTING VNET

In OpenShift Container Platform version 4.18, you can install a cluster into an existing Azure Virtual Network (VNet) on Microsoft Azure. The installation program provisions the rest of the required infrastructure, which you can further customize. To customize the installation, you modify parameters in the **install-config.yaml** file before you install the cluster.

3.6.1. About reusing a VNet for your OpenShift Container Platform cluster

In OpenShift Container Platform 4.18, you can deploy a cluster into an existing Azure Virtual Network (VNet) in Microsoft Azure. If you do, you must also use existing subnets within the VNet and routing rules.

By deploying OpenShift Container Platform into an existing Azure VNet, you might be able to avoid service limit constraints in new accounts or more easily abide by the operational constraints that your company's guidelines set. This is a good option to use if you cannot obtain the infrastructure creation permissions that are required to create the VNet.

3.6.1.1. Requirements for using your VNet

When you deploy a cluster by using an existing VNet, you must perform additional network configuration before you install the cluster. In installer-provisioned infrastructure clusters, the installer usually creates the following components, but it does not create them when you install into an existing VNet:

- Subnets
- Route tables
- VNets
- Network Security Groups

NOTE

The installation program requires that you use the cloud-provided DNS server. Using a custom DNS server is not supported and causes the installation to fail.

If you use a custom VNet, you must correctly configure it and its subnets for the installation program and the cluster to use. The installation program cannot subdivide network ranges for the cluster to use, set route tables for the subnets, or set VNet options like DHCP, so you must do so before you install the cluster.

The cluster must be able to access the resource group that contains the existing VNet and subnets. While all of the resources that the cluster creates are placed in a separate resource group that it creates, some network resources are used from a separate group. Some cluster Operators must be able to access resources in both resource groups. For example, the Machine API controller attaches NICS for the virtual machines that it creates to subnets from the networking resource group.

Your VNet must meet the following characteristics:

- The VNet's CIDR block must contain the **Networking.MachineCIDR** range, which is the IP address pool for cluster machines.
- The VNet and its subnets must belong to the same resource group, and the subnets must be configured to use Azure-assigned DHCP IP addresses instead of static IP addresses.

You must provide two subnets within your VNet, one for the control plane machines and one for the compute machines. Because Azure distributes machines in different availability zones within the region that you specify, your cluster will have high availability by default.

NOTE

By default, if you specify availability zones in the **install-config.yaml** file, the installation program distributes the control plane machines and the compute machines across these availability zones within a region. To ensure high availability for your cluster, select a region with at least three availability zones. If your region contains fewer than three availability zones, the installation program places more than one control plane machine in the available zones.

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the specified subnets exist.
- There are two private subnets, one for the control plane machines and one for the compute machines.
- The subnet CIDRs belong to the machine CIDR that you specified. Machines are not provisioned in availability zones that you do not provide private subnets for. If required, the installation program creates public load balancers that manage the control plane and worker nodes, and Azure allocates a public IP address to them.

NOTE

If you destroy a cluster that uses an existing VNet, the VNet is not deleted.

3.6.1.1.1. Network security group requirements

The network security groups for the subnets that host the compute and control plane machines require specific access to ensure that the cluster communication is correct. You must create rules to allow access to the required cluster communication ports.

IMPORTANT

The network security group rules must be in place before you install the cluster. If you attempt to install a cluster without the required access, the installation program cannot reach the Azure APIs, and installation fails.

Table 3.17. Required ports

Port	Description	Control plane	Compute
80	Allows HTTP traffic		x

Port	Description	Control plane	Compute
443	Allows HTTPS traffic		x
6443	Allows communication to the control plane machines	x	
22623	Allows internal communication to the machine config server for provisioning machines	x	

1. If you are using Azure Firewall to restrict the internet access, then you can configure Azure Firewall to allow the Azure APIs. A network security group rule is not needed. For more information, see "Configuring your firewall" in "Additional resources".

IMPORTANT

Currently, there is no supported way to block or restrict the machine config server endpoint. The machine config server must be exposed to the network so that newly-provisioned machines, which have no existing configuration or state, are able to fetch their configuration. In this model, the root of trust is the certificate signing requests (CSR) endpoint, which is where the kubelet sends its certificate signing request for approval to join the cluster. Because of this, machine configs should not be used to distribute sensitive information, such as secrets and certificates.

To ensure that the machine config server endpoints, ports 22623 and 22624, are secured in bare metal scenarios, customers must configure proper network policies.

Because cluster components do not modify the user-provided network security groups, which the Kubernetes controllers update, a pseudo-network security group is created for the Kubernetes controller to modify without impacting the rest of the environment.

Table 3.18. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	**1936**	Metrics
	9000-**9999**	Host level services, including the node exporter on ports **9100**-**9101** and the Cluster Version Operator on port**9099**.
	10250-**10259**	The default ports that Kubernetes reserves
UDP	**4789**	VXLAN
	6081	Geneve
	9000-**9999**	Host level services, including the node exporter on ports **9100**-**9101**.

Protocol	Port	Description
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port **123** If you configure an external NTP time server, you must open UDP port **123**.
TCP/UDP	**30000**-**32767**	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 3.19. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	**2379**-**2380**	etcd server and peer ports

3.6.1.2. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resources in your clouds than others. For example, you might be able to create application-specific items, like instances, storage, and load balancers, but not networking-related components such as VNets, subnet, or ingress rules.

The Azure credentials that you use when you create your cluster do not need the networking permissions that are required to make VNets and core networking components within the VNet, such as subnets, routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as load balancers, security groups, storage accounts, and nodes.

3.6.1.3. Isolation between clusters

Because the cluster is unable to modify network security groups in an existing subnet, there is no way to isolate clusters from each other on the VNet.

Additional resources

- About the OVN-Kubernetes network plugin
- Configuring your firewall

3.6.2. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Microsoft Azure.

Prerequisites

- You have the OpenShift Container Platform installation program and the pull secret for your cluster.
- You have an Azure subscription ID and tenant ID.
- If you are installing the cluster using a service principal, you have its application ID and password.
- If you are installing the cluster using a system-assigned managed identity, you have enabled it on the virtual machine that you will run the installation program from.
- If you are installing the cluster using a user-assigned managed identity, you have met these prerequisites:
  - You have its client ID.
  - You have assigned it to the virtual machine that you will run the installation program from.

Procedure

1. Optional: If you have run the installation program on this computer before, and want to use an alternative service principal or managed identity, go to the ~/**.azure**/ directory and delete the **osServicePrincipal.json** configuration file.
   Deleting this file prevents the installation program from automatically reusing subscription and authentication values from a previous installation.

2. Create the **install-config.yaml** file.

   a. Change to the directory that contains the installation program and run the following command:

   ```
   $ ./openshift-install create install-config --dir <installation_directory> 1
   ```

   1 For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

   When specifying the directory:

   - Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
   - Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

   b. At the prompts, provide the configuration details for your cloud:

      i. Optional: Select an SSH key to use to access your cluster machines.

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

ii. Select **azure** as the platform to target.
If the installation program cannot locate the **osServicePrincipal.json** configuration file from a previous installation, you are prompted for Azure subscription and authentication values.

iii. Enter the following Azure parameter values for your subscription:

- **azure subscription id** Enter the subscription ID to use for the cluster.
- **azure tenant id** Enter the tenant ID.

iv. Depending on the Azure identity you are using to deploy the cluster, do one of the following when prompted for the **azure service principal client id**

- If you are using a service principal, enter its application ID.
- If you are using a system-assigned managed identity, leave this value blank.
- If you are using a user-assigned managed identity, specify its client ID.

v. Depending on the Azure identity you are using to deploy the cluster, do one of the following when prompted for the **azure service principal client secret**

- If you are using a service principal, enter its password.
- If you are using a system-assigned managed identity, leave this value blank.
- If you are using a user-assigned managed identity, leave this value blank.

vi. Select the region to deploy the cluster to.

vii. Select the base domain to deploy the cluster to. The base domain corresponds to the Azure DNS Zone that you created for your cluster.

viii. Enter a descriptive name for your cluster.

IMPORTANT

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see Resolve reserved resource name errors in the Azure documentation.

3. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.

4. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

If previously not detected, the installation program creates an **osServicePrincipal.json** configuration file and stores this file in the ~/**.azure**/ directory on your computer. This ensures that the installation program can load the profile when it is creating an OpenShift Container Platform cluster on the target platform.

Additional resources

- Installation configuration parameters for Azure

3.6.2.1. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 3.20. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or Hyper-Threading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

NOTE

For OpenShift Container Platform version 4.18, RHCOS is based on RHEL version 9.4, which updates the micro-architecture requirements. The following list contains the minimum instruction set architectures (ISA) that each architecture requires:

- x86-64 architecture requires x86-64-v2 ISA
- ARM64 architecture requires ARMv8.0-A ISA
- IBM Power architecture requires Power 9 ISA
- s390x architecture requires z14 ISA

For more information, see Architectures (RHEL documentation).

IMPORTANT

You are required to use Azure virtual machines that have the **premiumIO** parameter set to **true**.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- Optimizing storage

3.6.2.2. Tested instance types for Azure

The following Microsoft Azure instance types have been tested with OpenShift Container Platform.

Example 3.7. Machine types based on 64-bit x86 architecture

- **standardBasv2Family**
- **standardBSFamily**
- **standardBsv2Family**
- **standardDADSv5Family**
- **standardDASv4Family**
- **standardDASv5Family**
- **standardDCACCV5Family**
- **standardDCADCCV5Family**
- **standardDCADSv5Family**
- **standardDCASv5Family**
- **standardDCSv3Family**

- **standardDCSv2Family**
- **standardDDCSv3Family**
- **standardDDSv4Family**
- **standardDDSv5Family**
- **standardDLDSv5Family**
- **standardDLSv5Family**
- **standardDSFamily**
- **standardDSv2Family**
- **standardDSv2PromoFamily**
- **standardDSv3Family**
- **standardDSv4Family**
- **standardDSv5Family**
- **standardEADSv5Family**
- **standardEASv4Family**
- **standardEASv5Family**
- **standardEBDSv5Family**
- **standardEBSv5Family**
- **standardECACCV5Family**
- **standardECADCCV5Family**
- **standardECADSv5Family**
- **standardECASv5Family**
- **standardEDSv4Family**
- **standardEDSv5Family**
- **standardEIADSv5Family**
- **standardEIASv4Family**
- **standardEIASv5Family**
- **standardEIBDSv5Family**
- **standardEIBSv5Family**
- **standardEIDSv5Family**

- **standardEISv3Family**
- **standardEISv5Family**
- **standardESv3Family**
- **standardESv4Family**
- **standardESv5Family**
- **standardFXMDVSFamily**
- **standardFSFamily**
- **standardFSv2Family**
- **standardGSFamily**
- **standardHBrsv2Family**
- **standardHBSFamily**
- **standardHBv4Family**
- **standardHCSFamily**
- **standardHXFamily**
- **standardLASv3Family**
- **standardLSFamily**
- **standardLSv2Family**
- **standardLSv3Family**
- **standardMDSHighMemoryv3Family**
- **standardMDSMediumMemoryv2Family**
- **standardMDSMediumMemoryv3Family**
- **standardMIDSHighMemoryv3Family**
- **standardMIDSMediumMemoryv2Family**
- **standardMISHighMemoryv3Family**
- **standardMISMediumMemoryv2Family**
- **standardMSFamily**
- **standardMSHighMemoryv3Family**
- **standardMSMediumMemoryv2Family**
- **standardMSMediumMemoryv3Family**

- **StandardNCADSA100v4Family**
- **Standard NCASv3_T4 Family**
- **standardNCSv3Family**
- **standardNDSv2Family**
- **StandardNGADSV620v1Family**
- **standardNPSFamily**
- **StandardNVADSA10v5Family**
- **standardNVSv3Family**
- **standardXEISv4Family**

3.6.2.3. Tested instance types for Azure on 64-bit ARM infrastructures

The following Microsoft Azure ARM64 instance types have been tested with OpenShift Container Platform.

Example 3.8. Machine types based on 64-bit ARM architecture

- **standardBpsv2Family**
- **standardDPSv5Family**
- **standardDPDSv5Family**
- **standardDPLDSv5Family**
- **standardDPLSv5Family**
- **standardEPSv5Family**
- **standardEPDSv5Family**
- **StandardDpdsv6Family**
- **StandardDpldsv6Famil**
- **StandardDplsv6Family**
- **StandardDpsv6Family**
- **StandardEpdsv6Family**
- **StandardEpsv6Family**

3.6.2.4. Enabling trusted launch for Azure VMs

You can enable two trusted launch features when installing your cluster on Azure: secure boot and virtualized Trusted Platform Modules.

For more information about the sizes of virtual machines that support the trusted launch features, see Virtual machine sizes.

IMPORTANT

Trusted launch is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see Technology Preview Features Support Scope .

Prerequisites

- You have created an **install-config.yaml** file.

Procedure

- Edit the **install-config.yaml** file before deploying your cluster:
  - Enable trusted launch only on control plane by adding the following stanza:

```
controlPlane:
  platform:
    azure:
      settings:
        securityType: TrustedLaunch
        trustedLaunch:
          uefiSettings:
            secureBoot: Enabled
            virtualizedTrustedPlatformModule: Enabled
```

  - Enable trusted launch only on compute node by adding the following stanza:

```
compute:
  platform:
    azure:
      settings:
        securityType: TrustedLaunch
        trustedLaunch:
          uefiSettings:
            secureBoot: Enabled
            virtualizedTrustedPlatformModule: Enabled
```

  - Enable trusted launch on all nodes by adding the following stanza:

```
platform:
  azure:
    settings:
      securityType: TrustedLaunch
      trustedLaunch:
        uefiSettings:
          secureBoot: Enabled
          virtualizedTrustedPlatformModule: Enabled
```

3.6.2.5. Enabling confidential VMs

You can enable confidential VMs when installing your cluster. You can enable confidential VMs for compute nodes, control plane nodes, or all nodes.

IMPORTANT

Using confidential VMs is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see Technology Preview Features Support Scope .

You can use confidential VMs with the following VM sizes:

- DCasv5-series
- DCadsv5-series
- ECasv5-series
- ECadsv5-series

IMPORTANT

Confidential VMs are currently not supported on 64-bit ARM architectures.

Prerequisites

- You have created an **install-config.yaml** file.

Procedure

- Edit the **install-config.yaml** file before deploying your cluster:
  - Enable confidential VMs only on control plane by adding the following stanza:

```
controlPlane:
  platform:
    azure:
      settings:
        securityType: ConfidentialVM
        confidentialVM:
          uefiSettings:
            secureBoot: Enabled
            virtualizedTrustedPlatformModule: Enabled
      osDisk:
        securityProfile:
          securityEncryptionType: VMGuestStateOnly
```

- Enable confidential VMs only on compute nodes by adding the following stanza:

```
compute:
  platform:
    azure:
      settings:
        securityType: ConfidentialVM
        confidentialVM:
          uefiSettings:
            secureBoot: Enabled
            virtualizedTrustedPlatformModule: Enabled
      osDisk:
        securityProfile:
          securityEncryptionType: VMGuestStateOnly
```

- Enable confidential VMs on all nodes by adding the following stanza:

```
platform:
  azure:
    settings:
      securityType: ConfidentialVM
      confidentialVM:
        uefiSettings:
          secureBoot: Enabled
          virtualizedTrustedPlatformModule: Enabled
    osDisk:
      securityProfile:
        securityEncryptionType: VMGuestStateOnly
```

3.6.2.6. Sample customized install-config.yaml file for Azure

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```
apiVersion: v1
baseDomain: example.com ❶
controlPlane: ❷
  hyperthreading: Enabled ❸ ❹
  name: master
  platform:
    azure:
      encryptionAtHost: true
      ultraSSDCapability: Enabled
      osDisk:
        diskSizeGB: 1024 5
        diskType: Premium_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      type: Standard_D8s_v3
  replicas: 3
compute: 6
- hyperthreading: Enabled 7 8
  name: worker
  platform:
    azure:
      ultraSSDCapability: Enabled
      type: Standard_D2s_v3
      encryptionAtHost: true
      osDisk:
        diskSizeGB: 512 9
        diskType: Standard_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      zones: 10
      - "1"
      - "2"
      - "3"
  replicas: 5
metadata:
  name: test-cluster 11
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 12
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    defaultMachinePlatform:
      osImage: 13
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      ultraSSDCapability: Enabled
    baseDomainResourceGroupName: resource_group 14
    region: centralus 15
    resourceGroupName: existing_resource_group 16
    networkResourceGroupName: vnet_resource_group 17
    virtualNetwork: vnet 18
    controlPlaneSubnet: control_plane_subnet 19
    computeSubnet: compute_subnet 20
    outboundType: Loadbalancer
    cloudName: AzurePublicCloud
pullSecret: '{"auths": ...}' 21
fips: false 22
sshKey: ssh-ed25519 AAAA... 23
```

1 11 15 21 Required. The installation program prompts you for this value.

2 6 If you do not provide these parameters and values, the installation program provides the default value.

3 7 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, **-**, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

4 8 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger virtual machine types, such as **Standard_D8s_v3**, for your machines if you disable simultaneous multithreading.

5 9 You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes is 1024 GB.

10 Specify a list of zones to deploy your machines to. For high availability, specify at least two zones.

12 The cluster network plugin to install. The default value **OVNKubernetes** is the only supported value.

13 Optional: A custom Red Hat Enterprise Linux CoreOS (RHCOS) image that should be used to boot control plane and compute machines. The **publisher**, **offer**, **sku**, and **version** parameters under **platform.azure.defaultMachinePlatform.osImage** apply to both control plane and compute machines. If the parameters under **controlPlane.platform.azure.osImage** or **compute.platform.azure.osImage** are set, they override the **platform.azure.defaultMachinePlatform.osImage** parameters.

14 Specify the name of the resource group that contains the DNS zone for your base domain.

16 Specify the name of an already existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.

17 If you use an existing VNet, specify the name of the resource group that contains it.

18 If you use an existing VNet, specify its name.

19 If you use an existing VNet, specify the name of the subnet to host the control plane machines.

20 If you use an existing VNet, specify the name of the subnet to host the compute machines.

22 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

> **IMPORTANT**
>
> To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Switching RHEL to FIPS mode.
>
> When running Red Hat Enterprise Linux (RHEL) or Red Hat Enterprise Linux CoreOS (RHCOS) booted in FIPS mode, OpenShift Container Platform core components use the RHEL cryptographic libraries that have been submitted to NIST for FIPS 140-2/140-3 Validation on only the x86_64, ppc64le, and s390x architectures.

23 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.

> **NOTE**
>
> For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

3.6.2.7. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.

NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.

2 A proxy URL to use for creating HTTPS connections outside the cluster.

3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use * to bypass the proxy for all destinations.

4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

NOTE

The installation program does not support the proxy **readinessEndpoints** field.

NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster** **Proxy** object is still created, but it will have a nil **spec**.

NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

Additional resources

- For more details about Accelerated Networking, see Accelerated Networking for Microsoft Azure VMs.

3.6.3. Alternatives to storing administrator-level secrets in the kube-system project

By default, administrator secrets are stored in the **kube-system** project. If you configured the **credentialsMode** parameter in the **install-config.yaml** file to **Manual**, you must use one of the following alternatives:

- To manage long-term cloud credentials manually, follow the procedure in Manually creating long-term credentials.
- To implement short-term credentials that are managed outside the cluster for individual components, follow the procedures in Configuring an Azure cluster to use short-term credentials.

3.6.3.1. Manually creating long-term credentials

The Cloud Credential Operator (CCO) can be put into manual mode prior to installation in environments where the cloud identity and access management (IAM) APIs are not reachable, or the administrator prefers not to store an administrator-level credential secret in the cluster **kube-system** namespace.

Procedure

1. If you did not set the **credentialsMode** parameter in the **install-config.yaml** configuration file to **Manual**, modify the value as shown:

   Sample configuration file snippet

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. If you have not previously created installation manifest files, do so by running the following command:

```
$ openshift-install create manifests --dir <installation_directory>
```

where **<installation_directory>** is the directory in which the installation program creates files.

3. Set a **$RELEASE_IMAGE** variable with the release image from your installation file by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

4. Extract the list of **CredentialsRequest** custom resources (CRs) from the OpenShift Container Platform release image by running the following command:

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1 The **--included** parameter includes only the manifests that your specific cluster configuration requires.

2 Specify the location of the **install-config.yaml** file.

3 Specify the path to the directory where you want to store the **CredentialsRequest** objects. If the specified directory does not exist, this command creates it.

This command creates a YAML file for each **CredentialsRequest** object.

Sample CredentialsRequest object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
  ...
```

5. Create YAML files for secrets in the **openshift-install** manifests directory that you generated previously. The secrets must be stored using the namespace and secret name defined in the **spec.secretRef** for each **CredentialsRequest** object.

Sample **CredentialsRequest** object with secrets

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Sample **Secret** object

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  azure_subscription_id: <base64_encoded_azure_subscription_id>
  azure_client_id: <base64_encoded_azure_client_id>
  azure_client_secret: <base64_encoded_azure_client_secret>
  azure_tenant_id: <base64_encoded_azure_tenant_id>
  azure_resource_prefix: <base64_encoded_azure_resource_prefix>
  azure_resourcegroup: <base64_encoded_azure_resourcegroup>
  azure_region: <base64_encoded_azure_region>
```

IMPORTANT

Before upgrading a cluster that uses manually maintained credentials, you must ensure that the CCO is in an upgradeable state.

3.6.3.2. Configuring an Azure cluster to use short-term credentials

To install a cluster that uses Microsoft Entra Workload ID, you must configure the Cloud Credential Operator utility and create the required Azure resources for your cluster.

3.6.3.2.1. Configuring the Cloud Credential Operator utility

To create and manage cloud credentials from outside of the cluster when the Cloud Credential Operator (CCO) is operating in manual mode, extract and prepare the CCO utility (**ccoctl**) binary.

NOTE

The **ccoctl** utility is a Linux binary that must run in a Linux environment.

Prerequisites

- You have access to an OpenShift Container Platform account with cluster administrator access.
- You have installed the OpenShift CLI (**oc**).
- You have created a global Azure account for the **ccoctl** utility to use with the following permissions:
  - **Microsoft.Resources/subscriptions/resourceGroups/read**
  - **Microsoft.Resources/subscriptions/resourceGroups/write**
  - **Microsoft.Resources/subscriptions/resourceGroups/delete**
  - **Microsoft.Authorization/roleAssignments/read**
  - **Microsoft.Authorization/roleAssignments/delete**
  - **Microsoft.Authorization/roleAssignments/write**
  - **Microsoft.Authorization/roleDefinitions/read**
  - **Microsoft.Authorization/roleDefinitions/write**
  - **Microsoft.Authorization/roleDefinitions/delete**
  - **Microsoft.Storage/storageAccounts/listkeys/action**
  - **Microsoft.Storage/storageAccounts/delete**
  - **Microsoft.Storage/storageAccounts/read**
  - **Microsoft.Storage/storageAccounts/write**
  - **Microsoft.Storage/storageAccounts/blobServices/containers/delete**
  - **Microsoft.Storage/storageAccounts/blobServices/containers/read**
  - **Microsoft.Storage/storageAccounts/blobServices/containers/write**
  - **Microsoft.ManagedIdentity/userAssignedIdentities/delete**
  - **Microsoft.ManagedIdentity/userAssignedIdentities/read**
  - **Microsoft.ManagedIdentity/userAssignedIdentities/write**
  - **Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/read**
  - **Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/write**
  - **Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/delete**
  - **Microsoft.Storage/register/action**

- **Microsoft.ManagedIdentity/register/action**

Procedure

1. Set a variable for the OpenShift Container Platform release image by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. Obtain the CCO container image from the OpenShift Container Platform release image by running the following command:

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

NOTE

Ensure that the architecture of the **$RELEASE_IMAGE** matches the architecture of the environment in which you will use the **ccoctl** tool.

3. Extract the **ccoctl** binary from the CCO container image within the OpenShift Container Platform release image by running the following command:

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 For **<rhel_version>**, specify the value that corresponds to the version of Red Hat Enterprise Linux (RHEL) that the host uses. If no value is specified, **ccoctl.rhel8** is used by default. The following values are valid:

- **rhel8**: Specify this value for hosts that use RHEL 8.
- **rhel9**: Specify this value for hosts that use RHEL 9.

4. Change the permissions to make **ccoctl** executable by running the following command:

```
$ chmod 775 ccoctl.<rhel_version>
```

Verification

- To verify that **ccoctl** is ready to use, display the help file. Use a relative file name when you run the command, for example:

```
$ ./ccoctl.rhel9
```

Example output

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]
```

```
Available Commands:
  aws          Manage credentials objects for AWS cloud
  azure        Manage credentials objects for Azure
  gcp          Manage credentials objects for Google cloud
  help         Help about any command
  ibmcloud     Manage credentials objects for {ibm-cloud-title}
  nutanix      Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

3.6.3.2.2. Creating Azure resources with the Cloud Credential Operator utility

You can use the **ccoctl azure create-all** command to automate the creation of Azure resources.

NOTE

By default, **ccoctl** creates objects in the directory in which the commands are run. To create the objects in a different directory, use the **--output-dir** flag. This procedure uses **<path_to_ccoctl_output_dir>** to refer to this directory.

Prerequisites

You must have:

- Extracted and prepared the **ccoctl** binary.
- Access to your Microsoft Azure account by using the Azure CLI.

Procedure

1. Set a **$RELEASE_IMAGE** variable with the release image from your installation file by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. Extract the list of **CredentialsRequest** objects from the OpenShift Container Platform release image by running the following command:

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1 The **--included** parameter includes only the manifests that your specific cluster configuration requires.

2 Specify the location of the **install-config.yaml** file.

3 Specify the path to the directory where you want to store the **CredentialsRequest** objects. If the specified directory does not exist, this command creates it.

NOTE

This command might take a few moments to run.

3. To enable the **ccoctl** utility to detect your Azure credentials automatically, log in to the Azure CLI by running the following command:

```
$ az login
```

4. Use the **ccoctl** tool to process all **CredentialsRequest** objects by running the following command:

```
$ ccoctl azure create-all \
  --name=<azure_infra_name> \ 1
  --output-dir=<ccoctl_output_dir> \ 2
  --region=<azure_region> \ 3
  --subscription-id=<azure_subscription_id> \ 4
  --credentials-requests-dir=<path_to_credentials_requests_directory> \ 5
  --dnszone-resource-group-name=<azure_dns_zone_resource_group_name> \ 6
  --tenant-id=<azure_tenant_id> 7
```

1 Specify the user-defined name for all created Azure resources used for tracking.

2 Optional: Specify the directory in which you want the **ccoctl** utility to create objects. By default, the utility creates objects in the directory in which the commands are run.

3 Specify the Azure region in which cloud resources will be created.

4 Specify the Azure subscription ID to use.

5 Specify the directory containing the files for the component **CredentialsRequest** objects.

6 Specify the name of the resource group containing the cluster's base domain Azure DNS zone.

7 Specify the Azure tenant ID to use.

NOTE

If your cluster uses Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set, you must include the **--enable-tech-preview** parameter.

To see additional optional parameters and explanations of how to use them, run the **azure create-all --help** command.

Verification

- To verify that the OpenShift Container Platform secrets are created, list the files in the **<path_to_ccoctl_output_dir>/manifests** directory:

  ```
  $ ls <path_to_ccoctl_output_dir>/manifests
  ```

  ### Example output

  ```
  azure-ad-pod-identity-webhook-config.yaml
  cluster-authentication-02-config.yaml
  openshift-cloud-controller-manager-azure-cloud-credentials-credentials.yaml
  openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
  openshift-cluster-api-capz-manager-bootstrap-credentials-credentials.yaml
  openshift-cluster-csi-drivers-azure-disk-credentials-credentials.yaml
  openshift-cluster-csi-drivers-azure-file-credentials-credentials.yaml
  openshift-image-registry-installer-cloud-credentials-credentials.yaml
  openshift-ingress-operator-cloud-credentials-credentials.yaml
  openshift-machine-api-azure-cloud-credentials-credentials.yaml
  ```

  You can verify that the Microsoft Entra ID service accounts are created by querying Azure. For more information, refer to Azure documentation on listing Entra ID service accounts.

3.6.3.2.3. Incorporating the Cloud Credential Operator utility manifests

To implement short-term security credentials managed outside the cluster for individual components, you must move the manifest files that the Cloud Credential Operator utility (**ccoctl**) created to the correct directories for the installation program.

Prerequisites

- You have configured an account with the cloud platform that hosts your cluster.
- You have configured the Cloud Credential Operator utility (**ccoctl**).
- You have created the cloud provider resources that are required for your cluster with the **ccoctl** utility.

Procedure

1. If you did not set the **credentialsMode** parameter in the **install-config.yaml** configuration file to **Manual**, modify the value as shown:

   #### Sample configuration file snippet

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. If you used the **ccoctl** utility to create a new Azure resource group instead of using an existing resource group, modify the **resourceGroupName** parameter in the **install-config.yaml** as shown:

   #### Sample configuration file snippet

```
apiVersion: v1
baseDomain: example.com
# ...
platform:
  azure:
    resourceGroupName: <azure_infra_name> 1
# ...
```

1 This value must match the user-defined name for Azure resources that was specified with the **--name** argument of the **ccoctl azure create-all** command.

3. If you have not previously created installation manifest files, do so by running the following command:

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   where **<installation_directory>** is the directory in which the installation program creates files.

4. Copy the manifests that the **ccoctl** utility generated to the **manifests** directory that the installation program created by running the following command:

   ```
   $ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
   ```

5. Copy the **tls** directory that contains the private key to the installation directory:

   ```
   $ cp -a /<path_to_ccoctl_output_dir>/tls .
   ```

3.6.4. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- You have configured an account with the cloud platform that hosts your cluster.
- You have the OpenShift Container Platform installation program and the pull secret for your cluster.
- You have an Azure subscription ID and tenant ID.

Procedure

- Change to the directory that contains the installation program and initialize the cluster deployment:

  ```
  $ ./openshift-install create cluster --dir <installation_directory> \ 1
      --log-level=info 2
  ```

1. For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.
2. To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/.openshift_install.log**.

IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Additional resources

- See Accessing the web console for more details about accessing and understanding the OpenShift Container Platform web console.

3.6.5. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting .

3.7. INSTALLING A PRIVATE CLUSTER ON AZURE

In OpenShift Container Platform version 4.18, you can install a private cluster into an existing Azure Virtual Network (VNet) on Microsoft Azure. The installation program provisions the rest of the required infrastructure, which you can further customize. To customize the installation, you modify parameters in the **install-config.yaml** file before you install the cluster.

3.7.1. Private clusters

You can deploy a private OpenShift Container Platform cluster that does not expose external endpoints. Private clusters are accessible from only an internal network and are not visible to the internet.

By default, OpenShift Container Platform is provisioned to use publicly-accessible DNS and endpoints. A private cluster sets the DNS, Ingress Controller, and API server to private when you deploy your cluster. This means that the cluster resources are only accessible from your internal network and are not visible to the internet.

IMPORTANT

If the cluster has any public subnets, load balancer services created by administrators might be publicly accessible. To ensure cluster security, verify that these services are explicitly annotated as private.

To deploy a private cluster, you must:

- Use existing networking that meets your requirements. Your cluster resources might be shared between other clusters on the network.
- Deploy from a machine that has access to:
  - The API services for the cloud to which you provision.
  - The hosts on the network that you provision.
  - The internet to obtain installation media.

You can use any machine that meets these access requirements and follows your company’s guidelines. For example, this machine can be a bastion host on your cloud network or a machine that has access to the network through a VPN.

3.7.1.1. Private clusters in Azure

To create a private cluster on Microsoft Azure, you must provide an existing private VNet and subnets to host the cluster. The installation program must also be able to resolve the DNS records that the cluster requires. The installation program configures the Ingress Operator and API server for only internal traffic.

Depending how your network connects to the private VNET, you might need to use a DNS forwarder to resolve the cluster’s private DNS records. The cluster’s machines use **168.63.129.16** internally for DNS resolution. For more information, see What is Azure Private DNS? and What is IP address 168.63.129.16?

in the Azure documentation.

The cluster still requires access to internet to access the Azure APIs.

The following items are not required or created when you install a private cluster:

- A **BaseDomainResourceGroup**, since the cluster does not create public records
- Public IP addresses
- Public DNS records
- Public endpoints

> The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

3.7.1.1.1. Limitations

Private clusters on Azure are subject to only the limitations that are associated with the use of an existing VNet.

3.7.1.2. User-defined outbound routing

In OpenShift Container Platform, you can choose your own outbound routing for a cluster to connect to the internet. This allows you to skip the creation of public IP addresses and the public load balancer.

You can configure user-defined routing by modifying parameters in the **install-config.yaml** file before installing your cluster. A pre-existing VNet is required to use outbound routing when installing a cluster; the installation program is not responsible for configuring this.

When configuring a cluster to use user-defined routing, the installation program does not create the following resources:

- Outbound rules for access to the internet.
- Public IPs for the public load balancer.
- Kubernetes Service object to add the cluster machines to the public load balancer for outbound requests.

You must ensure the following items are available before setting user-defined routing:

- Egress to the internet is possible to pull container images, unless using an OpenShift image registry mirror.
- The cluster can access Azure APIs.
- Various allowlist endpoints are configured. You can reference these endpoints in the *Configuring your firewall* section.

There are several pre-existing networking setups that are supported for internet access using user-defined routing.

Private cluster with network address translation

You can use Azure VNET network address translation (NAT) to provide outbound internet access for the subnets in your cluster. You can reference Create a NAT gateway using Azure CLI in the Azure documentation for configuration instructions.

When using a VNet setup with Azure NAT and user-defined routing configured, you can create a private cluster with no public endpoints.

Private cluster with Azure Firewall
You can use Azure Firewall to provide outbound routing for the VNet used to install the cluster. You can learn more about providing user-defined routing with Azure Firewall in the Azure documentation.

When using a VNet setup with Azure Firewall and user-defined routing configured, you can create a private cluster with no public endpoints.

Private cluster with a proxy configuration
You can use a proxy with user-defined routing to allow egress to the internet. You must ensure that cluster Operators do not access Azure APIs using a proxy; Operators must have access to Azure APIs outside of the proxy.

When using the default route table for subnets, with **0.0.0.0/0** populated automatically by Azure, all Azure API requests are routed over Azure's internal network even though the IP addresses are public. As long as the Network Security Group rules allow egress to Azure API endpoints, proxies with user-defined routing configured allow you to create private clusters with no public endpoints.

Private cluster with no internet access
You can install a private network that restricts all access to the internet, except the Azure API. This is accomplished by mirroring the release image registry locally. Your cluster must have access to the following:

- An OpenShift image registry mirror that allows for pulling container images
- Access to Azure APIs

With these requirements available, you can use user-defined routing to create private clusters with no public endpoints.

3.7.2. About reusing a VNet for your OpenShift Container Platform cluster

In OpenShift Container Platform 4.18, you can deploy a cluster into an existing Azure Virtual Network (VNet) in Microsoft Azure. If you do, you must also use existing subnets within the VNet and routing rules.

By deploying OpenShift Container Platform into an existing Azure VNet, you might be able to avoid service limit constraints in new accounts or more easily abide by the operational constraints that your company's guidelines set. This is a good option to use if you cannot obtain the infrastructure creation permissions that are required to create the VNet.

3.7.2.1. Requirements for using your VNet

When you deploy a cluster by using an existing VNet, you must perform additional network configuration before you install the cluster. In installer-provisioned infrastructure clusters, the installer usually creates the following components, but it does not create them when you install into an existing VNet:

- Subnets
- Route tables

- VNets
- Network Security Groups

NOTE

The installation program requires that you use the cloud-provided DNS server. Using a custom DNS server is not supported and causes the installation to fail.

If you use a custom VNet, you must correctly configure it and its subnets for the installation program and the cluster to use. The installation program cannot subdivide network ranges for the cluster to use, set route tables for the subnets, or set VNet options like DHCP, so you must do so before you install the cluster.

The cluster must be able to access the resource group that contains the existing VNet and subnets. While all of the resources that the cluster creates are placed in a separate resource group that it creates, some network resources are used from a separate group. Some cluster Operators must be able to access resources in both resource groups. For example, the Machine API controller attaches NICS for the virtual machines that it creates to subnets from the networking resource group.

Your VNet must meet the following characteristics:

- The VNet's CIDR block must contain the **Networking.MachineCIDR** range, which is the IP address pool for cluster machines.
- The VNet and its subnets must belong to the same resource group, and the subnets must be configured to use Azure-assigned DHCP IP addresses instead of static IP addresses.

You must provide two subnets within your VNet, one for the control plane machines and one for the compute machines. Because Azure distributes machines in different availability zones within the region that you specify, your cluster will have high availability by default.

NOTE

By default, if you specify availability zones in the **install-config.yaml** file, the installation program distributes the control plane machines and the compute machines across these availability zones within a region. To ensure high availability for your cluster, select a region with at least three availability zones. If your region contains fewer than three availability zones, the installation program places more than one control plane machine in the available zones.

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the specified subnets exist.
- There are two private subnets, one for the control plane machines and one for the compute machines.
- The subnet CIDRs belong to the machine CIDR that you specified. Machines are not provisioned in availability zones that you do not provide private subnets for.

NOTE

If you destroy a cluster that uses an existing VNet, the VNet is not deleted.

3.7.2.1.1. Network security group requirements

The network security groups for the subnets that host the compute and control plane machines require specific access to ensure that the cluster communication is correct. You must create rules to allow access to the required cluster communication ports.

IMPORTANT

The network security group rules must be in place before you install the cluster. If you attempt to install a cluster without the required access, the installation program cannot reach the Azure APIs, and installation fails.

Table 3.21. Required ports

Port	Description	Control plane	Compute
80	Allows HTTP traffic		x
443	Allows HTTPS traffic		x
6443	Allows communication to the control plane machines	x	
22623	Allows internal communication to the machine config server for provisioning machines	x	

1. If you are using Azure Firewall to restrict the internet access, then you can configure Azure Firewall to allow the Azure APIs. A network security group rule is not needed. For more information, see "Configuring your firewall" in "Additional resources".

IMPORTANT

Currently, there is no supported way to block or restrict the machine config server endpoint. The machine config server must be exposed to the network so that newly-provisioned machines, which have no existing configuration or state, are able to fetch their configuration. In this model, the root of trust is the certificate signing requests (CSR) endpoint, which is where the kubelet sends its certificate signing request for approval to join the cluster. Because of this, machine configs should not be used to distribute sensitive information, such as secrets and certificates.

To ensure that the machine config server endpoints, ports 22623 and 22624, are secured in bare metal scenarios, customers must configure proper network policies.

Because cluster components do not modify the user-provided network security groups, which the Kubernetes controllers update, a pseudo-network security group is created for the Kubernetes controller to modify without impacting the rest of the environment.

Table 3.22. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests

Protocol	Port	Description
TCP	**1936**	Metrics
	9000-**9999**	Host level services, including the node exporter on ports **9100**-**9101** and the Cluster Version Operator on port**9099**.
	10250-**10259**	The default ports that Kubernetes reserves
UDP	**4789**	VXLAN
	6081	Geneve
	9000-**9999**	Host level services, including the node exporter on ports **9100**-**9101**.
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port **123** If you configure an external NTP time server, you must open UDP port **123**.
TCP/UDP	**30000**-**32767**	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 3.23. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	**2379**-**2380**	etcd server and peer ports

3.7.2.2. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resources in your clouds than others. For example, you might be able to create application-specific items, like instances, storage, and load balancers, but not networking-related components such as VNets, subnet, or ingress rules.

The Azure credentials that you use when you create your cluster do not need the networking permissions that are required to make VNets and core networking components within the VNet, such as subnets, routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as load balancers, security groups, storage accounts, and nodes.

3.7.2.3. Isolation between clusters

Because the cluster is unable to modify network security groups in an existing subnet, there is no way to isolate clusters from each other on the VNet.

Additional resources

- About the OVN-Kubernetes network plugin
- Configuring your firewall

3.7.3. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```

IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.

NOTE

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

Additional resources

- Installation configuration parameters for Azure

3.7.3.1. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 3.24. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or Hyper-Threading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

NOTE

For OpenShift Container Platform version 4.18, RHCOS is based on RHEL version 9.4, which updates the micro-architecture requirements. The following list contains the minimum instruction set architectures (ISA) that each architecture requires:

- x86-64 architecture requires x86-64-v2 ISA
- ARM64 architecture requires ARMv8.0-A ISA
- IBM Power architecture requires Power 9 ISA
- s390x architecture requires z14 ISA

For more information, see Architectures (RHEL documentation).

IMPORTANT

You are required to use Azure virtual machines that have the **premiumIO** parameter set to **true**.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- Optimizing storage

3.7.3.2. Tested instance types for Azure

The following Microsoft Azure instance types have been tested with OpenShift Container Platform.

Example 3.9. Machine types based on 64-bit x86 architecture

- **standardBasv2Family**
- **standardBSFamily**
- **standardBsv2Family**
- **standardDADSv5Family**
- **standardDASv4Family**
- **standardDASv5Family**
- **standardDCACCV5Family**
- **standardDCADCCV5Family**
- **standardDCADSv5Family**
- **standardDCASv5Family**
- **standardDCSv3Family**
- **standardDCSv2Family**
- **standardDDCSv3Family**
- **standardDDSv4Family**
- **standardDDSv5Family**
- **standardDLDSv5Family**
- **standardDLSv5Family**
- **standardDSFamily**
- **standardDSv2Family**
- **standardDSv2PromoFamily**

- **standardDSv3Family**
- **standardDSv4Family**
- **standardDSv5Family**
- **standardEADSv5Family**
- **standardEASv4Family**
- **standardEASv5Family**
- **standardEBDSv5Family**
- **standardEBSv5Family**
- **standardECACCV5Family**
- **standardECADCCV5Family**
- **standardECADSv5Family**
- **standardECASv5Family**
- **standardEDSv4Family**
- **standardEDSv5Family**
- **standardEIADSv5Family**
- **standardEIASv4Family**
- **standardEIASv5Family**
- **standardEIBDSv5Family**
- **standardEIBSv5Family**
- **standardEIDSv5Family**
- **standardEISv3Family**
- **standardEISv5Family**
- **standardESv3Family**
- **standardESv4Family**
- **standardESv5Family**
- **standardFXMDVSFamily**
- **standardFSFamily**
- **standardFSv2Family**
- **standardGSFamily**

- **standardHBrsv2Family**
- **standardHBSFamily**
- **standardHBv4Family**
- **standardHCSFamily**
- **standardHXFamily**
- **standardLASv3Family**
- **standardLSFamily**
- **standardLSv2Family**
- **standardLSv3Family**
- **standardMDSHighMemoryv3Family**
- **standardMDSMediumMemoryv2Family**
- **standardMDSMediumMemoryv3Family**
- **standardMIDSHighMemoryv3Family**
- **standardMIDSMediumMemoryv2Family**
- **standardMISHighMemoryv3Family**
- **standardMISMediumMemoryv2Family**
- **standardMSFamily**
- **standardMSHighMemoryv3Family**
- **standardMSMediumMemoryv2Family**
- **standardMSMediumMemoryv3Family**
- **StandardNCADSA100v4Family**
- **Standard NCASv3_T4 Family**
- **standardNCSv3Family**
- **standardNDSv2Family**
- **StandardNGADSV620v1Family**
- **standardNPSFamily**
- **StandardNVADSA10v5Family**
- **standardNVSv3Family**
- **standardXEISv4Family**

3.7.3.3. Tested instance types for Azure on 64-bit ARM infrastructures

The following Microsoft Azure ARM64 instance types have been tested with OpenShift Container Platform.

Example 3.10. Machine types based on 64-bit ARM architecture

- **standardBpsv2Family**
- **standardDPSv5Family**
- **standardDPDSv5Family**
- **standardDPLDSv5Family**
- **standardDPLSv5Family**
- **standardEPSv5Family**
- **standardEPDSv5Family**
- **StandardDpdsv6Family**
- **StandardDpldsv6Famil**
- **StandardDplsv6Family**
- **StandardDpsv6Family**
- **StandardEpdsv6Family**
- **StandardEpsv6Family**

3.7.3.4. Enabling trusted launch for Azure VMs

You can enable two trusted launch features when installing your cluster on Azure: secure boot and virtualized Trusted Platform Modules.

For more information about the sizes of virtual machines that support the trusted launch features, see Virtual machine sizes.

IMPORTANT

Trusted launch is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see Technology Preview Features Support Scope .

Prerequisites

- You have created an **install-config.yaml** file.

Procedure

- Edit the **install-config.yaml** file before deploying your cluster:
  - Enable trusted launch only on control plane by adding the following stanza:

```
controlPlane:
  platform:
    azure:
      settings:
        securityType: TrustedLaunch
        trustedLaunch:
          uefiSettings:
            secureBoot: Enabled
            virtualizedTrustedPlatformModule: Enabled
```

  - Enable trusted launch only on compute node by adding the following stanza:

```
compute:
  platform:
    azure:
      settings:
        securityType: TrustedLaunch
        trustedLaunch:
          uefiSettings:
            secureBoot: Enabled
            virtualizedTrustedPlatformModule: Enabled
```

  - Enable trusted launch on all nodes by adding the following stanza:

```
platform:
  azure:
    settings:
      securityType: TrustedLaunch
      trustedLaunch:
        uefiSettings:
          secureBoot: Enabled
          virtualizedTrustedPlatformModule: Enabled
```

3.7.3.5. Enabling confidential VMs

You can enable confidential VMs when installing your cluster. You can enable confidential VMs for compute nodes, control plane nodes, or all nodes.

IMPORTANT

Using confidential VMs is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see Technology Preview Features Support Scope .

You can use confidential VMs with the following VM sizes:

- DCasv5-series
- DCadsv5-series
- ECasv5-series
- ECadsv5-series

IMPORTANT

Confidential VMs are currently not supported on 64-bit ARM architectures.

Prerequisites

- You have created an **install-config.yaml** file.

Procedure

- Edit the **install-config.yaml** file before deploying your cluster:
  - Enable confidential VMs only on control plane by adding the following stanza:

```
controlPlane:
  platform:
    azure:
      settings:
        securityType: ConfidentialVM
        confidentialVM:
          uefiSettings:
            secureBoot: Enabled
            virtualizedTrustedPlatformModule: Enabled
      osDisk:
        securityProfile:
          securityEncryptionType: VMGuestStateOnly
```

  - Enable confidential VMs only on compute nodes by adding the following stanza:

```
compute:
  platform:
    azure:
      settings:
        securityType: ConfidentialVM
        confidentialVM:
          uefiSettings:
            secureBoot: Enabled
            virtualizedTrustedPlatformModule: Enabled
      osDisk:
        securityProfile:
          securityEncryptionType: VMGuestStateOnly
```

  - Enable confidential VMs on all nodes by adding the following stanza:

```
platform:
```

```
azure:
  settings:
    securityType: ConfidentialVM
    confidentialVM:
      uefiSettings:
        secureBoot: Enabled
        virtualizedTrustedPlatformModule: Enabled
  osDisk:
    securityProfile:
      securityEncryptionType: VMGuestStateOnly
```

3.7.3.6. Sample customized install-config.yaml file for Azure

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster’s platform or modify the values of the required parameters.

IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```
apiVersion: v1
baseDomain: example.com 1
controlPlane: 2
  hyperthreading: Enabled 3 4
  name: master
  platform:
    azure:
      encryptionAtHost: true
      ultraSSDCapability: Enabled
      osDisk:
        diskSizeGB: 1024 5
        diskType: Premium_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      type: Standard_D8s_v3
  replicas: 3
compute: 6
- hyperthreading: Enabled 7 8
  name: worker
  platform:
    azure:
      ultraSSDCapability: Enabled
      type: Standard_D2s_v3
      encryptionAtHost: true
      osDisk:
        diskSizeGB: 512 9
        diskType: Standard_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      zones: 10
      - "1"
      - "2"
      - "3"
  replicas: 5
metadata:
  name: test-cluster 11
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 12
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    defaultMachinePlatform:
      osImage: 13
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      ultraSSDCapability: Enabled
    baseDomainResourceGroupName: resource_group 14
    region: centralus 15
    resourceGroupName: existing_resource_group 16
    networkResourceGroupName: vnet_resource_group 17
    virtualNetwork: vnet 18
    controlPlaneSubnet: control_plane_subnet 19
    computeSubnet: compute_subnet 20
    outboundType: UserDefinedRouting 21
    cloudName: AzurePublicCloud
pullSecret: '{"auths": ...}' 22
fips: false 23
sshKey: ssh-ed25519 AAAA... 24
publish: Internal 25
```

1 11 15 22 Required. The installation program prompts you for this value.

2 6 If you do not provide these parameters and values, the installation program provides the default value.

3 7 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute**

4 8 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

> IMPORTANT
>
> If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger virtual machine types, such as **Standard_D8s_v3**, for your machines if you disable simultaneous multithreading.

5 9 You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes is 1024 GB.

10 Specify a list of zones to deploy your machines to. For high availability, specify at least two zones.

12 The cluster network plugin to install. The default value **OVNKubernetes** is the only supported value.

13 Optional: A custom Red Hat Enterprise Linux CoreOS (RHCOS) image that should be used to boot control plane and compute machines. The **publisher**, **offer**, **sku**, and **version** parameters under **platform.azure.defaultMachinePlatform.osImage** apply to both control plane and compute machines. If the parameters under **controlPlane.platform.azure.osImage** or **compute.platform.azure.osImage** are set, they override the **platform.azure.defaultMachinePlatform.osImage** parameters.

14 Specify the name of the resource group that contains the DNS zone for your base domain.

16 Specify the name of an already existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.

17 If you use an existing VNet, specify the name of the resource group that contains it.

18 If you use an existing VNet, specify its name.

19 If you use an existing VNet, specify the name of the subnet to host the control plane machines.

20 If you use an existing VNet, specify the name of the subnet to host the compute machines.

21 You can customize your own outbound routing. Configuring user-defined routing prevents exposing external endpoints in your cluster. User-defined routing for egress requires deploying your cluster to an existing VNet.

23 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

IMPORTANT

To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Switching RHEL to FIPS mode.

When running Red Hat Enterprise Linux (RHEL) or Red Hat Enterprise Linux CoreOS (RHCOS) booted in FIPS mode, OpenShift Container Platform core components use the RHEL cryptographic libraries that have been submitted to NIST for FIPS 140-2/140-3 Validation on only the x86_64, ppc64le, and s390x architectures.

24 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

25 How to publish the user-facing endpoints of your cluster. Set **publish** to **Internal** to deploy a private cluster, which cannot be accessed from the internet. The default value is **External**.

3.7.3.7. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.

NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.

2. A proxy URL to use for creating HTTPS connections outside the cluster.

3. A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use * to bypass the proxy for all destinations.

4. If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

5. Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

NOTE

The installation program does not support the proxy **readinessEndpoints** field.

NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster** **Proxy** object is still created, but it will have a nil **spec**.

NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

Additional resources

- For more details about Accelerated Networking, see Accelerated Networking for Microsoft Azure VMs.

3.7.4. Alternatives to storing administrator-level secrets in the kube-system project

By default, administrator secrets are stored in the **kube-system** project. If you configured the **credentialsMode** parameter in the **install-config.yaml** file to **Manual**, you must use one of the following alternatives:

- To manage long-term cloud credentials manually, follow the procedure in Manually creating long-term credentials.

- To implement short-term credentials that are managed outside the cluster for individual components, follow the procedures in Configuring an Azure cluster to use short-term credentials.

3.7.4.1. Manually creating long-term credentials

The Cloud Credential Operator (CCO) can be put into manual mode prior to installation in environments where the cloud identity and access management (IAM) APIs are not reachable, or the administrator prefers not to store an administrator-level credential secret in the cluster **kube-system** namespace.

Procedure

1. If you did not set the **credentialsMode** parameter in the **install-config.yaml** configuration file to **Manual**, modify the value as shown:

   Sample configuration file snippet

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. If you have not previously created installation manifest files, do so by running the following command:

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   where **<installation_directory>** is the directory in which the installation program creates files.

3. Set a **$RELEASE_IMAGE** variable with the release image from your installation file by running the following command:

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

4. Extract the list of **CredentialsRequest** custom resources (CRs) from the OpenShift Container Platform release image by running the following command:

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1 The **--included** parameter includes only the manifests that your specific cluster configuration requires.

2 Specify the location of the **install-config.yaml** file.

3 Specify the path to the directory where you want to store the **CredentialsRequest** objects. If the specified directory does not exist, this command creates it.

This command creates a YAML file for each **CredentialsRequest** object.

Sample CredentialsRequest object

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
  ...
```

5. Create YAML files for secrets in the **openshift-install** manifests directory that you generated previously. The secrets must be stored using the namespace and secret name defined in the **spec.secretRef** for each **CredentialsRequest** object.

Sample CredentialsRequest object with secrets

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
    ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Sample Secret object

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  azure_subscription_id: <base64_encoded_azure_subscription_id>
  azure_client_id: <base64_encoded_azure_client_id>
  azure_client_secret: <base64_encoded_azure_client_secret>
  azure_tenant_id: <base64_encoded_azure_tenant_id>
  azure_resource_prefix: <base64_encoded_azure_resource_prefix>
  azure_resourcegroup: <base64_encoded_azure_resourcegroup>
  azure_region: <base64_encoded_azure_region>
```

IMPORTANT

Before upgrading a cluster that uses manually maintained credentials, you must ensure that the CCO is in an upgradeable state.

3.7.4.2. Configuring an Azure cluster to use short-term credentials

To install a cluster that uses Microsoft Entra Workload ID, you must configure the Cloud Credential Operator utility and create the required Azure resources for your cluster.

3.7.4.2.1. Configuring the Cloud Credential Operator utility

To create and manage cloud credentials from outside of the cluster when the Cloud Credential Operator (CCO) is operating in manual mode, extract and prepare the CCO utility (**ccoctl**) binary.

NOTE

The **ccoctl** utility is a Linux binary that must run in a Linux environment.

Prerequisites

- You have access to an OpenShift Container Platform account with cluster administrator access.
- You have installed the OpenShift CLI (**oc**).
- You have created a global Azure account for the **ccoctl** utility to use with the following permissions:
  - **Microsoft.Resources/subscriptions/resourceGroups/read**
  - **Microsoft.Resources/subscriptions/resourceGroups/write**

- **Microsoft.Resources/subscriptions/resourceGroups/delete**
- **Microsoft.Authorization/roleAssignments/read**
- **Microsoft.Authorization/roleAssignments/delete**
- **Microsoft.Authorization/roleAssignments/write**
- **Microsoft.Authorization/roleDefinitions/read**
- **Microsoft.Authorization/roleDefinitions/write**
- **Microsoft.Authorization/roleDefinitions/delete**
- **Microsoft.Storage/storageAccounts/listkeys/action**
- **Microsoft.Storage/storageAccounts/delete**
- **Microsoft.Storage/storageAccounts/read**
- **Microsoft.Storage/storageAccounts/write**
- **Microsoft.Storage/storageAccounts/blobServices/containers/delete**
- **Microsoft.Storage/storageAccounts/blobServices/containers/read**
- **Microsoft.Storage/storageAccounts/blobServices/containers/write**
- **Microsoft.ManagedIdentity/userAssignedIdentities/delete**
- **Microsoft.ManagedIdentity/userAssignedIdentities/read**
- **Microsoft.ManagedIdentity/userAssignedIdentities/write**
- **Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/read**
- **Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/write**
- **Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/delete**
- **Microsoft.Storage/register/action**
- **Microsoft.ManagedIdentity/register/action**

Procedure

1. Set a variable for the OpenShift Container Platform release image by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. Obtain the CCO container image from the OpenShift Container Platform release image by running the following command:

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

NOTE

Ensure that the architecture of the **$RELEASE_IMAGE** matches the architecture of the environment in which you will use the **ccoctl** tool.

3. Extract the **ccoctl** binary from the CCO container image within the OpenShift Container Platform release image by running the following command:

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 For **<rhel_version>**, specify the value that corresponds to the version of Red Hat Enterprise Linux (RHEL) that the host uses. If no value is specified, **ccoctl.rhel8** is used by default. The following values are valid:

- **rhel8**: Specify this value for hosts that use RHEL 8.
- **rhel9**: Specify this value for hosts that use RHEL 9.

4. Change the permissions to make **ccoctl** executable by running the following command:

```
$ chmod 775 ccoctl.<rhel_version>
```

Verification

- To verify that **ccoctl** is ready to use, display the help file. Use a relative file name when you run the command, for example:

```
$ ./ccoctl.rhel9
```

Example output

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws          Manage credentials objects for AWS cloud
  azure        Manage credentials objects for Azure
  gcp          Manage credentials objects for Google cloud
  help         Help about any command
  ibmcloud     Manage credentials objects for {ibm-cloud-title}
  nutanix      Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

3.7.4.2.2. Creating Azure resources with the Cloud Credential Operator utility

You can use the **ccoctl azure create-all** command to automate the creation of Azure resources.

NOTE

By default, **ccoctl** creates objects in the directory in which the commands are run. To create the objects in a different directory, use the **--output-dir** flag. This procedure uses **<path_to_ccoctl_output_dir>** to refer to this directory.

Prerequisites

You must have:

- Extracted and prepared the **ccoctl** binary.
- Access to your Microsoft Azure account by using the Azure CLI.

Procedure

1. Set a **$RELEASE_IMAGE** variable with the release image from your installation file by running the following command:

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. Extract the list of **CredentialsRequest** objects from the OpenShift Container Platform release image by running the following command:

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1 The **--included** parameter includes only the manifests that your specific cluster configuration requires.

2 Specify the location of the **install-config.yaml** file.

3 Specify the path to the directory where you want to store the **CredentialsRequest** objects. If the specified directory does not exist, this command creates it.

NOTE

This command might take a few moments to run.

3. To enable the **ccoctl** utility to detect your Azure credentials automatically, log in to the Azure CLI by running the following command:

```
$ az login
```

4. Use the **ccoctl** tool to process all **CredentialsRequest** objects by running the following command:

```
$ ccoctl azure create-all \
  --name=<azure_infra_name> \ ①
  --output-dir=<ccoctl_output_dir> \ ②
  --region=<azure_region> \ ③
  --subscription-id=<azure_subscription_id> \ ④
  --credentials-requests-dir=<path_to_credentials_requests_directory> \ ⑤
  --dnszone-resource-group-name=<azure_dns_zone_resource_group_name> \ ⑥
  --tenant-id=<azure_tenant_id> ⑦
```

1. Specify the user-defined name for all created Azure resources used for tracking.

2. Optional: Specify the directory in which you want the **ccoctl** utility to create objects. By default, the utility creates objects in the directory in which the commands are run.

3. Specify the Azure region in which cloud resources will be created.

4. Specify the Azure subscription ID to use.

5. Specify the directory containing the files for the component **CredentialsRequest** objects.

6. Specify the name of the resource group containing the cluster's base domain Azure DNS zone.

7. Specify the Azure tenant ID to use.

NOTE

If your cluster uses Technology Preview features that are enabled by the **TechPreviewNoUpgrade** feature set, you must include the **--enable-tech-preview** parameter.

To see additional optional parameters and explanations of how to use them, run the **azure create-all --help** command.

Verification

- To verify that the OpenShift Container Platform secrets are created, list the files in the **<path_to_ccoctl_output_dir>/manifests** directory:

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

Example output

```
azure-ad-pod-identity-webhook-config.yaml
cluster-authentication-02-config.yaml
openshift-cloud-controller-manager-azure-cloud-credentials-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capz-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-azure-disk-credentials-credentials.yaml
openshift-cluster-csi-drivers-azure-file-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-azure-cloud-credentials-credentials.yaml
```

You can verify that the Microsoft Entra ID service accounts are created by querying Azure. For more information, refer to Azure documentation on listing Entra ID service accounts.

3.7.4.2.3. Incorporating the Cloud Credential Operator utility manifests

To implement short-term security credentials managed outside the cluster for individual components, you must move the manifest files that the Cloud Credential Operator utility (**ccoctl**) created to the correct directories for the installation program.

Prerequisites

- You have configured an account with the cloud platform that hosts your cluster.
- You have configured the Cloud Credential Operator utility (**ccoctl**).
- You have created the cloud provider resources that are required for your cluster with the **ccoctl** utility.

Procedure

1. If you did not set the **credentialsMode** parameter in the **install-config.yaml** configuration file to **Manual**, modify the value as shown:

   Sample configuration file snippet

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. If you used the **ccoctl** utility to create a new Azure resource group instead of using an existing resource group, modify the **resourceGroupName** parameter in the **install-config.yaml** as shown:

   Sample configuration file snippet

   ```
   apiVersion: v1
   baseDomain: example.com
   # ...
   platform:
     azure:
       resourceGroupName: <azure_infra_name> 1
   # ...
   ```

   1 This value must match the user-defined name for Azure resources that was specified with the **--name** argument of the **ccoctl azure create-all** command.

3. If you have not previously created installation manifest files, do so by running the following command:

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   where **<installation_directory>** is the directory in which the installation program creates files.

4. Copy the manifests that the **ccoctl** utility generated to the **manifests** directory that the installation program created by running the following command:

   ```
   $ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
   ```

5. Copy the **tls** directory that contains the private key to the installation directory:

   ```
   $ cp -a /<path_to_ccoctl_output_dir>/tls .
   ```

3.7.5. Optional: Preparing a private Microsoft Azure cluster for a private image registry

By installing a private image registry on a private Microsoft Azure cluster, you can create private storage endpoints. Private storage endpoints disable public facing endpoints to the registry's storage account, adding an extra layer of security to your OpenShift Container Platform deployment.

> **IMPORTANT**
>
> Do not install a private image registry on Microsoft Azure Red Hat OpenShift (ARO), because the endpoint can put your Microsoft Azure Red Hat OpenShift cluster in an unrecoverable state.

Use the following guide to prepare your private Microsoft Azure cluster for installation with a private image registry.

Prerequisites

- You have access to an OpenShift Container Platform account with cluster administrator access.
- You have installed the OpenShift CLI (oc).
- You have prepared an **install-config.yaml** that includes the following information:
  - The **publish** field is set to **Internal**
- You have set the permissions for creating a private storage endpoint. For more information, see "Azure permissions for installer-provisioned infrastructure".

Procedure

1. If you have not previously created installation manifest files, do so by running the following command:

   ```
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   This command displays the following messages:

   Example output

   ```
   INFO Consuming Install Config from target directory
   INFO Manifests created in: <installation_directory>/manifests and
   <installation_directory>/openshift
   ```

2. Create an image registry configuration object and pass in the **networkResourceGroupName**, **subnetName**, and **vnetName** provided by Microsoft Azure. For example:

```
$ touch imageregistry-config.yaml
```

```
apiVersion: imageregistry.operator.openshift.io/v1
kind: Config
metadata:
  name: cluster
spec:
  managementState: "Managed"
  replicas: 2
  rolloutStrategy: RollingUpdate
  storage:
    azure:
      networkAccess:
        internal:
          networkResourceGroupName: <vnet_resource_group> 1
          subnetName: <subnet_name> 2
          vnetName: <vnet_name> 3
        type: Internal
```

1 Optional. If you have an existing VNet and subnet setup, replace **<vnet_resource_group>** with the resource group name that contains the existing virtual network (VNet).

2 Optional. If you have an existing VNet and subnet setup, replace **<subnet_name>** with the name of the existing compute subnet within the specified resource group.

3 Optional. If you have an existing VNet and subnet setup, replace **<vnet_name>** with the name of the existing virtual network (VNet) in the specified resource group.

NOTE

The **imageregistry-config.yaml** file is consumed during the installation process. If desired, you must back it up before installation.

3. Move the **imageregistry-config.yaml** file to the **<installation_directory/manifests>** folder by running the following command:

```
$ mv imageregistry-config.yaml <installation_directory/manifests/>
```

Next steps

- After you have moved the **imageregistry-config.yaml** file to the **<installation_directory/manifests>** folder and set the required permissions, proceed to "Deploying the cluster".

Additional resources

- For the list of permissions needed to create a private storage endpoint, see Required Azure permissions for installer-provisioned infrastructure.

3.7.6. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- You have configured an account with the cloud platform that hosts your cluster.
- You have the OpenShift Container Platform installation program and the pull secret for your cluster.
- You have an Azure subscription ID and tenant ID.
- If you are installing the cluster using a service principal, you have its application ID and password.
- If you are installing the cluster using a system-assigned managed identity, you have enabled it on the virtual machine that you will run the installation program from.
- If you are installing the cluster using a user-assigned managed identity, you have met these prerequisites:
    - You have its client ID.
    - You have assigned it to the virtual machine that you will run the installation program from.

Procedure

1. Optional: If you have run the installation program on this computer before, and want to use an alternative service principal or managed identity, go to the ~/**.azure**/ directory and delete the **osServicePrincipal.json** configuration file.
   Deleting this file prevents the installation program from automatically reusing subscription and authentication values from a previous installation.

2. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```
   $ ./openshift-install create cluster --dir <installation_directory> \ 1
       --log-level=info 2
   ```

   1 For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.

   2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

   If the installation program cannot locate the **osServicePrincipal.json** configuration file from a previous installation, you are prompted for Azure subscription and authentication values.

3. Enter the following Azure parameter values for your subscription:
    - **azure subscription id** Enter the subscription ID to use for the cluster.

- **azure tenant id** Enter the tenant ID.

4. Depending on the Azure identity you are using to deploy the cluster, do one of the following when prompted for the **azure service principal client id**
   - If you are using a service principal, enter its application ID.
   - If you are using a system-assigned managed identity, leave this value blank.
   - If you are using a user-assigned managed identity, specify its client ID.
5. Depending on the Azure identity you are using to deploy the cluster, do one of the following when prompted for the **azure service principal client secret**
   - If you are using a service principal, enter its password.
   - If you are using a system-assigned managed identity, leave this value blank.
   - If you are using a user-assigned managed identity,leave this value blank.

If previously not detected, the installation program creates an **osServicePrincipal.json** configuration file and stores this file in the ~/**.azure**/ directory on your computer. This ensures that the installation program can load the profile when it is creating an OpenShift Container Platform cluster on the target platform.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.
- Credential information also outputs to **<installation_directory>/.openshift_install.log**.

IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

3.7.7. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.

- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- See Accessing the web console for more details about accessing and understanding the OpenShift Container Platform web console.

3.7.8. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting .

3.8. INSTALLING A CLUSTER ON AZURE INTO A GOVERNMENT REGION

In OpenShift Container Platform version 4.18, you can install a cluster on Microsoft Azure into a government region. To configure the government region, you modify parameters in the **install-config.yaml** file before you install the cluster.

3.8.1. Azure government regions

OpenShift Container Platform supports deploying a cluster to Microsoft Azure Government (MAG) regions. MAG is specifically designed for US government agencies at the federal, state, and local level, as well as contractors, educational institutions, and other US customers that must run sensitive workloads on Azure. MAG is composed of government-only data center regions, all granted an Impact Level 5 Provisional Authorization.

Installing to a MAG region requires manually configuring the Azure Government dedicated cloud instance and region in the **install-config.yaml** file. You must also update your service principal to reference the appropriate government environment.

NOTE

The Azure government region cannot be selected using the guided terminal prompts from the installation program. You must define the region manually in the **install-config.yaml** file. Remember to also set the dedicated cloud instance, like **AzureUSGovernmentCloud**, based on the region specified.

3.8.2. Private clusters

You can deploy a private OpenShift Container Platform cluster that does not expose external endpoints. Private clusters are accessible from only an internal network and are not visible to the internet.

By default, OpenShift Container Platform is provisioned to use publicly-accessible DNS and endpoints. A private cluster sets the DNS, Ingress Controller, and API server to private when you deploy your cluster. This means that the cluster resources are only accessible from your internal network and are not visible to the internet.

IMPORTANT

If the cluster has any public subnets, load balancer services created by administrators might be publicly accessible. To ensure cluster security, verify that these services are explicitly annotated as private.

To deploy a private cluster, you must:

- Use existing networking that meets your requirements. Your cluster resources might be shared between other clusters on the network.
- Deploy from a machine that has access to:
  - The API services for the cloud to which you provision.

  - The API services for the cloud to which you provision.
  - The hosts on the network that you provision.
  - The internet to obtain installation media.

You can use any machine that meets these access requirements and follows your company's guidelines. For example, this machine can be a bastion host on your cloud network or a machine that has access to the network through a VPN.

3.8.2.1. Private clusters in Azure

To create a private cluster on Microsoft Azure, you must provide an existing private VNet and subnets to host the cluster. The installation program must also be able to resolve the DNS records that the cluster requires. The installation program configures the Ingress Operator and API server for only internal traffic.

Depending how your network connects to the private VNET, you might need to use a DNS forwarder to resolve the cluster's private DNS records. The cluster's machines use **168.63.129.16** internally for DNS resolution. For more information, see What is Azure Private DNS? and What is IP address 168.63.129.16? in the Azure documentation.

The cluster still requires access to internet to access the Azure APIs.

The following items are not required or created when you install a private cluster:

- A **BaseDomainResourceGroup**, since the cluster does not create public records
- Public IP addresses
- Public DNS records
- Public endpoints

> The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

3.8.2.1.1. Limitations

Private clusters on Azure are subject to only the limitations that are associated with the use of an existing VNet.

3.8.2.2. User-defined outbound routing

In OpenShift Container Platform, you can choose your own outbound routing for a cluster to connect to the internet. This allows you to skip the creation of public IP addresses and the public load balancer.

You can configure user-defined routing by modifying parameters in the **install-config.yaml** file before installing your cluster. A pre-existing VNet is required to use outbound routing when installing a cluster; the installation program is not responsible for configuring this.

When configuring a cluster to use user-defined routing, the installation program does not create the following resources:

- Outbound rules for access to the internet.

- Public IPs for the public load balancer.
- Kubernetes Service object to add the cluster machines to the public load balancer for outbound requests.

You must ensure the following items are available before setting user-defined routing:

- Egress to the internet is possible to pull container images, unless using an OpenShift image registry mirror.
- The cluster can access Azure APIs.
- Various allowlist endpoints are configured. You can reference these endpoints in the *Configuring your firewall* section.

There are several pre-existing networking setups that are supported for internet access using user-defined routing.

3.8.3. About reusing a VNet for your OpenShift Container Platform cluster

In OpenShift Container Platform 4.18, you can deploy a cluster into an existing Azure Virtual Network (VNet) in Microsoft Azure. If you do, you must also use existing subnets within the VNet and routing rules.

By deploying OpenShift Container Platform into an existing Azure VNet, you might be able to avoid service limit constraints in new accounts or more easily abide by the operational constraints that your company's guidelines set. This is a good option to use if you cannot obtain the infrastructure creation permissions that are required to create the VNet.

3.8.3.1. Requirements for using your VNet

When you deploy a cluster by using an existing VNet, you must perform additional network configuration before you install the cluster. In installer-provisioned infrastructure clusters, the installer usually creates the following components, but it does not create them when you install into an existing VNet:

- Subnets
- Route tables
- VNets
- Network Security Groups

> **NOTE**
>
> The installation program requires that you use the cloud-provided DNS server. Using a custom DNS server is not supported and causes the installation to fail.

If you use a custom VNet, you must correctly configure it and its subnets for the installation program and the cluster to use. The installation program cannot subdivide network ranges for the cluster to use, set route tables for the subnets, or set VNet options like DHCP, so you must do so before you install the cluster.

The cluster must be able to access the resource group that contains the existing VNet and subnets. While all of the resources that the cluster creates are placed in a separate resource group that it creates, some network resources are used from a separate group. Some cluster Operators must be able to access resources in both resource groups. For example, the Machine API controller attaches NICS for the virtual machines that it creates to subnets from the networking resource group.

Your VNet must meet the following characteristics:

- The VNet's CIDR block must contain the **Networking.MachineCIDR** range, which is the IP address pool for cluster machines.
- The VNet and its subnets must belong to the same resource group, and the subnets must be configured to use Azure-assigned DHCP IP addresses instead of static IP addresses.

You must provide two subnets within your VNet, one for the control plane machines and one for the compute machines. Because Azure distributes machines in different availability zones within the region that you specify, your cluster will have high availability by default.

NOTE

By default, if you specify availability zones in the **install-config.yaml** file, the installation program distributes the control plane machines and the compute machines across these availability zones within a region. To ensure high availability for your cluster, select a region with at least three availability zones. If your region contains fewer than three availability zones, the installation program places more than one control plane machine in the available zones.

To ensure that the subnets that you provide are suitable, the installation program confirms the following data:

- All the specified subnets exist.
- There are two private subnets, one for the control plane machines and one for the compute machines.
- The subnet CIDRs belong to the machine CIDR that you specified. Machines are not provisioned in availability zones that you do not provide private subnets for. If required, the installation program creates public load balancers that manage the control plane and worker nodes, and Azure allocates a public IP address to them.

NOTE

If you destroy a cluster that uses an existing VNet, the VNet is not deleted.

3.8.3.1.1. Network security group requirements

The network security groups for the subnets that host the compute and control plane machines require specific access to ensure that the cluster communication is correct. You must create rules to allow access to the required cluster communication ports.

IMPORTANT

The network security group rules must be in place before you install the cluster. If you attempt to install a cluster without the required access, the installation program cannot reach the Azure APIs, and installation fails.

Table 3.25. Required ports

Port	Description	Control plane	Compute
80	Allows HTTP traffic		x
443	Allows HTTPS traffic		x
6443	Allows communication to the control plane machines	x	
22623	Allows internal communication to the machine config server for provisioning machines	x	

1. If you are using Azure Firewall to restrict the internet access, then you can configure Azure Firewall to allow the Azure APIs. A network security group rule is not needed. For more information, see "Configuring your firewall" in "Additional resources".

IMPORTANT

Currently, there is no supported way to block or restrict the machine config server endpoint. The machine config server must be exposed to the network so that newly-provisioned machines, which have no existing configuration or state, are able to fetch their configuration. In this model, the root of trust is the certificate signing requests (CSR) endpoint, which is where the kubelet sends its certificate signing request for approval to join the cluster. Because of this, machine configs should not be used to distribute sensitive information, such as secrets and certificates.

To ensure that the machine config server endpoints, ports 22623 and 22624, are secured in bare metal scenarios, customers must configure proper network policies.

Because cluster components do not modify the user-provided network security groups, which the Kubernetes controllers update, a pseudo-network security group is created for the Kubernetes controller to modify without impacting the rest of the environment.

Table 3.26. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	**1936**	Metrics
	9000-**9999**	Host level services, including the node exporter on ports **9100**-**9101** and the Cluster Version Operator on port**9099**.
	10250-**10259**	The default ports that Kubernetes reserves
UDP	**4789**	VXLAN
	6081	Geneve

Protocol	Port	Description
	9000-9999	Host level services, including the node exporter on ports **9100-9101**.
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port **123** If you configure an external NTP time server, you must open UDP port **123**.
TCP/UDP	**30000-32767**	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 3.27. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	**2379-2380**	etcd server and peer ports

3.8.3.2. Division of permissions

Starting with OpenShift Container Platform 4.3, you do not need all of the permissions that are required for an installation program-provisioned infrastructure cluster to deploy a cluster. This change mimics the division of permissions that you might have at your company: some individuals can create different resources in your clouds than others. For example, you might be able to create application-specific items, like instances, storage, and load balancers, but not networking-related components such as VNets, subnet, or ingress rules.

The Azure credentials that you use when you create your cluster do not need the networking permissions that are required to make VNets and core networking components within the VNet, such as subnets, routing tables, internet gateways, NAT, and VPN. You still need permission to make the application resources that the machines within the cluster require, such as load balancers, security groups, storage accounts, and nodes.

3.8.3.3. Isolation between clusters

Because the cluster is unable to modify network security groups in an existing subnet, there is no way to isolate clusters from each other on the VNet.

Additional resources

- About the OVN-Kubernetes network plugin
- Configuring your firewall

3.8.4. Manually creating the installation configuration file

Installing the cluster requires that you manually create the installation configuration file.

Prerequisites

- You have an SSH public key on your local machine to provide to the installation program. The key will be used for SSH authentication onto your cluster nodes for debugging and disaster recovery.
- You have obtained the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Create an installation directory to store your required installation assets in:

```
$ mkdir <installation_directory>
```

IMPORTANT

You must create a directory. Some installation assets, like bootstrap X.509 certificates have short expiration intervals, so you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

2. Customize the sample **install-config.yaml** file template that is provided and save it in the **<installation_directory>**.

NOTE

You must name this configuration file **install-config.yaml**.

3. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

IMPORTANT

The **install-config.yaml** file is consumed during the next step of the installation process. You must back it up now.

Additional resources

- Installation configuration parameters for Azure

3.8.4.1. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 3.28. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or Hyper-Threading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

NOTE

For OpenShift Container Platform version 4.18, RHCOS is based on RHEL version 9.4, which updates the micro-architecture requirements. The following list contains the minimum instruction set architectures (ISA) that each architecture requires:

- x86-64 architecture requires x86-64-v2 ISA
- ARM64 architecture requires ARMv8.0-A ISA
- IBM Power architecture requires Power 9 ISA
- s390x architecture requires z14 ISA

For more information, see Architectures (RHEL documentation).

IMPORTANT

You are required to use Azure virtual machines that have the **premiumIO** parameter set to **true**.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- Optimizing storage

3.8.4.2. Tested instance types for Azure

The following Microsoft Azure instance types have been tested with OpenShift Container Platform.

Example 3.11. Machine types based on 64-bit x86 architecture

- **standardBasv2Family**
- **standardBSFamily**
- **standardBsv2Family**
- **standardDADSv5Family**
- **standardDASv4Family**
- **standardDASv5Family**
- **standardDCACCV5Family**
- **standardDCADCCV5Family**
- **standardDCADSv5Family**
- **standardDCASv5Family**
- **standardDCSv3Family**
- **standardDCSv2Family**
- **standardDDCSv3Family**
- **standardDDSv4Family**
- **standardDDSv5Family**
- **standardDLDSv5Family**
- **standardDLSv5Family**
- **standardDSFamily**
- **standardDSv2Family**
- **standardDSv2PromoFamily**
- **standardDSv3Family**
- **standardDSv4Family**
- **standardDSv5Family**
- **standardEADSv5Family**
- **standardEASv4Family**

- standardEASv5Family
- standardEBDSv5Family
- standardEBSv5Family
- standardECACCV5Family
- standardECADCCV5Family
- standardECADSv5Family
- standardECASv5Family
- standardEDSv4Family
- standardEDSv5Family
- standardEIADSv5Family
- standardEIASv4Family
- standardEIASv5Family
- standardEIBDSv5Family
- standardEIBSv5Family
- standardEIDSv5Family
- standardEISv3Family
- standardEISv5Family
- standardESv3Family
- standardESv4Family
- standardESv5Family
- standardFXMDVSFamily
- standardFSFamily
- standardFSv2Family
- standardGSFamily
- standardHBrsv2Family
- standardHBSFamily
- standardHBv4Family
- standardHCSFamily
- standardHXFamily

- **standardLASv3Family**
- **standardLSFamily**
- **standardLSv2Family**
- **standardLSv3Family**
- **standardMDSHighMemoryv3Family**
- **standardMDSMediumMemoryv2Family**
- **standardMDSMediumMemoryv3Family**
- **standardMIDSHighMemoryv3Family**
- **standardMIDSMediumMemoryv2Family**
- **standardMISHighMemoryv3Family**
- **standardMISMediumMemoryv2Family**
- **standardMSFamily**
- **standardMSHighMemoryv3Family**
- **standardMSMediumMemoryv2Family**
- **standardMSMediumMemoryv3Family**
- **StandardNCADSA100v4Family**
- **Standard NCASv3_T4 Family**
- **standardNCSv3Family**
- **standardNDSv2Family**
- **StandardNGADSV620v1Family**
- **standardNPSFamily**
- **StandardNVADSA10v5Family**
- **standardNVSv3Family**
- **standardXEISv4Family**

3.8.4.3. Enabling trusted launch for Azure VMs

You can enable two trusted launch features when installing your cluster on Azure: secure boot and virtualized Trusted Platform Modules.

For more information about the sizes of virtual machines that support the trusted launch features, see Virtual machine sizes.

IMPORTANT

Trusted launch is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see Technology Preview Features Support Scope .

Prerequisites

- You have created an **install-config.yaml** file.

Procedure

- Edit the **install-config.yaml** file before deploying your cluster:
  - Enable trusted launch only on control plane by adding the following stanza:

    ```
    controlPlane:
      platform:
        azure:
          settings:
            securityType: TrustedLaunch
            trustedLaunch:
              uefiSettings:
                secureBoot: Enabled
                virtualizedTrustedPlatformModule: Enabled
    ```

  - Enable trusted launch only on compute node by adding the following stanza:

    ```
    compute:
      platform:
        azure:
          settings:
            securityType: TrustedLaunch
            trustedLaunch:
              uefiSettings:
                secureBoot: Enabled
                virtualizedTrustedPlatformModule: Enabled
    ```

  - Enable trusted launch on all nodes by adding the following stanza:

    ```
    platform:
      azure:
        settings:
          securityType: TrustedLaunch
          trustedLaunch:
            uefiSettings:
              secureBoot: Enabled
              virtualizedTrustedPlatformModule: Enabled
    ```

3.8.4.4. Enabling confidential VMs

You can enable confidential VMs when installing your cluster. You can enable confidential VMs for compute nodes, control plane nodes, or all nodes.

IMPORTANT

Using confidential VMs is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see Technology Preview Features Support Scope .

You can use confidential VMs with the following VM sizes:

- DCasv5-series
- DCadsv5-series
- ECasv5-series
- ECadsv5-series

IMPORTANT

Confidential VMs are currently not supported on 64-bit ARM architectures.

Prerequisites

- You have created an **install-config.yaml** file.

Procedure

- Edit the **install-config.yaml** file before deploying your cluster:
  - Enable confidential VMs only on control plane by adding the following stanza:

```
controlPlane:
  platform:
    azure:
      settings:
        securityType: ConfidentialVM
        confidentialVM:
          uefiSettings:
            secureBoot: Enabled
            virtualizedTrustedPlatformModule: Enabled
      osDisk:
        securityProfile:
          securityEncryptionType: VMGuestStateOnly
```

  - Enable confidential VMs only on compute nodes by adding the following stanza:

```
compute:
  platform:
    azure:
      settings:
        securityType: ConfidentialVM
        confidentialVM:
          uefiSettings:
            secureBoot: Enabled
            virtualizedTrustedPlatformModule: Enabled
      osDisk:
        securityProfile:
          securityEncryptionType: VMGuestStateOnly
```

- Enable confidential VMs on all nodes by adding the following stanza:

```
platform:
  azure:
    settings:
      securityType: ConfidentialVM
      confidentialVM:
        uefiSettings:
          secureBoot: Enabled
          virtualizedTrustedPlatformModule: Enabled
    osDisk:
      securityProfile:
        securityEncryptionType: VMGuestStateOnly
```

3.8.4.5. Sample customized install-config.yaml file for Azure

You can customize the **install-config.yaml** file to specify more details about your OpenShift Container Platform cluster's platform or modify the values of the required parameters.

IMPORTANT

This sample YAML file is provided for reference only. You must obtain your **install-config.yaml** file by using the installation program and modify it.

```
apiVersion: v1
baseDomain: example.com 1
controlPlane: 2
  hyperthreading: Enabled 3 4
  name: master
  platform:
    azure:
      encryptionAtHost: true
      ultraSSDCapability: Enabled
      osDisk:
        diskSizeGB: 1024 5
        diskType: Premium_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      type: Standard_D8s_v3
  replicas: 3
compute: 6
- hyperthreading: Enabled 7 8
  name: worker
  platform:
    azure:
      ultraSSDCapability: Enabled
      type: Standard_D2s_v3
      encryptionAtHost: true
      osDisk:
        diskSizeGB: 512 9
        diskType: Standard_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      zones: 10
      - "1"
      - "2"
      - "3"
  replicas: 5
metadata:
  name: test-cluster 11
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 12
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    defaultMachinePlatform:
      osImage: 13
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      ultraSSDCapability: Enabled
    baseDomainResourceGroupName: resource_group 14
    region: usgovvirginia
    resourceGroupName: existing_resource_group 15
    networkResourceGroupName: vnet_resource_group 16
    virtualNetwork: vnet 17
    controlPlaneSubnet: control_plane_subnet 18
    computeSubnet: compute_subnet 19
    outboundType: UserDefinedRouting 20
    cloudName: AzureUSGovernmentCloud 21
pullSecret: '{"auths": ...}' 22
fips: false 23
sshKey: ssh-ed25519 AAAA... 24
publish: Internal 25
```

1 11 22 Required.

2 6 If you do not provide these parameters and values, the installation program provides the default value.

3 7 The **controlPlane** section is a single mapping, but the **compute** section is a sequence of mappings. To meet the requirements of the different data structures, the first line of the **compute** section must begin with a hyphen, **-**, and the first line of the **controlPlane** section must not. Only one control plane pool is used.

4 8 Whether to enable or disable simultaneous multithreading, or **hyperthreading**. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. You can disable it by setting the parameter value to **Disabled**. If you disable simultaneous multithreading in some cluster machines, you must disable it in all cluster machines.

IMPORTANT

If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance. Use larger virtual machine types, such as **Standard_D8s_v3**, for your machines if you disable simultaneous multithreading.

5 9 You can specify the size of the disk to use in GB. Minimum recommendation for control plane nodes is 1024 GB.

10 Specify a list of zones to deploy your machines to. For high availability, specify at least two zones.

12 The cluster network plugin to install. The default value **OVNKubernetes** is the only supported value.

13 Optional: A custom Red Hat Enterprise Linux CoreOS (RHCOS) image that should be used to boot control plane and compute machines. The **publisher**, **offer**, **sku**, and **version** parameters under **platform.azure.defaultMachinePlatform.osImage** apply to both control plane and compute machines. If the parameters under **controlPlane.platform.azure.osImage** or **compute.platform.azure.osImage** are set, they override the **platform.azure.defaultMachinePlatform.osImage** parameters.

14 Specify the name of the resource group that contains the DNS zone for your base domain.

15 Specify the name of an already existing resource group to install your cluster to. If undefined, a new resource group is created for the cluster.

16 If you use an existing VNet, specify the name of the resource group that contains it.

17 If you use an existing VNet, specify its name.

18 If you use an existing VNet, specify the name of the subnet to host the control plane machines.

19 If you use an existing VNet, specify the name of the subnet to host the compute machines.

20 You can customize your own outbound routing. Configuring user-defined routing prevents exposing external endpoints in your cluster. User-defined routing for egress requires deploying your cluster to an existing VNet.

21 Specify the name of the Azure cloud environment to deploy your cluster to. Set **AzureUSGovernmentCloud** to deploy to a Microsoft Azure Government (MAG) region. The default value is **AzurePublicCloud**.

23 Whether to enable or disable FIPS mode. By default, FIPS mode is not enabled. If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead.

> IMPORTANT
>
> To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Switching RHEL to FIPS mode.
>
> When running Red Hat Enterprise Linux (RHEL) or Red Hat Enterprise Linux CoreOS (RHCOS) booted in FIPS mode, OpenShift Container Platform core components use the RHEL cryptographic libraries that have been submitted to NIST for FIPS 140-2/140-3 Validation on only the x86_64, ppc64le, and s390x architectures.

24 You can optionally provide the **sshKey** value that you use to access the machines in your cluster.

> NOTE
>
> For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

25 How to publish the user-facing endpoints of your cluster. Set **publish** to **Internal** to deploy a private cluster, which cannot be accessed from the internet. The default value is **External**.

3.8.4.6. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of

them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.

NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.

2 A proxy URL to use for creating HTTPS connections outside the cluster.

3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use * to bypass the proxy for all destinations.

4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

NOTE

The installation program does not support the proxy **readinessEndpoints** field.

NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

Additional resources

- For more details about Accelerated Networking, see Accelerated Networking for Microsoft Azure VMs.

3.8.5. Deploying the cluster

You can install OpenShift Container Platform on a compatible cloud platform.

IMPORTANT

You can run the **create cluster** command of the installation program only once, during initial installation.

Prerequisites

- You have configured an account with the cloud platform that hosts your cluster.
- You have the OpenShift Container Platform installation program and the pull secret for your cluster.
- You have an Azure subscription ID and tenant ID.
- If you are installing the cluster using a service principal, you have its application ID and password.
- If you are installing the cluster using a system-assigned managed identity, you have enabled it on the virtual machine that you will run the installation program from.
- If you are installing the cluster using a user-assigned managed identity, you have met these prerequisites:
  - You have its client ID.

- ○ You have assigned it to the virtual machine that you will run the installation program from.

Procedure

1. Optional: If you have run the installation program on this computer before, and want to use an alternative service principal or managed identity, go to the ~/**.azure**/ directory and delete the **osServicePrincipal.json** configuration file.
   Deleting this file prevents the installation program from automatically reusing subscription and authentication values from a previous installation.

2. Change to the directory that contains the installation program and initialize the cluster deployment:

   ```
   $ ./openshift-install create cluster --dir <installation_directory> \ ①
       --log-level=info ②
   ```

   ① For **<installation_directory>**, specify the location of your customized **./install-config.yaml** file.

   ② To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

   If the installation program cannot locate the **osServicePrincipal.json** configuration file from a previous installation, you are prompted for Azure subscription and authentication values.

3. Enter the following Azure parameter values for your subscription:
   - **azure subscription id** Enter the subscription ID to use for the cluster.
   - **azure tenant id** Enter the tenant ID.

4. Depending on the Azure identity you are using to deploy the cluster, do one of the following when prompted for the **azure service principal client id**
   - If you are using a service principal, enter its application ID.
   - If you are using a system-assigned managed identity, leave this value blank.
   - If you are using a user-assigned managed identity, specify its client ID.

5. Depending on the Azure identity you are using to deploy the cluster, do one of the following when prompted for the **azure service principal client secret**
   - If you are using a service principal, enter its password.
   - If you are using a system-assigned managed identity, leave this value blank.
   - If you are using a user-assigned managed identity,leave this value blank.

If previously not detected, the installation program creates an **osServicePrincipal.json** configuration file and stores this file in the ~/**.azure**/ directory on your computer. This ensures that the installation program can load the profile when it is creating an OpenShift Container Platform cluster on the target platform.

Verification

When the cluster deployment completes successfully:

- The terminal displays directions for accessing your cluster, including a link to the web console and credentials for the **kubeadmin** user.

- Credential information also outputs to **<installation_directory>/.openshift_install.log**.

IMPORTANT

Do not delete the installation program or the files that the installation program creates. Both are required to delete the cluster.

Example output

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

3.8.6. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.

- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

Additional resources

- See Accessing the web console for more details about accessing and understanding the OpenShift Container Platform web console.

3.8.7. Next steps

- Customize your cluster.
- If necessary, you can opt out of remote health reporting .

CHAPTER 4. USER-PROVISIONED INFRASTRUCTURE

4.1. PREPARING TO INSTALL A CLUSTER ON AZURE

To prepare for installation of an OpenShift Container Platform cluster on Azure, complete the following steps:

- You have selected a cluster installation method.
- You configured an Azure account to host the cluster and determined the tested and validated region to deploy the cluster to.
- If you use a firewall, you have configured it to allow the sites that your cluster requires access to.

4.1.1. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.18, you require access to the internet to install your cluster.

You must have internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

4.1.2. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the **~/.ssh/authorized_keys** list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The **./openshift-install gather** command also requires the SSH public key to be in place on the cluster nodes.

IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

NOTE

You must use a local key, not one that you configured with platform-specific approaches.

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
   ```

   1 Specify the path and file name, such as **~/.ssh/id_ed25519**, of the new SSH key. If you have an existing key pair, ensure your public key is in the your **~/.ssh** directory.

   NOTE

   If you plan to install an OpenShift Container Platform cluster that uses the RHEL cryptographic libraries that have been submitted to NIST for FIPS 140-2/140-3 Validation on only the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

   ```
   $ cat <path>/<file_name>.pub
   ```

   For example, run the following to view the **~/.ssh/id_ed25519.pub** public key:

   ```
   $ cat ~/.ssh/id_ed25519.pub
   ```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the **./openshift-install gather** command.

   NOTE

   On some distributions, default SSH private key identities such as **~/.ssh/id_rsa** and **~/.ssh/id_dsa** are managed automatically.

   a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

      ```
      $ eval "$(ssh-agent -s)"
      ```

      Example output

```
Agent pid 31874
```

NOTE

If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> 1
```

1 Specify the path and file name for your SSH private key, such as **~/.ssh/id_ed25519**

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program.

4.1.3. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Go to the Cluster Type page on the Red Hat Hybrid Cloud Console. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

   TIP

   You can also download the binaries for a specific OpenShift Container Platform release .

2. Select your infrastructure provider from the **Run it yourself** section of the page.

3. Select your host operating system and architecture from the dropdown menus under **OpenShift Installer** and click **Download Installer**.

4. Place the downloaded file in the directory where you want to store the installation configuration files.

IMPORTANT

- The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both of the files are required to delete the cluster.

- Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

5. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

6. Download your installation pull secret from Red Hat OpenShift Cluster Manager . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

TIP

Alternatively, you can retrieve the installation program from the Red Hat Customer Portal, where you can specify a version of the installation program to download. However, you must have an active subscription to access this page.

4.1.4. Installing the OpenShift CLI

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.

IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.18. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the OpenShift Container Platform downloads page on the Red Hat Customer Portal.

2. Select the architecture from the **Product Variant** drop-down list.

3. Select the appropriate version from the **Version** drop-down list.

4. Click **Download Now** next to the **OpenShift v4.18 Linux Clients** entry and save the file.

5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
   To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows
You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the OpenShift Container Platform downloads page on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.18 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
   To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS
You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the OpenShift Container Platform downloads page on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.18 macOS Clients** entry and save the file.

NOTE

For macOS arm64, choose the **OpenShift v4.18 macOS arm64 Client**entry.

4. Unpack and unzip the archive.

5. Move the **oc** binary to a directory on your PATH.
   To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- Verify your installation by using an **oc** command:

```
$ oc <command>
```

4.1.5. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.18, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- For more information about the Telemetry service, see About remote health monitoring.

4.2. INSTALLING A CLUSTER ON AZURE IN A DISCONNECTED ENVIRONMENT WITH USER-PROVISIONED INFRASTRUCTURE

In OpenShift Container Platform, you can install a cluster on Microsoft Azure by using infrastructure that you provide.

Several Azure Resource Manager (ARM) templates are provided to assist in completing these steps or to help model your own.

IMPORTANT

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the cloud provider and the installation process of OpenShift Container Platform. Several ARM templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods.

Prerequisites

- You reviewed details about the OpenShift Container Platform installation and update processes.

- You read the documentation on selecting a cluster installation method and preparing it for users.
- You configured an Azure account to host the cluster and determined the tested and validated region to deploy the cluster to.
- You mirrored the images for a disconnected installation to your registry and obtained the **imageContentSources** data for your version of OpenShift Container Platform.

  IMPORTANT

  Because the installation media is on the mirror host, you must use that computer to complete all installation steps.

- If you use a firewall, you configured it to allow the sites that your cluster requires access to.
- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, you have manually created long-term credentials.
- If you use customer-managed encryption keys, you prepared your Azure environment for encryption.

4.2.1. About installations in restricted networks

In OpenShift Container Platform 4.18, you can perform an installation that does not require an active connection to the internet to obtain software components. Restricted network installations can be completed using installer-provisioned infrastructure or user-provisioned infrastructure, depending on the cloud platform to which you are installing the cluster.

If you choose to perform a restricted network installation on a cloud platform, you still require access to its cloud APIs. Some cloud functions, like Amazon Web Service's Route 53 DNS and IAM services, require internet access. Depending on your network, you might require less internet access for an installation on bare metal hardware, Nutanix, or on VMware vSphere.

To complete a restricted network installation, you must create a registry that mirrors the contents of the OpenShift image registry and contains the installation media. You can create this registry on a mirror host, which can access both the internet and your closed network, or by using other methods that meet your restrictions.

IMPORTANT

Because of the complexity of the configuration for user-provisioned installations, consider completing a standard user-provisioned infrastructure installation before you attempt a restricted network installation using user-provisioned infrastructure. Completing this test installation might make it easier to isolate and troubleshoot any issues that might arise during your installation in a restricted network.

4.2.1.1. Additional limits

Clusters in restricted networks have the following additional limitations and restrictions:

- The **ClusterVersion** status includes an **Unable to retrieve available updates** error.

- By default, you cannot use the contents of the Developer Catalog because you cannot access the required image stream tags.

4.2.1.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.18, you require access to the internet to install your cluster.

You must have internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.
- Access Quay.io to obtain the packages that are required to install your cluster.
- Obtain the packages that are required to perform cluster updates.

IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

4.2.2. Configuring your Azure project

Before you can install OpenShift Container Platform, you must configure an Azure project to host it.

IMPORTANT

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see Resolve reserved resource name errors in the Azure documentation.

4.2.2.1. Azure account limits

The OpenShift Container Platform cluster uses a number of Microsoft Azure components, and the default Azure subscription and service limits, quotas, and constraints affect your ability to install OpenShift Container Platform clusters.

IMPORTANT

Default limits vary by offer category types, such as Free Trial and Pay-As-You-Go, and by series, such as Dv2, F, and G. For example, the default for Enterprise Agreement subscriptions is 350 cores.

Check the limits for your subscription type and if necessary, increase quota limits for your account before you install a default cluster on Azure.

The following table summarizes the Azure components whose limits can impact your ability to install and run OpenShift Container Platform clusters.

Component	Number of components required by default	Default Azure limit	Description
vCPU	44	20 per region	A default cluster requires 44 vCPUs, so you must increase the account limit. By default, each cluster creates the following instances: ● One bootstrap machine, which is removed after installation ● Three control plane machines ● Three compute machines Because the bootstrap and control plane machines use **Standard_D8s_v3** virtual machines, which use 8 vCPUs, and the compute machines use **Standard_D4s_v3** virtual machines, which use 4 vCPUs, a default cluster requires 44 vCPUs. The bootstrap node VM, which uses 8 vCPUs, is used only during installation. To deploy more worker nodes, enable autoscaling, deploy large workloads, or use a different instance type, you must further increase the vCPU limit for your account to ensure that your cluster can deploy the machines that you require.
OS Disk	7		Each cluster machine must have a minimum of 100 GB of storage and 300 IOPS. **NOTE** Faster storage is recommended for production clusters and clusters with intensive workloads. For more information about optimizing storage for performance, see the page titled "Optimizing storage" in the "Scalability and performance" section.
VNet	1	1000 per region	Each default cluster requires one Virtual Network (VNet), which contains two subnets.
Network interfaces	7	65,536 per region	Each default cluster requires seven network interfaces. If you create more machines or your deployed workloads create load balancers, your cluster uses more network interfaces.

<table>
<thead>
<tr><th>Component</th><th>Number of components required by default</th><th>Default Azure limit</th><th>Description</th></tr>
</thead>
<tbody>
<tr><td>Network security groups</td><td>2</td><td>5000</td><td>Each cluster creates network security groups for each subnet in the VNet. The default cluster creates network security groups for the control plane and for the compute node subnets:
<table>
<tr><td><b>controlplane</b></td><td>Allows the control plane machines to be reached on port 6443 from anywhere</td></tr>
<tr><td><b>node</b></td><td>Allows worker nodes to be reached from the internet on ports 80 and 443</td></tr>
</table>
</td></tr>
<tr><td>Network load balancers</td><td>3</td><td>1000 per region</td><td>Each cluster creates the following load balancers:
<table>
<tr><td><b>default</b></td><td>Public IP address that load balances requests to ports 80 and 443 across worker machines</td></tr>
<tr><td><b>internal</b></td><td>Private IP address that load balances requests to ports 6443 and 22623 across control plane machines</td></tr>
<tr><td><b>external</b></td><td>Public IP address that load balances requests to port 6443 across control plane machines</td></tr>
</table>
If your applications create more Kubernetes <b>LoadBalancer</b> service objects, your cluster uses more load balancers.</td></tr>
<tr><td>Public IP addresses</td><td>3</td><td></td><td>Each of the two public load balancers uses a public IP address. The bootstrap machine also uses a public IP address so that you can SSH into the machine to troubleshoot issues during installation. The IP address for the bootstrap node is used only during installation.</td></tr>
<tr><td>Private IP addresses</td><td>7</td><td></td><td>The internal load balancer, each of the three control plane machines, and each of the three worker machines each use a private IP address.</td></tr>
</tbody>
</table>

Component	Number of components required by default	Default Azure limit	Description
Spot VM vCPUs (optional)	0 If you configure spot VMs, your cluster must have two spot VM vCPUs for every compute node.	20 per region	This is an optional component. To use spot VMs, you must increase the Azure default limit to at least twice the number of compute nodes in your cluster. **NOTE** Using spot VMs for control plane nodes is not recommended.

To increase an account limit, file a support request on the Azure portal. For more information, see Request a quota limit increase for Azure Deployment Environments resources .

Additional resources

- Optimizing storage

4.2.2.2. Configuring a public DNS zone in Azure

To install OpenShift Container Platform, the Microsoft Azure account you use must have a dedicated public hosted DNS zone in your account. This zone must be authoritative for the domain. This service provides cluster DNS resolution and name lookup for external connections to the cluster.

Procedure

1. Identify your domain, or subdomain, and registrar. You can transfer an existing domain and registrar or obtain a new one through Azure or another source.
   - To purchase a new domain through Azure, see Buy a custom domain name for Azure App Service.
   - If you are using an existing domain and registrar, migrate its DNS to Azure. For more information, see Migrate an active DNS name to Azure App Service in the Azure documentation.
2. Configure DNS for your domain, which includes creating a public hosted zone for your domain or subdomain, extracting the new authoritative name servers, and updating the registrar records for the name servers that your domain uses. For more information, see Tutorial: Host your domain in Azure DNS.
   Use an appropriate root domain, such as **openshiftcorp.com**, or subdomain, such as **clusters.openshiftcorp.com**.
3. If you use a subdomain, follow your organization's procedures to add its delegation records to the parent domain.

You can view Azure's DNS solution by visiting this example for creating DNS zones .

4.2.2.3. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

4.2.2.4. Required Azure roles

Before you create the identity, verify that your environment meets the following requirements based on the identity:

- The Azure account that you use to create the identity is assigned the **User Access Administrator** and **Contributor** roles. These roles are required when:
  - Creating a service principal or user-assigned managed identity.
  - Enabling a system-assigned managed identity on a virtual machine.
- If you are going to use a service principal to complete the installation, verify that the Azure account that you use to create the identity is assigned the **microsoft.directory/servicePrincipals/createAsOwner** permission in Microsoft Entra ID.

To set roles on the Azure portal, see Assign Azure roles using the Azure portal in the Azure documentation.

4.2.2.5. Required Azure permissions for user-provisioned infrastructure

The installation program requires access to an Azure service principal or managed identity with the necessary permissions to deploy the cluster and to maintain its daily operation. These permissions must be granted to the Azure subscription that is associated with the identity.

The following options are available to you:

- You can assign the identity the **Contributor** and **User Access Administrator** roles. Assigning these roles is the quickest way to grant all of the required permissions.
  For more information about assigning roles, see the Azure documentation for managing access to Azure resources using the Azure portal.
- If your organization's security policies require a more restrictive set of permissions, you can create a custom role with the necessary permissions.

The following permissions are required for creating an OpenShift Container Platform cluster on Microsoft Azure.

> **Example 4.1. Required permissions for creating authorization resources**
>
> - **Microsoft.Authorization/policies/audit/action**
> - **Microsoft.Authorization/policies/auditIfNotExists/action**
> - **Microsoft.Authorization/roleAssignments/read**
> - **Microsoft.Authorization/roleAssignments/write**

Example 4.2. Required permissions for creating compute resources

- **Microsoft.Compute/images/read**
- **Microsoft.Compute/images/write**
- **Microsoft.Compute/images/delete**
- **Microsoft.Compute/availabilitySets/read**
- **Microsoft.Compute/disks/beginGetAccess/action**
- **Microsoft.Compute/disks/delete**
- **Microsoft.Compute/disks/read**
- **Microsoft.Compute/disks/write**
- **Microsoft.Compute/galleries/images/read**
- **Microsoft.Compute/galleries/images/versions/read**
- **Microsoft.Compute/galleries/images/versions/write**
- **Microsoft.Compute/galleries/images/write**
- **Microsoft.Compute/galleries/read**
- **Microsoft.Compute/galleries/write**
- **Microsoft.Compute/snapshots/read**
- **Microsoft.Compute/snapshots/write**
- **Microsoft.Compute/snapshots/delete**
- **Microsoft.Compute/virtualMachines/delete**
- **Microsoft.Compute/virtualMachines/powerOff/action**
- **Microsoft.Compute/virtualMachines/read**
- **Microsoft.Compute/virtualMachines/write**
- **Microsoft.Compute/virtualMachines/deallocate/action**

Example 4.3. Required permissions for creating identity management resources

- **Microsoft.ManagedIdentity/userAssignedIdentities/assign/action**
- **Microsoft.ManagedIdentity/userAssignedIdentities/read**
- **Microsoft.ManagedIdentity/userAssignedIdentities/write**

Example 4.4. Required permissions for creating network resources

- **Microsoft.Network/dnsZones/A/write**
- **Microsoft.Network/dnsZones/CNAME/write**
- **Microsoft.Network/dnszones/CNAME/read**
- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/loadBalancers/backendAddressPools/join/action**
- **Microsoft.Network/loadBalancers/backendAddressPools/read**
- **Microsoft.Network/loadBalancers/backendAddressPools/write**
- **Microsoft.Network/loadBalancers/read**
- **Microsoft.Network/loadBalancers/write**
- **Microsoft.Network/networkInterfaces/delete**
- **Microsoft.Network/networkInterfaces/join/action**
- **Microsoft.Network/networkInterfaces/read**
- **Microsoft.Network/networkInterfaces/write**
- **Microsoft.Network/networkSecurityGroups/join/action**
- **Microsoft.Network/networkSecurityGroups/read**
- **Microsoft.Network/networkSecurityGroups/securityRules/delete**
- **Microsoft.Network/networkSecurityGroups/securityRules/read**
- **Microsoft.Network/networkSecurityGroups/securityRules/write**
- **Microsoft.Network/networkSecurityGroups/write**
- **Microsoft.Network/privateDnsZones/A/read**
- **Microsoft.Network/privateDnsZones/A/write**
- **Microsoft.Network/privateDnsZones/A/delete**
- **Microsoft.Network/privateDnsZones/SOA/read**
- **Microsoft.Network/privateDnsZones/read**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/read**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/write**
- **Microsoft.Network/privateDnsZones/write**
- **Microsoft.Network/publicIPAddresses/delete**
- **Microsoft.Network/publicIPAddresses/join/action**

- **Microsoft.Network/publicIPAddresses/read**
- **Microsoft.Network/publicIPAddresses/write**
- **Microsoft.Network/virtualNetworks/join/action**
- **Microsoft.Network/virtualNetworks/read**
- **Microsoft.Network/virtualNetworks/subnets/join/action**
- **Microsoft.Network/virtualNetworks/subnets/read**
- **Microsoft.Network/virtualNetworks/subnets/write**
- **Microsoft.Network/virtualNetworks/write**

Example 4.5. Required permissions for checking the health of resources

- **Microsoft.Resourcehealth/healthevent/Activated/action**
- **Microsoft.Resourcehealth/healthevent/InProgress/action**
- **Microsoft.Resourcehealth/healthevent/Pending/action**
- **Microsoft.Resourcehealth/healthevent/Resolved/action**
- **Microsoft.Resourcehealth/healthevent/Updated/action**

Example 4.6. Required permissions for creating a resource group

- **Microsoft.Resources/subscriptions/resourceGroups/read**
- **Microsoft.Resources/subscriptions/resourcegroups/write**

Example 4.7. Required permissions for creating resource tags

- **Microsoft.Resources/tags/write**

Example 4.8. Required permissions for creating storage resources

- **Microsoft.Storage/storageAccounts/blobServices/read**
- **Microsoft.Storage/storageAccounts/blobServices/containers/write**
- **Microsoft.Storage/storageAccounts/fileServices/read**
- **Microsoft.Storage/storageAccounts/fileServices/shares/read**
- **Microsoft.Storage/storageAccounts/fileServices/shares/write**
- **Microsoft.Storage/storageAccounts/fileServices/shares/delete**

- **Microsoft.Storage/storageAccounts/listKeys/action**
- **Microsoft.Storage/storageAccounts/read**
- **Microsoft.Storage/storageAccounts/write**

Example 4.9. Required permissions for creating deployments

- **Microsoft.Resources/deployments/read**
- **Microsoft.Resources/deployments/write**
- **Microsoft.Resources/deployments/validate/action**
- **Microsoft.Resources/deployments/operationstatuses/read**

Example 4.10. Optional permissions for creating compute resources

- **Microsoft.Compute/availabilitySets/delete**
- **Microsoft.Compute/availabilitySets/write**

Example 4.11. Optional permissions for creating marketplace virtual machine resources

- **Microsoft.MarketplaceOrdering/offertypes/publishers/offers/plans/agreements/read**
- **Microsoft.MarketplaceOrdering/offertypes/publishers/offers/plans/agreements/write**

Example 4.12. Optional permissions for enabling user-managed encryption

- **Microsoft.Compute/diskEncryptionSets/read**
- **Microsoft.Compute/diskEncryptionSets/write**
- **Microsoft.Compute/diskEncryptionSets/delete**
- **Microsoft.KeyVault/vaults/read**
- **Microsoft.KeyVault/vaults/write**
- **Microsoft.KeyVault/vaults/delete**
- **Microsoft.KeyVault/vaults/deploy/action**
- **Microsoft.KeyVault/vaults/keys/read**
- **Microsoft.KeyVault/vaults/keys/write**
- **Microsoft.Features/providers/features/register/action**

The following permissions are required for deleting an OpenShift Container Platform cluster on Microsoft Azure.

Example 4.13. Required permissions for deleting authorization resources

- **Microsoft.Authorization/roleAssignments/delete**

Example 4.14. Required permissions for deleting compute resources

- **Microsoft.Compute/disks/delete**
- **Microsoft.Compute/galleries/delete**
- **Microsoft.Compute/galleries/images/delete**
- **Microsoft.Compute/galleries/images/versions/delete**
- **Microsoft.Compute/virtualMachines/delete**
- **Microsoft.Compute/images/delete**

Example 4.15. Required permissions for deleting identity management resources

- **Microsoft.ManagedIdentity/userAssignedIdentities/delete**

Example 4.16. Required permissions for deleting network resources

- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/dnsZones/A/read**
- **Microsoft.Network/dnsZones/A/delete**
- **Microsoft.Network/dnsZones/CNAME/read**
- **Microsoft.Network/dnsZones/CNAME/delete**
- **Microsoft.Network/loadBalancers/delete**
- **Microsoft.Network/networkInterfaces/delete**
- **Microsoft.Network/networkSecurityGroups/delete**
- **Microsoft.Network/privateDnsZones/read**
- **Microsoft.Network/privateDnsZones/A/read**
- **Microsoft.Network/privateDnsZones/delete**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/delete**
- **Microsoft.Network/publicIPAddresses/delete**

- **Microsoft.Network/virtualNetworks/delete**

Example 4.17. Required permissions for checking the health of resources

- **Microsoft.Resourcehealth/healthevent/Activated/action**
- **Microsoft.Resourcehealth/healthevent/Resolved/action**
- **Microsoft.Resourcehealth/healthevent/Updated/action**

Example 4.18. Required permissions for deleting a resource group

- **Microsoft.Resources/subscriptions/resourcegroups/delete**

Example 4.19. Required permissions for deleting storage resources

- **Microsoft.Storage/storageAccounts/delete**
- **Microsoft.Storage/storageAccounts/listKeys/action**

NOTE

To install OpenShift Container Platform on Azure, you must scope the permissions related to resource group creation to your subscription. After the resource group is created, you can scope the rest of the permissions to the created resource group. If the public DNS zone is present in a different resource group, then the network DNS zone related permissions must always be applied to your subscription.

You can scope all the permissions to your subscription when deleting an OpenShift Container Platform cluster.

4.2.2.6. Creating a service principal

Because OpenShift Container Platform and its installation program create Microsoft Azure resources by using the Azure Resource Manager, you must create a service principal to represent it.

Prerequisites

- Install or update the Azure CLI.
- Your Azure account has the required roles for the subscription that you use.
- If you want to use a custom role, you have created a custom role with the required permissions listed in the *Required Azure permissions for user-provisioned infrastructure* section.

Procedure

1. Log in to the Azure CLI:

```
$ az login
```

2. If your Azure account uses subscriptions, ensure that you are using the right subscription:

   a. View the list of available accounts and record the **tenantId** value for the subscription you want to use for your cluster:

   ```
   $ az account list --refresh
   ```

   Example output

   ```
   [
     {
       "cloudName": "AzureCloud",
       "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
       "isDefault": true,
       "name": "Subscription Name",
       "state": "Enabled",
       "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee",
       "user": {
         "name": "you@example.com",
         "type": "user"
       }
     }
   ]
   ```

   b. View your active account details and confirm that the **tenantId** value matches the subscription you want to use:

   ```
   $ az account show
   ```

   Example output

   ```
   {
     "environmentName": "AzureCloud",
     "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
     "isDefault": true,
     "name": "Subscription Name",
     "state": "Enabled",
     "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee", 1
     "user": {
       "name": "you@example.com",
       "type": "user"
     }
   }
   ```

   1 Ensure that the value of the **tenantId** parameter is the correct subscription ID.

   c. If you are not using the right subscription, change the active subscription:

   ```
   $ az account set -s <subscription_id> 1
   ```

   1 Specify the subscription ID.

d. Verify the subscription ID update:

```
$ az account show
```

Example output

```
{
  "environmentName": "AzureCloud",
  "id": "33212d16-bdf6-45cb-b038-f6565b61edda",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee",
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

3. Record the **tenantId** and **id** parameter values from the output. You need these values during the OpenShift Container Platform installation.

4. Create the service principal for your account:

```
$ az ad sp create-for-rbac --role <role_name> \ 1
     --name <service_principal> \ 2
     --scopes /subscriptions/<subscription_id> 3
```

1 Defines the role name. You can use the **Contributor** role, or you can specify a custom role which contains the necessary permissions.

2 Defines the service principal name.

3 Specifies the subscription ID.

Example output

```
Creating 'Contributor' role assignment under scope '/subscriptions/<subscription_id>'
The output includes credentials that you must protect. Be sure that you do not
include these credentials in your code or check the credentials into your source
control. For more information, see https://aka.ms/azadsp-cli
{
  "appId": "ac461d78-bf4b-4387-ad16-7e32e328aec6",
  "displayName": <service_principal>",
  "password": "00000000-0000-0000-0000-000000000000",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee"
}
```

5. Record the values of the **appId** and **password** parameters from the previous output. You need these values during OpenShift Container Platform installation.

6. If you applied the **Contributor** role to your service principal, assign the **User Administrator Access** role by running the following command:

```
$ az role assignment create --role "User Access Administrator" \
  --assignee-object-id $(az ad sp show --id <appId> --query id -o tsv) 1
```

1 Specify the **appId** parameter value for your service principal.

Additional resources

- For more information about CCO modes, see About the Cloud Credential Operator.

4.2.2.7. Supported Azure regions

The installation program dynamically generates the list of available Microsoft Azure regions based on your subscription.

Supported Azure public regions

- **australiacentral** (Australia Central)
- **australiaeast** (Australia East)
- **australiasoutheast** (Australia South East)
- **brazilsouth** (Brazil South)
- **canadacentral** (Canada Central)
- **canadaeast** (Canada East)
- **centralindia** (Central India)
- **centralus** (Central US)
- **chilecentral** (Chile Central)
- **eastasia** (East Asia)
- **eastus** (East US)
- **eastus2** (East US 2)
- **francecentral** (France Central)
- **germanywestcentral** (Germany West Central)
- **indonesiacentral** (Indonesia Central)
- **israelcentral** (Israel Central)
- **italynorth** (Italy North)
- **japaneast** (Japan East)
- **japanwest** (Japan West)
- **koreacentral** (Korea Central)

- **koreasouth** (Korea South)
- **malaysiawest** (Malaysia West)
- **mexicocentral** (Mexico Central)
- **newzealandnorth** (New Zealand North)
- **northcentralus** (North Central US)
- **northeurope** (North Europe)
- **norwayeast** (Norway East)
- **polandcentral** (Poland Central)
- **qatarcentral** (Qatar Central)
- **southafricanorth** (South Africa North)
- **southcentralus** (South Central US)
- **southeastasia** (Southeast Asia)
- **southindia** (South India)
- **spaincentral** (Spain Central)
- **swedencentral** (Sweden Central)
- **switzerlandnorth** (Switzerland North)
- **uaenorth** (UAE North)
- **uksouth** (UK South)
- **ukwest** (UK West)
- **westcentralus** (West Central US)
- **westeurope** (West Europe)
- **westindia** (West India)
- **westus** (West US)
- **westus2** (West US 2)
- **westus3** (West US 3)

Supported Azure Government regions

Support for the following Microsoft Azure Government (MAG) regions was added in OpenShift Container Platform version 4.6:

- **usgovtexas** (US Gov Texas)
- **usgovvirginia** (US Gov Virginia)

You can reference all available MAG regions in the Azure documentation. Other provided MAG regions are expected to work with OpenShift Container Platform, but have not been tested.

4.2.3. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

4.2.3.1. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 4.1. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.

IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL) 8.6 and later.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 9.2 and inherits all of its hardware certifications and requirements. See Red Hat Enterprise Linux technology capabilities and limits .

4.2.3.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 4.2. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or Hyper-Threading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

NOTE

For OpenShift Container Platform version 4.18, RHCOS is based on RHEL version 9.4, which updates the micro-architecture requirements. The following list contains the minimum instruction set architectures (ISA) that each architecture requires:

- x86-64 architecture requires x86-64-v2 ISA
- ARM64 architecture requires ARMv8.0-A ISA
- IBM Power architecture requires Power 9 ISA
- s390x architecture requires z14 ISA

For more information, see Architectures (RHEL documentation).

IMPORTANT

You are required to use Azure virtual machines that have the **premiumIO** parameter set to **true**.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

4.2.3.3. Tested instance types for Azure

The following Microsoft Azure instance types have been tested with OpenShift Container Platform.

Example 4.20. Machine types based on 64-bit x86 architecture

- **standardBasv2Family**
- **standardBSFamily**
- **standardBsv2Family**
- **standardDADSv5Family**
- **standardDASv4Family**
- **standardDASv5Family**
- **standardDCACCV5Family**
- **standardDCADCCV5Family**
- **standardDCADSv5Family**
- **standardDCASv5Family**
- **standardDCSv3Family**
- **standardDCSv2Family**
- **standardDDCSv3Family**
- **standardDDSv4Family**
- **standardDDSv5Family**
- **standardDLDSv5Family**
- **standardDLSv5Family**
- **standardDSFamily**
- **standardDSv2Family**
- **standardDSv2PromoFamily**
- **standardDSv3Family**
- **standardDSv4Family**
- **standardDSv5Family**
- **standardEADSv5Family**
- **standardEASv4Family**
- **standardEASv5Family**
- **standardEBDSv5Family**

- **standardEBSv5Family**
- **standardECACCV5Family**
- **standardECADCCV5Family**
- **standardECADSv5Family**
- **standardECASv5Family**
- **standardEDSv4Family**
- **standardEDSv5Family**
- **standardEIADSv5Family**
- **standardEIASv4Family**
- **standardEIASv5Family**
- **standardEIBDSv5Family**
- **standardEIBSv5Family**
- **standardEIDSv5Family**
- **standardEISv3Family**
- **standardEISv5Family**
- **standardESv3Family**
- **standardESv4Family**
- **standardESv5Family**
- **standardFXMDVSFamily**
- **standardFSFamily**
- **standardFSv2Family**
- **standardGSFamily**
- **standardHBrsv2Family**
- **standardHBSFamily**
- **standardHBv4Family**
- **standardHCSFamily**
- **standardHXFamily**
- **standardLASv3Family**
- **standardLSFamily**

- **standardLSv2Family**
- **standardLSv3Family**
- **standardMDSHighMemoryv3Family**
- **standardMDSMediumMemoryv2Family**
- **standardMDSMediumMemoryv3Family**
- **standardMIDSHighMemoryv3Family**
- **standardMIDSMediumMemoryv2Family**
- **standardMISHighMemoryv3Family**
- **standardMISMediumMemoryv2Family**
- **standardMSFamily**
- **standardMSHighMemoryv3Family**
- **standardMSMediumMemoryv2Family**
- **standardMSMediumMemoryv3Family**
- **StandardNCADSA100v4Family**
- **Standard NCASv3_T4 Family**
- **standardNCSv3Family**
- **standardNDSv2Family**
- **StandardNGADSV620v1Family**
- **standardNPSFamily**
- **StandardNVADSA10v5Family**
- **standardNVSv3Family**
- **standardXEISv4Family**

4.2.3.4. Tested instance types for Azure on 64-bit ARM infrastructures

The following Microsoft Azure ARM64 instance types have been tested with OpenShift Container Platform.

Example 4.21. Machine types based on 64-bit ARM architecture

- **standardBpsv2Family**
- **standardDPSv5Family**
- **standardDPDSv5Family**

- **standardDPLDSv5Family**
- **standardDPLSv5Family**
- **standardEPSv5Family**
- **standardEPDSv5Family**
- **StandardDpdsv6Family**
- **StandardDpldsv6Famil**
- **StandardDplsv6Family**
- **StandardDpsv6Family**
- **StandardEpdsv6Family**
- **StandardEpsv6Family**

4.2.4. Using the Azure Marketplace offering

Using the Azure Marketplace offering lets you deploy an OpenShift Container Platform cluster, which is billed on pay-per-use basis (hourly, per core) through Azure, while still being supported directly by Red Hat.

To deploy an OpenShift Container Platform cluster using the Azure Marketplace offering, you must first obtain the Azure Marketplace image. The installation program uses this image to deploy worker or control plane nodes. When obtaining your image, consider the following:

- While the images are the same, the Azure Marketplace publisher is different depending on your region. If you are located in North America, specify **redhat** as the publisher. If you are located in EMEA, specify **redhat-limited** as the publisher.
- The offer includes a **rh-ocp-worker** SKU and a **rh-ocp-worker-gen1** SKU. The **rh-ocp-worker** SKU represents a Hyper-V generation version 2 VM image. The default instance types used in OpenShift Container Platform are version 2 compatible. If you plan to use an instance type that is only version 1 compatible, use the image associated with the **rh-ocp-worker-gen1** SKU. The **rh-ocp-worker-gen1** SKU represents a Hyper-V version 1 VM image.

IMPORTANT

Installing images with the Azure marketplace is not supported on clusters with 64-bit ARM instances.

You should only modify the RHCOS image for compute machines to use an Azure Marketplace image. Control plane machines and infrastructure nodes do not require an OpenShift Container Platform subscription and use the public RHCOS default image by default, which does not incur subscription costs on your Azure bill. Therefore, you should not modify the cluster default boot image or the control plane boot images. Applying the Azure Marketplace image to them will incur additional licensing costs that cannot be recovered.

Prerequisites

- You have installed the Azure CLI client **(az)**.
- Your Azure account is entitled for the offer and you have logged into this account with the Azure CLI client.

Procedure

1. Display all of the available OpenShift Container Platform images by running one of the following commands:
   - North America:

     ```
     $  az vm image list --all --offer rh-ocp-worker --publisher redhat -o table
     ```

     #### Example output

     ```
     Offer         Publisher       Sku                 Urn                                                           Version
     ------------  -------------  -----------------  ----------------------------------------------------------
     ----  -----------------
     rh-ocp-worker  RedHat         rh-ocp-worker       RedHat:rh-ocp-worker:rh-ocp-
     worker:4.15.2024072409             4.15.2024072409
     rh-ocp-worker  RedHat         rh-ocp-worker-gen1  RedHat:rh-ocp-worker:rh-ocp-worker-
     gen1:4.15.2024072409         4.15.2024072409
     ```

   - EMEA:

     ```
     $  az vm image list --all --offer rh-ocp-worker --publisher redhat-limited -o table
     ```

     #### Example output

     ```
     Offer         Publisher       Sku                 Urn
     Version
     ------------  -------------  -----------------  ----------------------------------------------------------
     ----          -----------------
     rh-ocp-worker  redhat-limited  rh-ocp-worker       redhat-limited:rh-ocp-worker:rh-ocp-
     worker:4.15.2024072409             4.15.2024072409
     rh-ocp-worker  redhat-limited  rh-ocp-worker-gen1  redhat-limited:rh-ocp-worker:rh-ocp-
     worker-gen1:4.15.2024072409         4.15.2024072409
     ```

   #### NOTE

   Use the latest image that is available for compute and control plane nodes. If required, your VMs are automatically upgraded as part of the installation process.

2. Inspect the image for your offer by running one of the following commands:
   - North America:

     ```
     $ az vm image show --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
     ```

   - EMEA:

     ```
     $ az vm image show --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
     ```

3. Review the terms of the offer by running one of the following commands:

   - North America:

     ```
     $ az vm image terms show --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
     ```

   - EMEA:

     ```
     $ az vm image terms show --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
     ```

4. Accept the terms of the offering by running one of the following commands:

   - North America:

     ```
     $ az vm image terms accept --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
     ```

   - EMEA:

     ```
     $ az vm image terms accept --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
     ```

5. Record the image details of your offer. If you use the Azure Resource Manager (ARM) template to deploy your compute nodes:

   a. Update **storageProfile.imageReference** by deleting the **id** parameter and adding the **offer**, **publisher**, **sku**, and **version** parameters by using the values from your offer.

   b. Specify a **plan** for the virtual machines (VMs).

   Example 06_workers.json ARM template with an updated storageProfile.imageReference object and a specified plan

   ```
   ...
    "plan" : {
     "name": "rh-ocp-worker",
     "product": "rh-ocp-worker",
     "publisher": "redhat"
    },
    "dependsOn" : [
     "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')
   [copyIndex()], '-nic'))]"
    ],
    "properties" : {
   ...
    "storageProfile": {
     "imageReference": {
     "offer": "rh-ocp-worker",
     "publisher": "redhat",
     "sku": "rh-ocp-worker",
     "version": "413.92.2023101700"
     }
     ...
    }
   ...
    }
   ```

4.2.4.1. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Go to the Cluster Type page on the Red Hat Hybrid Cloud Console. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

   TIP

   You can also download the binaries for a specific OpenShift Container Platform release .

2. Select your infrastructure provider from the **Run it yourself** section of the page.

3. Select your host operating system and architecture from the dropdown menus under **OpenShift Installer** and click **Download Installer**.

4. Place the downloaded file in the directory where you want to store the installation configuration files.

   IMPORTANT

   - The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both of the files are required to delete the cluster.

   - Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

5. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

   ```
   $ tar -xvf openshift-install-linux.tar.gz
   ```

6. Download your installation pull secret from Red Hat OpenShift Cluster Manager . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

TIP

Alternatively, you can retrieve the installation program from the Red Hat Customer Portal, where you can specify a version of the installation program to download. However, you must have an active subscription to access this page.

4.2.4.2. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the **~/.ssh/authorized_keys** list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The **./openshift-install gather** command also requires the SSH public key to be in place on the cluster nodes.

IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

NOTE

You must use a local key, not one that you configured with platform-specific approaches.

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 Specify the path and file name, such as **~/.ssh/id_ed25519**, of the new SSH key. If you have an existing key pair, ensure your public key is in the your **~/.ssh** directory.

NOTE

If you plan to install an OpenShift Container Platform cluster that uses the RHEL cryptographic libraries that have been submitted to NIST for FIPS 140-2/140-3 Validation on only the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the **~/.ssh/id_ed25519.pub** public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the **./openshift-install gather** command.

   NOTE

   On some distributions, default SSH private key identities such as **~/.ssh/id_rsa** and **~/.ssh/id_dsa** are managed automatically.

   a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

   ```
   $ eval "$(ssh-agent -s)"
   ```

   Example output

   ```
   Agent pid 31874
   ```

   NOTE

   If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

   ```
   $ ssh-add <path>/<file_name> 1
   ```

   1 Specify the path and file name for your SSH private key, such as **~/.ssh/id_ed25519**

   Example output

   ```
   Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
   ```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide the key to the installation program.

4.2.5. Creating the installation files for Azure

To install OpenShift Container Platform on Microsoft Azure using user-provisioned infrastructure, you must generate the files that the installation program needs to deploy your cluster and modify them so that the cluster creates only the machines that it will use. You generate and customize the **install-config.yaml** file, Kubernetes manifests, and Ignition config files. You also have the option to first set up a separate **var** partition during the preparation phases of installation.

4.2.5.1. Optional: Creating a separate /var partition

It is recommended that disk partitioning for OpenShift Container Platform be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the **/var** partition or a subdirectory of **/var**. For example:

- **/var/lib/containers**: Holds container-related content that can grow as more images and containers are added to a system.
- **/var/lib/etcd**: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- **/var**: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a **/var** directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because **/var** must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate **/var** partition by creating a machine config manifest that is inserted during the **openshift-install** preparation phases of an OpenShift Container Platform installation.

IMPORTANT

If you follow the steps to create a separate **/var** partition in this procedure, it is not necessary to create the Kubernetes manifest and Ignition config files again as described later in this section.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

```
$ mkdir $HOME/clusterconfig
```

2. Run **openshift-install** to create a set of files in the **manifest** and **openshift** subdirectories. Answer the system questions as you are prompted:

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

Example output

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. Optional: Confirm that the installation program created manifests in the **clusterconfig/openshift** directory:

```
$ ls $HOME/clusterconfig/openshift/
```

Example output

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. Create a Butane config that configures the additional partition. For example, name the file **$HOME/clusterconfig/98-var-partition.bu**, change the disk device name to the name of the storage device on the **worker** systems, and set the storage size as appropriate. This example places the **/var** directory on a separate partition:

```
variant: openshift
version: 4.18.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

1 The storage device name of the disk that you want to partition.

2 When adding a data partition to the boot disk, a minimum value of 25000 MiB (Mebibytes) is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.

3 The size of the data partition in mebibytes.

4 The **prjquota** mount option must be enabled for filesystems used for container storage.

NOTE

When creating a separate **/var** partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

5. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

   ```
   $ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
   ```

6. Run **openshift-install** again to create Ignition configs from a set of files in the **manifest** and **openshift** subdirectories:

   ```
   $ openshift-install create ignition-configs --dir $HOME/clusterconfig
   $ ls $HOME/clusterconfig/
   auth  bootstrap.ign  master.ign  metadata.json  worker.ign
   ```

Now you can use the Ignition config files as input to the installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

4.2.5.2. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Microsoft Azure.

Prerequisites

- You have the OpenShift Container Platform installation program and the pull secret for your cluster. For a restricted network installation, these files are on your mirror host.
- You have the **imageContentSources** values that were generated during mirror registry creation.
- You have obtained the contents of the certificate for your mirror registry.
- You have retrieved a Red Hat Enterprise Linux CoreOS (RHCOS) image and uploaded it to an accessible location.
- You have an Azure subscription ID and tenant ID.
- If you are installing the cluster using a service principal, you have its application ID and password.
- If you are installing the cluster using a system-assigned managed identity, you have enabled it on the virtual machine that you will run the installation program from.
- If you are installing the cluster using a user-assigned managed identity, you have met these prerequisites:
  - You have its client ID.
  - You have assigned it to the virtual machine that you will run the installation program from.

Procedure

1. Optional: If you have run the installation program on this computer before, and want to use an alternative service principal or managed identity, go to the **~/.azure/** directory and delete the **osServicePrincipal.json** configuration file.
   Deleting this file prevents the installation program from automatically reusing subscription and authentication values from a previous installation.

2. Create the **install-config.yaml** file.

a. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1 For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

When specifying the directory:

- Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
- Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:

i. Optional: Select an SSH key to use to access your cluster machines.

NOTE

For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

ii. Select **azure** as the platform to target.
If the installation program cannot locate the **osServicePrincipal.json** configuration file from a previous installation, you are prompted for Azure subscription and authentication values.

iii. Enter the following Azure parameter values for your subscription:

- **azure subscription id** Enter the subscription ID to use for the cluster.
- **azure tenant id** Enter the tenant ID.

iv. Depending on the Azure identity you are using to deploy the cluster, do one of the following when prompted for the **azure service principal client id**

- If you are using a service principal, enter its application ID.
- If you are using a system-assigned managed identity, leave this value blank.
- If you are using a user-assigned managed identity, specify its client ID.

v. Depending on the Azure identity you are using to deploy the cluster, do one of the following when prompted for the **azure service principal client secret**

- If you are using a service principal, enter its password.
- If you are using a system-assigned managed identity, leave this value blank.

- If you are using a user-assigned managed identity, leave this value blank.

vi. Select the region to deploy the cluster to.

vii. Select the base domain to deploy the cluster to. The base domain corresponds to the Azure DNS Zone that you created for your cluster.

viii. Enter a descriptive name for your cluster.

> **IMPORTANT**
>
> All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see Resolve reserved resource name errors in the Azure documentation.

ix. Paste the pull secret from Red Hat OpenShift Cluster Manager .

3. Edit the **install-config.yaml** file to give the additional information that is required for an installation in a restricted network.

   a. Update the **pullSecret** value to contain the authentication information for your registry:

   ```
   pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email": "you@example.com"}}}'
   ```

   For **<mirror_host_name>**, specify the registry domain name that you specified in the certificate for your mirror registry, and for **<credentials>**, specify the base64-encoded user name and password for your mirror registry.

   b. Add the **additionalTrustBundle** parameter and value.

   ```
   additionalTrustBundle: |
     -----BEGIN CERTIFICATE-----
   ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
     -----END CERTIFICATE-----
   ```

   The value must be the contents of the certificate file that you used for your mirror registry. The certificate file can be an existing, trusted certificate authority, or the self-signed certificate that you generated for the mirror registry.

   c. Define the network and subnets for the VNet to install the cluster under the **platform.azure** field:

   ```
   networkResourceGroupName: <vnet_resource_group> 1
   virtualNetwork: <vnet> 2
   controlPlaneSubnet: <control_plane_subnet> 3
   computeSubnet: <compute_subnet> 4
   ```

   1 Replace **<vnet_resource_group>** with the resource group name that contains the existing virtual network (VNet).

   2 Replace **<vnet>** with the existing virtual network name.

3 Replace **<control_plane_subnet>** with the existing subnet name to deploy the control plane machines.

4 Replace **<compute_subnet>** with the existing subnet name to deploy compute machines.

d. Add the image content resources, which resemble the following YAML excerpt:

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.redhat.io/ocp/release
```

For these values, use the **imageContentSources** that you recorded during mirror registry creation.

e. Optional: Set the publishing strategy to **Internal**:

```
publish: Internal
```

By setting this option, you create an internal Ingress Controller and a private load balancer.

> **IMPORTANT**
>
> Azure Firewall does not work seamlessly with Azure Public Load balancers. Thus, when using Azure Firewall for restricting internet access, the **publish** field in **install-config.yaml** should be set to **Internal**.

4. Make any other modifications to the **install-config.yaml** file that you require. For more information about the parameters, see "Installation configuration parameters".

5. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

> **IMPORTANT**
>
> The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

If previously not detected, the installation program creates an **osServicePrincipal.json** configuration file and stores this file in the **~/.azure/** directory on your computer. This ensures that the installation program can load the profile when it is creating an OpenShift Container Platform cluster on the target platform.

4.2.5.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.

> NOTE
>
> The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.
>
> For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1 A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.

2 A proxy URL to use for creating HTTPS connections outside the cluster.

3 A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use * to bypass the proxy for all destinations.

4 If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

5 Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and

NOTE

The installation program does not support the proxy **readinessEndpoints** field.

NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

4.2.5.4. Exporting common variables for ARM templates

You must export a common set of variables that are used with the provided Azure Resource Manager (ARM) templates used to assist in completing a user-provided infrastructure install on Microsoft Azure.

NOTE

Specific ARM templates can also require additional exported variables, which are detailed in their related procedures.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Export common variables found in the **install-config.yaml** to be used by the provided ARM templates:

```
$ export CLUSTER_NAME=<cluster_name> 1
$ export AZURE_REGION=<azure_region> 2
$ export SSH_KEY=<ssh_key> 3
$ export BASE_DOMAIN=<base_domain> 4
$ export BASE_DOMAIN_RESOURCE_GROUP=<base_domain_resource_group> 5
```

1 The value of the **.metadata.name** attribute from the **install-config.yaml** file.

2 The region to deploy the cluster into, for example **centralus**. This is the value of the **.platform.azure.region** attribute from the **install-config.yaml** file.

3 The SSH RSA public key file as a string. You must enclose the SSH key in quotes since it contains spaces. This is the value of the **.sshKey** attribute from the **install-config.yaml**

4 The base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster. This is the value of the **.baseDomain** attribute from the **install-config.yaml** file.

5 The resource group where the public DNS zone exists. This is the value of the **.platform.azure.baseDomainResourceGroupName** attribute from the **install-config.yaml** file.

For example:

```
$ export CLUSTER_NAME=test-cluster
$ export AZURE_REGION=centralus
$ export SSH_KEY="ssh-rsa xxx/xxx/xxx= user@email.com"
$ export BASE_DOMAIN=example.com
$ export BASE_DOMAIN_RESOURCE_GROUP=ocp-cluster
```

2. Export the kubeadmin credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

4.2.5.5. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.

IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

   ```
   $ ./openshift-install create manifests --dir <installation_directory> 1
   ```

   1 For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

2. Remove the Kubernetes manifest files that define the control plane machines:

   ```
   $ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
   ```

   By removing these files, you prevent the cluster from automatically generating control plane machines.

3. Remove the Kubernetes manifest files that define the control plane machine set:

   ```
   $ rm -f <installation_directory>/openshift/99_openshift-machine-api_master-control-plane-machine-set.yaml
   ```

4. Remove the Kubernetes manifest files that define the worker machines:

   ```
   $ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
   ```

   > **IMPORTANT**
   >
   > If you disabled the **MachineAPI** capability when installing a cluster on user-provisioned infrastructure, you must remove the Kubernetes manifest files that define the worker machines. Otherwise, your cluster fails to install.

   Because you create and manage the worker machines yourself, you do not need to initialize these machines.

5. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:

   a. Open the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file.

   b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.

   c. Save and exit the file.

6. Optional: If you do not want the Ingress Operator to create DNS records on your behalf, remove the **privateZone** and **publicZone** sections from the **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS configuration file:

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
    id: mycluster-100419-private-zone
  publicZone: 2
    id: example.openshift.com
status: {}
```

1 2 Remove this section completely.

If you do so, you must add ingress DNS records manually in a later step.

7. When configuring Azure on user-provisioned infrastructure, you must export some common variables defined in the manifest files to use later in the Azure Resource Manager (ARM) templates:

   a. Export the infrastructure ID by using the following command:

      ```
      $ export INFRA_ID=<infra_id> 1
      ```

      1 The OpenShift Container Platform cluster has been assigned an identifier (**INFRA_ID**) in the form of **<cluster_name>-<random_string>**. This will be used as the base name for most resources created using the provided ARM templates. This is the value of the **.status.infrastructureName** attribute from the **manifests/cluster-infrastructure-02-config.yml** file.

   b. Export the resource group by using the following command:

      ```
      $ export RESOURCE_GROUP=<resource_group> 1
      ```

      1 All resources created in this Azure deployment exists as part of a resource group. The resource group name is also based on the **INFRA_ID**, in the form of **<cluster_name>-<random_string>-rg**. This is the value of the **.status.platformStatus.azure.resourceGroupName** attribute from the **manifests/cluster-infrastructure-02-config.yml** file.

8. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

   ```
   $ ./openshift-install create ignition-configs --dir <installation_directory> 1
   ```

   1 For **<installation_directory>**, specify the same installation directory.

   Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the **./<installation_directory>/auth** directory:

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

4.2.6. Creating the Azure resource group

You must create a Microsoft Azure resource group and an identity for that resource group. These are both used during the installation of your OpenShift Container Platform cluster on Azure.

Procedure

1. Create the resource group in a supported Azure region:

   ```
   $ az group create --name ${RESOURCE_GROUP} --location ${AZURE_REGION}
   ```

2. Create an Azure identity for the resource group:

   ```
   $ az identity create -g ${RESOURCE_GROUP} -n ${INFRA_ID}-identity
   ```

   This is used to grant the required access to Operators in your cluster. For example, this allows the Ingress Operator to create a public IP and its load balancer. You must assign the Azure identity to a role.

3. Grant the Contributor role to the Azure identity:

   a. Export the following variables required by the Azure role assignment:

      ```
      $ export PRINCIPAL_ID=`az identity show -g ${RESOURCE_GROUP} -n ${INFRA_ID}-identity --query principalId --out tsv`
      ```

      ```
      $ export RESOURCE_GROUP_ID=`az group show -g ${RESOURCE_GROUP} --query id --out tsv`
      ```

   b. Assign the Contributor role to the identity:

      ```
      $ az role assignment create --assignee "${PRINCIPAL_ID}" --role 'Contributor' --scope "${RESOURCE_GROUP_ID}"
      ```

NOTE

If you want to assign a custom role with all the required permissions to the identity, run the following command:

```
$ az role assignment create --assignee "${PRINCIPAL_ID}" --role
<custom_role> \ 1
--scope "${RESOURCE_GROUP_ID}"
```

1 Specifies the custom role name.

4.2.7. Uploading the RHCOS cluster image and bootstrap Ignition config file

The Azure client does not support deployments based on files existing locally. You must copy and store the RHCOS virtual hard disk (VHD) cluster image and bootstrap Ignition config file in a storage container so they are accessible during deployment.

Prerequisites

- Generate the Ignition config files for your cluster.

Procedure

1. Create an Azure storage account to store the VHD cluster image:

```
$ az storage account create -g ${RESOURCE_GROUP} --location ${AZURE_REGION} --
name ${CLUSTER_NAME}sa --kind Storage --sku Standard_LRS
```

WARNING

The Azure storage account name must be between 3 and 24 characters in length and use numbers and lower-case letters only. If your **CLUSTER_NAME** variable does not follow these restrictions, you must manually define the Azure storage account name. For more information on Azure storage account name restrictions, see Resolve errors for storage account names in the Azure documentation.

2. Export the storage account key as an environment variable:

```
$ export ACCOUNT_KEY=`az storage account keys list -g ${RESOURCE_GROUP} --
account-name ${CLUSTER_NAME}sa --query "[0].value" -o tsv`
```

3. Export the URL of the RHCOS VHD to an environment variable:

```
$ export VHD_URL=`openshift-install coreos print-stream-json | jq -r '.architectures.
<architecture>."rhel-coreos-extensions"."azure-disk".url'`
```

where:

<architecture>

Specifies the architecture, valid values include **x86_64** or **aarch64**.

IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must specify an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

4. Create the storage container for the VHD:

```
$ az storage container create --name vhd --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY}
```

5. Copy the local VHD to a blob:

```
$ az storage blob copy start --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} --destination-blob "rhcos.vhd" --destination-container vhd --source-uri "${VHD_URL}"
```

6. Create a blob storage container and upload the generated **bootstrap.ign** file:

```
$ az storage container create --name files --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY}
```

```
$ az storage blob upload --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} -c "files" -f "<installation_directory>/bootstrap.ign" -n "bootstrap.ign"
```

4.2.8. Example for creating DNS zones

DNS records are required for clusters that use user-provisioned infrastructure. You should choose the DNS strategy that fits your scenario.

For this example, Azure's DNS solution is used, so you will create a new public DNS zone for external (internet) visibility and a private DNS zone for internal cluster resolution.

NOTE

The public DNS zone is not required to exist in the same resource group as the cluster deployment and might already exist in your organization for the desired base domain. If that is the case, you can skip creating the public DNS zone; be sure the installation config you generated earlier reflects that scenario.

Procedure

1. Create the new public DNS zone in the resource group exported in the **BASE_DOMAIN_RESOURCE_GROUP** environment variable:

```
$ az network dns zone create -g ${BASE_DOMAIN_RESOURCE_GROUP} -n
${CLUSTER_NAME}.${BASE_DOMAIN}
```

You can skip this step if you are using a public DNS zone that already exists.

2. Create the private DNS zone in the same resource group as the rest of this deployment:

```
$ az network private-dns zone create -g ${RESOURCE_GROUP} -n
${CLUSTER_NAME}.${BASE_DOMAIN}
```

You can learn more about configuring a public DNS zone in Azure by visiting that section.

4.2.9. Creating a VNet in Azure

You must create a virtual network (VNet) in Microsoft Azure for your OpenShift Container Platform cluster to use. You can customize the VNet to meet your requirements. One way to create the VNet is to modify the provided Azure Resource Manager (ARM) template.

NOTE

If you do not use the provided ARM template to create your Azure infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Procedure

1. Copy the template from the **ARM template for the VNet**section of this topic and save it as **01_vnet.json** in your cluster's installation directory. This template describes the VNet that your cluster requires.

2. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/01_vnet.json" \
  --parameters baseName="${INFRA_ID}" 1
```

1 The base name to be used in resource names; this is usually the cluster's infrastructure ID.

3. Link the VNet template to the private DNS zone:

```
$ az network private-dns link vnet create -g ${RESOURCE_GROUP} -z
${CLUSTER_NAME}.${BASE_DOMAIN} -n ${INFRA_ID}-network-link -v "${INFRA_ID}-vnet"
-e false
```

4.2.9.1. ARM template for the VNet

You can use the following Azure Resource Manager (ARM) template to deploy the VNet that you need for your OpenShift Container Platform cluster:

Example 4.22. 01_vnet.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(parameters('baseName'), '-vnet')]",
    "addressPrefix" : "10.0.0.0/16",
    "masterSubnetName" : "[concat(parameters('baseName'), '-master-subnet')]",
    "masterSubnetPrefix" : "10.0.0.0/24",
    "nodeSubnetName" : "[concat(parameters('baseName'), '-worker-subnet')]",
    "nodeSubnetPrefix" : "10.0.1.0/24",
    "clusterNsgName" : "[concat(parameters('baseName'), '-nsg')]"
  },
  "resources" : [
    {
      "apiVersion" : "2018-12-01",
      "type" : "Microsoft.Network/virtualNetworks",
      "name" : "[variables('virtualNetworkName')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[concat('Microsoft.Network/networkSecurityGroups/', variables('clusterNsgName'))]"
      ],
      "properties" : {
        "addressSpace" : {
          "addressPrefixes" : [
            "[variables('addressPrefix')]"
          ]
        },
        "subnets" : [
          {
            "name" : "[variables('masterSubnetName')]",
            "properties" : {
              "addressPrefix" : "[variables('masterSubnetPrefix')]",
              "serviceEndpoints": [],
              "networkSecurityGroup" : {
                "id" : "[resourceId('Microsoft.Network/networkSecurityGroups',
variables('clusterNsgName'))]"
              }
            }
          },
          {
            "name" : "[variables('nodeSubnetName')]",
            "properties" : {
              "addressPrefix" : "[variables('nodeSubnetPrefix')]",
              "serviceEndpoints": [],
              "networkSecurityGroup" : {
```

```
        "id" : "[resourceId('Microsoft.Network/networkSecurityGroups',
variables('clusterNsgName'))]"
       }
      }
     }
    ]
   }
  },
  {
   "type" : "Microsoft.Network/networkSecurityGroups",
   "name" : "[variables('clusterNsgName')]",
   "apiVersion" : "2018-10-01",
   "location" : "[variables('location')]",
   "properties" : {
    "securityRules" : [
     {
      "name" : "apiserver_in",
      "properties" : {
       "protocol" : "Tcp",
       "sourcePortRange" : "*",
       "destinationPortRange" : "6443",
       "sourceAddressPrefix" : "*",
       "destinationAddressPrefix" : "*",
       "access" : "Allow",
       "priority" : 101,
       "direction" : "Inbound"
      }
     }
    ]
   }
  }
 ]
}
```

4.2.10. Deploying the RHCOS cluster image for the Azure infrastructure

You must use a valid Red Hat Enterprise Linux CoreOS (RHCOS) image for Microsoft Azure for your OpenShift Container Platform nodes.

Prerequisites

- Store the RHCOS virtual hard disk (VHD) cluster image in an Azure storage container.

- Store the bootstrap Ignition config file in an Azure storage container.

Procedure

1. Copy the template from the **ARM template for image storage**section of this topic and save it as **02_storage.json** in your cluster's installation directory. This template describes the image storage that your cluster requires.

2. Export the RHCOS VHD blob URL as a variable:

```
$ export VHD_BLOB_URL=`az storage blob url --account-name ${CLUSTER_NAME}sa --
account-key ${ACCOUNT_KEY} -c vhd -n "rhcos.vhd" -o tsv`
```

3. Deploy the cluster image:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/02_storage.json" \
  --parameters vhdBlobURL="${VHD_BLOB_URL}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameters storageAccount="${CLUSTER_NAME}sa" \ 3
  --parameters architecture="<architecture>" 4
```

1 The blob URL of the RHCOS VHD to be used to create master and worker machines.

2 The base name to be used in resource names; this is usually the cluster’s infrastructure ID.

3 The name of your Azure storage account.

4 Specify the system architecture. Valid values are **x64** (default) or **Arm64**.

4.2.10.1. ARM template for image storage

You can use the following Azure Resource Manager (ARM) template to deploy the stored Red Hat Enterprise Linux CoreOS (RHCOS) image that you need for your OpenShift Container Platform cluster:

Example 4.23. 02_storage.json ARM template

```
{
  "$schema": "https://schema.management.azure.com/schemas/2019-04-
01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "architecture": {
      "type": "string",
      "metadata": {
        "description": "The architecture of the Virtual Machines"
      },
      "defaultValue": "x64",
      "allowedValues": [
        "Arm64",
        "x64"
      ]
    },
    "baseName": {
      "type": "string",
      "minLength": 1,
      "metadata": {
        "description": "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "storageAccount": {
      "type": "string",
      "metadata": {
        "description": "The Storage Account name"
      }
    },
    "vhdBlobURL": {
      "type": "string",
      "metadata": {
        "description": "URL pointing to the blob where the VHD to be used to create master and worker machines is located"
      }
    }
  },
  "variables": {
    "location": "[resourceGroup().location]",
    "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
    "imageName": "[parameters('baseName')]",
    "imageNameGen2": "[concat(parameters('baseName'), '-gen2')]",
    "imageRelease": "1.0.0"
  },
  "resources": [
    {
      "apiVersion": "2021-10-01",
      "type": "Microsoft.Compute/galleries",
      "name": "[variables('galleryName')]",
      "location": "[variables('location')]",
      "resources": [
        {
          "apiVersion": "2021-10-01",
          "type": "images",
          "name": "[variables('imageName')]",
          "location": "[variables('location')]",
          "dependsOn": [
            "[variables('galleryName')]"
          ],
          "properties": {
            "architecture": "[parameters('architecture')]",
            "hyperVGeneration": "V1",
            "identifier": {
              "offer": "rhcos",
              "publisher": "RedHat",
              "sku": "basic"
            },
            "osState": "Generalized",
            "osType": "Linux"
          },
          "resources": [
            {
              "apiVersion": "2021-10-01",
              "type": "versions",
              "name": "[variables('imageRelease')]",
              "location": "[variables('location')]",
              "dependsOn": [
                "[variables('imageName')]"
              ],
              "properties": {
                "publishingProfile": {
                  "storageAccountType": "Standard_LRS",
         "targetRegions": [
          {
           "name": "[variables('location')]",
           "regionalReplicaCount": "1"
          }
         ]
        },
        "storageProfile": {
         "osDiskImage": {
          "source": {
           "id": "[resourceId('Microsoft.Storage/storageAccounts',
parameters('storageAccount'))]",
           "uri": "[parameters('vhdBlobURL')]"
          }
         }
        }
       }
      }
     ]
    },
    {
     "apiVersion": "2021-10-01",
     "type": "images",
     "name": "[variables('imageNameGen2')]",
     "location": "[variables('location')]",
     "dependsOn": [
      "[variables('galleryName')]"
     ],
     "properties": {
      "architecture": "[parameters('architecture')]",
      "hyperVGeneration": "V2",
      "identifier": {
       "offer": "rhcos-gen2",
       "publisher": "RedHat-gen2",
       "sku": "gen2"
      },
      "osState": "Generalized",
      "osType": "Linux"
     },
     "resources": [
      {
       "apiVersion": "2021-10-01",
       "type": "versions",
       "name": "[variables('imageRelease')]",
       "location": "[variables('location')]",
       "dependsOn": [
        "[variables('imageNameGen2')]"
       ],
       "properties": {
        "publishingProfile": {
         "storageAccountType": "Standard_LRS",
         "targetRegions": [
          {
           "name": "[variables('location')]",
           "regionalReplicaCount": "1"
          }
```

```
                  ]
                },
                "storageProfile": {
                  "osDiskImage": {
                    "source": {
                      "id": "[resourceId('Microsoft.Storage/storageAccounts',
parameters('storageAccount'))]",
                      "uri": "[parameters('vhdBlobURL')]"
                    }
                  }
                }
              }
            }
          ]
        }
      ]
    }
  ]
}
```

4.2.11. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

4.2.11.1. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.

IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 4.3. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	**1936**	Metrics
	9000-**9999**	Host level services, including the node exporter on ports **9100**-**9101** and the Cluster Version Operator on port**9099**.
	10250-**10259**	The default ports that Kubernetes reserves

Protocol	Port	Description
UDP	**4789**	VXLAN
	6081	Geneve
	9000-**9999**	Host level services, including the node exporter on ports **9100**-**9101**.
	500	IPsec IKE packets
	4500	IPsec NAT-T packets
	123	Network Time Protocol (NTP) on UDP port **123** If an external NTP time server is configured, you must open UDP port **123**.
TCP/UDP	**30000**-**32767**	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 4.4. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	**6443**	Kubernetes API

Table 4.5. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	**2379**-**2380**	etcd server and peer ports

4.2.12. Creating networking and load balancing components in Azure

You must configure networking and load balancing in Microsoft Azure for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Azure Resource Manager (ARM) template.

NOTE

If you do not use the provided ARM template to create your Azure infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Create and configure a VNet and associated subnets in Azure.

Procedure

1. Copy the template from the **ARM template for the network and load balancers**section of this topic and save it as **03_infra.json** in your cluster's installation directory. This template describes the networking and load balancing objects that your cluster requires.

2. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/03_infra.json" \
  --parameters privateDNSZoneName="${CLUSTER_NAME}.${BASE_DOMAIN}" \ ①
  --parameters baseName="${INFRA_ID}"②
```

① The name of the private DNS zone.

② The base name to be used in resource names; this is usually the cluster's infrastructure ID.

3. Create an **api** DNS record in the public zone for the API public load balancer. The **${BASE_DOMAIN_RESOURCE_GROUP}** variable must point to the resource group where the public DNS zone exists.

   a. Export the following variable:

```
$ export PUBLIC_IP=`az network public-ip list -g ${RESOURCE_GROUP} --query "[?
name=='${INFRA_ID}-master-pip'] | [0].ipAddress" -o tsv`
```

   b. Create the **api** DNS record in a new public zone:

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${CLUSTER_NAME}.${BASE_DOMAIN} -n api -a ${PUBLIC_IP} --ttl 60
```

   If you are adding the cluster to an existing public zone, you can create the **api** DNS record in it instead:

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${BASE_DOMAIN} -n api.${CLUSTER_NAME} -a ${PUBLIC_IP} --ttl 60
```

4.2.12.1. ARM template for the network and load balancers

You can use the following Azure Resource Manager (ARM) template to deploy the networking objects and load balancers that you need for your OpenShift Container Platform cluster:

Example 4.24. 03_infra.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
   "minLength" : 1,
   "metadata" : {
    "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
   }
  },
  "vnetBaseName": {
   "type": "string",
   "defaultValue": "",
   "metadata" : {
    "description" : "The specific customer vnet's base name (optional)"
   }
  },
  "privateDNSZoneName" : {
   "type" : "string",
   "metadata" : {
    "description" : "Name of the private DNS zone"
   }
  }
 },
 "variables" : {
  "location" : "[resourceGroup().location]",
  "virtualNetworkName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
  "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
  "masterSubnetName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-master-subnet')]",
  "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
  "masterPublicIpAddressName" : "[concat(parameters('baseName'), '-master-pip')]",
  "masterPublicIpAddressID" : "[resourceId('Microsoft.Network/publicIPAddresses',
variables('masterPublicIpAddressName'))]",
  "masterLoadBalancerName" : "[parameters('baseName')]",
  "masterLoadBalancerID" : "[resourceId('Microsoft.Network/loadBalancers',
variables('masterLoadBalancerName'))]",
  "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal-lb')]",
  "internalLoadBalancerID" : "[resourceId('Microsoft.Network/loadBalancers',
variables('internalLoadBalancerName'))]",
  "skuName": "Standard"
 },
 "resources" : [
  {
   "apiVersion" : "2018-12-01",
   "type" : "Microsoft.Network/publicIPAddresses",
   "name" : "[variables('masterPublicIpAddressName')]",
   "location" : "[variables('location')]",
   "sku": {
    "name": "[variables('skuName')]"
   },
   "properties" : {
    "publicIPAllocationMethod" : "Static",
    "dnsSettings" : {
     "domainNameLabel" : "[variables('masterPublicIpAddressName')]"
    }
   }
  },
    {
      "apiVersion" : "2018-12-01",
      "type" : "Microsoft.Network/loadBalancers",
      "name" : "[variables('masterLoadBalancerName')]",
      "location" : "[variables('location')]",
      "sku": {
        "name": "[variables('skuName')]"
      },
      "dependsOn" : [
        "[concat('Microsoft.Network/publicIPAddresses/', variables('masterPublicIpAddressName'))]"
      ],
      "properties" : {
        "frontendIPConfigurations" : [
          {
            "name" : "public-lb-ip-v4",
            "properties" : {
              "publicIPAddress" : {
                "id" : "[variables('masterPublicIpAddressID')]"
              }
            }
          }
        ],
        "backendAddressPools" : [
          {
            "name" : "[variables('masterLoadBalancerName')]"
          }
        ],
        "loadBalancingRules" : [
          {
            "name" : "api-internal",
            "properties" : {
              "frontendIPConfiguration" : {
                "id" :"[concat(variables('masterLoadBalancerID'), '/frontendIPConfigurations/public-lb-ip-v4')]"
              },
              "backendAddressPool" : {
                "id" : "[concat(variables('masterLoadBalancerID'), '/backendAddressPools/', variables('masterLoadBalancerName'))]"
              },
              "protocol" : "Tcp",
              "loadDistribution" : "Default",
              "idleTimeoutInMinutes" : 30,
              "frontendPort" : 6443,
              "backendPort" : 6443,
              "probe" : {
                "id" : "[concat(variables('masterLoadBalancerID'), '/probes/api-internal-probe')]"
              }
            }
          }
        ],
        "probes" : [
          {
            "name" : "api-internal-probe",
            "properties" : {
              "protocol" : "Https",
              "port" : 6443,
       "requestPath": "/readyz",
       "intervalInSeconds" : 10,
       "numberOfProbes" : 3
      }
     }
    ]
   }
  },
  {
   "apiVersion" : "2018-12-01",
   "type" : "Microsoft.Network/loadBalancers",
   "name" : "[variables('internalLoadBalancerName')]",
   "location" : "[variables('location')]",
   "sku": {
    "name": "[variables('skuName')]"
   },
   "properties" : {
    "frontendIPConfigurations" : [
     {
      "name" : "internal-lb-ip",
      "properties" : {
       "privateIPAllocationMethod" : "Dynamic",
       "subnet" : {
        "id" : "[variables('masterSubnetRef')]"
       },
       "privateIPAddressVersion" : "IPv4"
      }
     }
    ],
    "backendAddressPools" : [
     {
      "name" : "internal-lb-backend"
     }
    ],
    "loadBalancingRules" : [
     {
      "name" : "api-internal",
      "properties" : {
       "frontendIPConfiguration" : {
        "id" : "[concat(variables('internalLoadBalancerID'), '/frontendIPConfigurations/internal-lb-
ip')]"
       },
       "frontendPort" : 6443,
       "backendPort" : 6443,
       "enableFloatingIP" : false,
       "idleTimeoutInMinutes" : 30,
       "protocol" : "Tcp",
       "enableTcpReset" : false,
       "loadDistribution" : "Default",
       "backendAddressPool" : {
        "id" : "[concat(variables('internalLoadBalancerID'), '/backendAddressPools/internal-lb-
backend')]"
       },
       "probe" : {
        "id" : "[concat(variables('internalLoadBalancerID'), '/probes/api-internal-probe')]"
       }
      }
     },
     {
      "name" : "sint",
      "properties" : {
       "frontendIPConfiguration" : {
        "id" : "[concat(variables('internalLoadBalancerID'), '/frontendIPConfigurations/internal-lb-ip')]"
       },
       "frontendPort" : 22623,
       "backendPort" : 22623,
       "enableFloatingIP" : false,
       "idleTimeoutInMinutes" : 30,
       "protocol" : "Tcp",
       "enableTcpReset" : false,
       "loadDistribution" : "Default",
       "backendAddressPool" : {
        "id" : "[concat(variables('internalLoadBalancerID'), '/backendAddressPools/internal-lb-backend')]"
       },
       "probe" : {
        "id" : "[concat(variables('internalLoadBalancerID'), '/probes/sint-probe')]"
       }
      }
     }
    ],
    "probes" : [
     {
      "name" : "api-internal-probe",
      "properties" : {
       "protocol" : "Https",
       "port" : 6443,
       "requestPath": "/readyz",
       "intervalInSeconds" : 10,
       "numberOfProbes" : 3
      }
     },
     {
      "name" : "sint-probe",
      "properties" : {
       "protocol" : "Https",
       "port" : 22623,
       "requestPath": "/healthz",
       "intervalInSeconds" : 10,
       "numberOfProbes" : 3
      }
     }
    ]
   }
  },
  {
   "apiVersion": "2018-09-01",
   "type": "Microsoft.Network/privateDnsZones/A",
   "name": "[concat(parameters('privateDNSZoneName'), '/api')]",
   "location" : "[variables('location')]",
   "dependsOn" : [
        "[concat('Microsoft.Network/loadBalancers/', variables('internalLoadBalancerName'))]"
      ],
      "properties": {
        "ttl": 60,
        "aRecords": [
          {
            "ipv4Address": "
[reference(variables('internalLoadBalancerName')).frontendIPConfigurations[0].properties.privateIP
Address]"
          }
        ]
      }
    },
    {
      "apiVersion": "2018-09-01",
      "type": "Microsoft.Network/privateDnsZones/A",
      "name": "[concat(parameters('privateDNSZoneName'), '/api-int')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[concat('Microsoft.Network/loadBalancers/', variables('internalLoadBalancerName'))]"
      ],
      "properties": {
        "ttl": 60,
        "aRecords": [
          {
            "ipv4Address": "
[reference(variables('internalLoadBalancerName')).frontendIPConfigurations[0].properties.privateIP
Address]"
          }
        ]
      }
    }
  ]
}
```

4.2.13. Creating the bootstrap machine in Azure

You must create the bootstrap machine in Microsoft Azure to use during OpenShift Container Platform cluster initialization. One way to create this machine is to modify the provided Azure Resource Manager (ARM) template.

NOTE

If you do not use the provided ARM template to create your bootstrap machine, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Create and configure networking and load balancers in Azure.
- Create the Azure identity and grant the appropriate roles.

Procedure

1. Copy the template from the **ARM template for the bootstrap machine**section of this topic and save it as **04_bootstrap.json** in your cluster's installation directory. This template describes the bootstrap machine that your cluster requires.

2. Export the bootstrap URL variable:

```
$ bootstrap_url_expiry=`date -u -d "10 hours" '+%Y-%m-%dT%H:%MZ'`
```

```
$ export BOOTSTRAP_URL=`az storage blob generate-sas -c 'files' -n 'bootstrap.ign' --https-only --full-uri --permissions r --expiry $bootstrap_url_expiry --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} -o tsv`
```

3. Export the bootstrap ignition variable:

```
$ export BOOTSTRAP_IGNITION=`jq -rcnM --arg v "3.2.0" --arg url ${BOOTSTRAP_URL} '{ignition:{version:$v,config:{replace:{source:$url}}}}' | base64 | tr -d '\n'`
```

4. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/04_bootstrap.json" \
  --parameters bootstrapIgnition="${BOOTSTRAP_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameter bootstrapVMSize="Standard_D4s_v3" 3
```

1 The bootstrap Ignition content for the bootstrap cluster.

2 The base name to be used in resource names; this is usually the cluster's infrastructure ID.

3 Optional: Specify the size of the bootstrap VM. Use a VM size compatible with your specified architecture. If this value is not defined, the default value from the template is set.

4.2.13.1. ARM template for the bootstrap machine

You can use the following Azure Resource Manager (ARM) template to deploy the bootstrap machine that you need for your OpenShift Container Platform cluster:

Example 4.25. 04_bootstrap.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vnetBaseName": {
      "type": "string",
      "defaultValue": "",
      "metadata" : {
        "description" : "The specific customer vnet's base name (optional)"
      }
    },
    "bootstrapIgnition" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Bootstrap ignition content for the bootstrap cluster"
      }
    },
    "sshKeyData" : {
      "type" : "securestring",
      "defaultValue" : "Unused",
      "metadata" : {
        "description" : "Unused"
      }
    },
    "bootstrapVMSize" : {
      "type" : "string",
      "defaultValue" : "Standard_D4s_v3",
      "metadata" : {
        "description" : "The size of the Bootstrap Virtual Machine"
      }
    },
    "hyperVGen": {
      "type": "string",
      "metadata": {
        "description": "VM generation image to use"
      },
      "defaultValue": "V2",
      "allowedValues": [
        "V1",
        "V2"
      ]
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "masterSubnetName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-master-subnet')]",
    "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
    "masterLoadBalancerName" : "[parameters('baseName')]",
    "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal-lb')]",
    "sshKeyPath" : "/home/core/.ssh/authorized_keys",
    "identityName" : "[concat(parameters('baseName'), '-identity')]",
    "vmName" : "[concat(parameters('baseName'), '-bootstrap')]",
  "nicName" : "[concat(variables('vmName'), '-nic')]",
  "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
  "imageName" : "[concat(parameters('baseName'), if(equals(parameters('hyperVGen'), 'V2'), '-
gen2', ''))]",
  "clusterNsgName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-nsg')]",
  "sshPublicIpAddressName" : "[concat(variables('vmName'), '-ssh-pip')]"
 },
 "resources" : [
  {
   "apiVersion" : "2018-12-01",
   "type" : "Microsoft.Network/publicIPAddresses",
   "name" : "[variables('sshPublicIpAddressName')]",
   "location" : "[variables('location')]",
   "sku": {
    "name": "Standard"
   },
   "properties" : {
    "publicIPAllocationMethod" : "Static",
    "dnsSettings" : {
     "domainNameLabel" : "[variables('sshPublicIpAddressName')]"
    }
   }
  },
  {
   "apiVersion" : "2018-06-01",
   "type" : "Microsoft.Network/networkInterfaces",
   "name" : "[variables('nicName')]",
   "location" : "[variables('location')]",
   "dependsOn" : [
    "[resourceId('Microsoft.Network/publicIPAddresses', variables('sshPublicIpAddressName'))]"
   ],
   "properties" : {
    "ipConfigurations" : [
     {
      "name" : "pipConfig",
      "properties" : {
       "privateIPAllocationMethod" : "Dynamic",
       "publicIPAddress": {
        "id": "[resourceId('Microsoft.Network/publicIPAddresses',
variables('sshPublicIpAddressName'))]"
       },
       "subnet" : {
        "id" : "[variables('masterSubnetRef')]"
       },
       "loadBalancerBackendAddressPools" : [
        {
         "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('masterLoadBalancerName'), '/backendAddressPools/',
variables('masterLoadBalancerName'))]"
        },
        {
         "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('internalLoadBalancerName'), '/backendAddressPools/internal-lb-backend')]"
```

```
                }
              ]
            }
          }
        ]
      }
    },
    {
      "apiVersion" : "2018-06-01",
      "type" : "Microsoft.Compute/virtualMachines",
      "name" : "[variables('vmName')]",
      "location" : "[variables('location')]",
      "identity" : {
        "type" : "userAssigned",
        "userAssignedIdentities" : {
          "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',
variables('identityName'))]" : {}
        }
      },
      "dependsOn" : [
        "[concat('Microsoft.Network/networkInterfaces/', variables('nicName'))]"
      ],
      "properties" : {
        "hardwareProfile" : {
          "vmSize" : "[parameters('bootstrapVMSize')]"
        },
        "osProfile" : {
          "computerName" : "[variables('vmName')]",
          "adminUsername" : "core",
          "adminPassword" : "NotActuallyApplied!",
          "customData" : "[parameters('bootstrapIgnition')]",
          "linuxConfiguration" : {
            "disablePasswordAuthentication" : false
          }
        },
        "storageProfile" : {
          "imageReference": {
            "id": "[resourceId('Microsoft.Compute/galleries/images', variables('galleryName'),
variables('imageName'))]"
          },
          "osDisk" : {
            "name": "[concat(variables('vmName'),'_OSDisk')]",
            "osType" : "Linux",
            "createOption" : "FromImage",
            "managedDisk": {
              "storageAccountType": "Premium_LRS"
            },
            "diskSizeGB" : 100
          }
        },
        "networkProfile" : {
          "networkInterfaces" : [
            {
              "id" : "[resourceId('Microsoft.Network/networkInterfaces', variables('nicName'))]"
            }
          ]
        }
      }
    },
    {
      "apiVersion" : "2018-06-01",
      "type": "Microsoft.Network/networkSecurityGroups/securityRules",
      "name" : "[concat(variables('clusterNsgName'), '/bootstrap_ssh_in')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[resourceId('Microsoft.Compute/virtualMachines', variables('vmName'))]"
      ],
      "properties": {
        "protocol" : "Tcp",
        "sourcePortRange" : "*",
        "destinationPortRange" : "22",
        "sourceAddressPrefix" : "*",
        "destinationAddressPrefix" : "*",
        "access" : "Allow",
        "priority" : 100,
        "direction" : "Inbound"
      }
    }
  ]
}
```

4.2.14. Creating the control plane machines in Azure

You must create the control plane machines in Microsoft Azure for your cluster to use. One way to create these machines is to modify the provided Azure Resource Manager (ARM) template.

NOTE

By default, Microsoft Azure places control plane machines and compute machines in a pre-set availability zone. You can manually set an availability zone for a compute node or control plane node. To do this, modify a vendor's Azure Resource Manager (ARM) template by specifying each of your availability zones in the **zones** parameter of the virtual machine resource.

If you do not use the provided ARM template to create your control plane machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, consider contacting Red Hat support with your installation logs.

Prerequisites

- Create the bootstrap machine.

Procedure

1. Copy the template from the **ARM template for control plane machines**section of this topic and save it as **05_masters.json** in your cluster's installation directory. This template describes the control plane machines that your cluster requires.

2. Export the following variable needed by the control plane machine deployment:

```
$ export MASTER_IGNITION=`cat <installation_directory>/master.ign | base64 | tr -d '\n'`
```

3. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/05_masters.json" \
  --parameters masterIgnition="${MASTER_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameters masterVMSize="Standard_D8s_v3" 3
```

1 The Ignition content for the control plane nodes.

2 The base name to be used in resource names; this is usually the cluster’s infrastructure ID.

3 Optional: Specify the size of the Control Plane VM. Use a VM size compatible with your specified architecture. If this value is not defined, the default value from the template is set.

4.2.14.1. ARM template for control plane machines

You can use the following Azure Resource Manager (ARM) template to deploy the control plane machines that you need for your OpenShift Container Platform cluster:

Example 4.26. 05_masters.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vnetBaseName": {
      "type": "string",
      "defaultValue": "",
      "metadata" : {
        "description" : "The specific customer vnet's base name (optional)"
      }
    },
    "masterIgnition" : {
      "type" : "string",
      "metadata" : {
        "description" : "Ignition content for the master nodes"
      }
    },
    "numberOfMasters" : {
      "type" : "int",
      "defaultValue" : 3,
   "minValue" : 2,
   "maxValue" : 30,
   "metadata" : {
    "description" : "Number of OpenShift masters to deploy"
   }
  },
  "sshKeyData" : {
   "type" : "securestring",
   "defaultValue" : "Unused",
   "metadata" : {
    "description" : "Unused"
   }
  },
  "privateDNSZoneName" : {
   "type" : "string",
   "defaultValue" : "",
   "metadata" : {
    "description" : "unused"
   }
  },
  "masterVMSize" : {
   "type" : "string",
   "defaultValue" : "Standard_D8s_v3",
   "metadata" : {
    "description" : "The size of the Master Virtual Machines"
   }
  },
  "diskSizeGB" : {
   "type" : "int",
   "defaultValue" : 1024,
   "metadata" : {
    "description" : "Size of the Master VM OS disk, in GB"
   }
  },
  "hyperVGen": {
   "type": "string",
   "metadata": {
    "description": "VM generation image to use"
   },
   "defaultValue": "V2",
   "allowedValues": [
    "V1",
    "V2"
   ]
  }
 },
 "variables" : {
  "location" : "[resourceGroup().location]",
  "virtualNetworkName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
  "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
  "masterSubnetName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-master-subnet')]",
  "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
  "masterLoadBalancerName" : "[parameters('baseName')]",
  "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal-lb')]",
  "sshKeyPath" : "/home/core/.ssh/authorized_keys",
  "identityName" : "[concat(parameters('baseName'), '-identity')]",
  "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
  "imageName" : "[concat(parameters('baseName'), if(equals(parameters('hyperVGen'), 'V2'), '-
gen2', ''))]",
  "copy" : [
   {
    "name" : "vmNames",
    "count" :  "[parameters('numberOfMasters')]",
    "input" : "[concat(parameters('baseName'), '-master-', copyIndex('vmNames'))]"
   }
  ]
 },
 "resources" : [
  {
   "apiVersion" : "2018-06-01",
   "type" : "Microsoft.Network/networkInterfaces",
   "copy" : {
    "name" : "nicCopy",
    "count" : "[length(variables('vmNames'))]"
   },
   "name" : "[concat(variables('vmNames')[copyIndex()], '-nic')]",
   "location" : "[variables('location')]",
   "properties" : {
    "ipConfigurations" : [
     {
      "name" : "pipConfig",
      "properties" : {
       "privateIPAllocationMethod" : "Dynamic",
       "subnet" : {
        "id" : "[variables('masterSubnetRef')]"
       },
       "loadBalancerBackendAddressPools" : [
        {
         "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('masterLoadBalancerName'), '/backendAddressPools/',
variables('masterLoadBalancerName'))]"
        },
        {
         "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('internalLoadBalancerName'), '/backendAddressPools/internal-lb-backend')]"
        }
       ]
      }
     }
    ]
   }
  },
  {
   "apiVersion" : "2018-06-01",
   "type" : "Microsoft.Compute/virtualMachines",
   "copy" : {
        "name" : "vmCopy",
        "count" : "[length(variables('vmNames'))]"
      },
      "name" : "[variables('vmNames')[copyIndex()]]",
      "location" : "[variables('location')]",
      "identity" : {
        "type" : "userAssigned",
        "userAssignedIdentities" : {
          "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',
variables('identityName'))]" : {}
        }
      },
      "dependsOn" : [
        "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')[copyIndex()], '-
nic'))]"
      ],
      "properties" : {
        "hardwareProfile" : {
          "vmSize" : "[parameters('masterVMSize')]"
        },
        "osProfile" : {
          "computerName" : "[variables('vmNames')[copyIndex()]]",
          "adminUsername" : "core",
          "adminPassword" : "NotActuallyApplied!",
          "customData" : "[parameters('masterIgnition')]",
          "linuxConfiguration" : {
            "disablePasswordAuthentication" : false
          }
        },
        "storageProfile" : {
          "imageReference": {
            "id": "[resourceId('Microsoft.Compute/galleries/images', variables('galleryName'),
variables('imageName'))]"
          },
          "osDisk" : {
            "name": "[concat(variables('vmNames')[copyIndex()], '_OSDisk')]",
            "osType" : "Linux",
            "createOption" : "FromImage",
            "caching": "ReadOnly",
            "writeAcceleratorEnabled": false,
            "managedDisk": {
              "storageAccountType": "Premium_LRS"
            },
            "diskSizeGB" : "[parameters('diskSizeGB')]"
          }
        },
        "networkProfile" : {
          "networkInterfaces" : [
            {
              "id" : "[resourceId('Microsoft.Network/networkInterfaces', concat(variables('vmNames')
[copyIndex()], '-nic'))]",
              "properties": {
                "primary": false
              }
            }
          ]
        }
      }
    }
  ]
}
```

4.2.15. Wait for bootstrap completion and remove bootstrap resources in Azure

After you create all of the required infrastructure in Microsoft Azure, wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

Prerequisites

- Create the control plane machines.

Procedure

1. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ ①
    --log-level info ②
```

① For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

② To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

If the command exits without a **FATAL** warning, your production control plane has initialized.

2. Delete the bootstrap resources:

```
$ az network nsg rule delete -g ${RESOURCE_GROUP} --nsg-name ${INFRA_ID}-nsg --name bootstrap_ssh_in
$ az vm stop -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
$ az vm deallocate -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
$ az vm delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap --yes
$ az disk delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap_OSDisk --no-wait --yes
$ az network nic delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-nic --no-wait
$ az storage blob delete --account-key ${ACCOUNT_KEY} --account-name ${CLUSTER_NAME}sa --container-name files --name bootstrap.ign
$ az network public-ip delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-ssh-pip
```

NOTE

If you do not delete the bootstrap server, installation may not succeed due to API traffic being routed to the bootstrap server.

4.2.16. Creating additional worker machines in Azure

You can create worker machines in Microsoft Azure for your cluster to use by launching individual instances discretely or by automated processes outside the cluster, such as auto scaling groups. You can also take advantage of the built-in cluster scaling mechanisms and the machine API in OpenShift Container Platform.

In this example, you manually launch one instance by using the Azure Resource Manager (ARM) template. Additional instances can be launched by including additional resources of type **06_workers.json** in the file.

> NOTE
>
> By default, Microsoft Azure places control plane machines and compute machines in a pre-set availability zone. You can manually set an availability zone for a compute node or control plane node. To do this, modify a vendor's ARM template by specifying each of your availability zones in the **zones** parameter of the virtual machine resource.

If you do not use the provided ARM template to create your control plane machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, consider contacting Red Hat support with your installation logs.

Procedure

1. Copy the template from the **ARM template for worker machines** section of this topic and save it as **06_workers.json** in your cluster's installation directory. This template describes the worker machines that your cluster requires.

2. Export the following variable needed by the worker machine deployment:

```
$ export WORKER_IGNITION=`cat <installation_directory>/worker.ign | base64 | tr -d '\n'`
```

3. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/06_workers.json" \
  --parameters workerIgnition="${WORKER_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameters nodeVMSize="Standard_D4s_v3" 3
```

1 The Ignition content for the worker nodes.

2 The base name to be used in resource names; this is usually the cluster's infrastructure ID.

3 Optional: Specify the size of the compute node VM. Use a VM size compatible with your specified architecture. If this value is not defined, the default value from the template is set.

4.2.16.1. ARM template for worker machines

You can use the following Azure Resource Manager (ARM) template to deploy the worker machines that you need for your OpenShift Container Platform cluster:

Example 4.27. 06_workers.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vnetBaseName": {
      "type": "string",
      "defaultValue": "",
      "metadata" : {
        "description" : "The specific customer vnet's base name (optional)"
      }
    },
    "workerIgnition" : {
      "type" : "string",
      "metadata" : {
        "description" : "Ignition content for the worker nodes"
      }
    },
    "numberOfNodes" : {
      "type" : "int",
      "defaultValue" : 3,
      "minValue" : 2,
      "maxValue" : 30,
      "metadata" : {
        "description" : "Number of OpenShift compute nodes to deploy"
      }
    },
    "sshKeyData" : {
      "type" : "securestring",
      "defaultValue" : "Unused",
      "metadata" : {
        "description" : "Unused"
      }
    },
    "nodeVMSize" : {
      "type" : "string",
      "defaultValue" : "Standard_D4s_v3",
      "metadata" : {
        "description" : "The size of the each Node Virtual Machine"
      }
    },
    "hyperVGen": {
      "type": "string",
      "metadata": {
        "description": "VM generation image to use"
      },
      "defaultValue": "V2",
      "allowedValues": [
        "V1",
        "V2"
      ]
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "nodeSubnetName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-worker-subnet')]",
    "nodeSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('nodeSubnetName'))]",
    "infraLoadBalancerName" : "[parameters('baseName')]",
    "sshKeyPath" : "/home/capi/.ssh/authorized_keys",
    "identityName" : "[concat(parameters('baseName'), '-identity')]",
    "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
    "imageName" : "[concat(parameters('baseName'), if(equals(parameters('hyperVGen'), 'V2'), '-
gen2', ''))]",
    "copy" : [
      {
        "name" : "vmNames",
        "count" :  "[parameters('numberOfNodes')]",
        "input" : "[concat(parameters('baseName'), '-worker-', variables('location'), '-',
copyIndex('vmNames', 1))]"
      }
    ]
  },
  "resources" : [
    {
      "apiVersion" : "2019-05-01",
      "name" : "[concat('node', copyIndex())]",
      "type" : "Microsoft.Resources/deployments",
      "copy" : {
        "name" : "nodeCopy",
        "count" : "[length(variables('vmNames'))]"
      },
      "properties" : {
        "mode" : "Incremental",
        "template" : {
          "$schema" : "http://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
          "contentVersion" : "1.0.0.0",
          "resources" : [
            {
              "apiVersion" : "2018-06-01",
              "type" : "Microsoft.Network/networkInterfaces",
              "name" : "[concat(variables('vmNames')[copyIndex()], '-nic')]",
              "location" : "[variables('location')]",
              "properties" : {
                "ipConfigurations" : [
                  {
                    "name" : "pipConfig",
                    "properties" : {
                      "privateIPAllocationMethod" : "Dynamic",
                      "subnet" : {
                        "id" : "[variables('nodeSubnetRef')]"
                      }
                    }
                  }
                ]
              }
            },
            {
              "apiVersion" : "2018-06-01",
              "type" : "Microsoft.Compute/virtualMachines",
              "name" : "[variables('vmNames')[copyIndex()]]",
              "location" : "[variables('location')]",
              "tags" : {
                "kubernetes.io-cluster-ffranzupi": "owned"
              },
              "identity" : {
                "type" : "userAssigned",
                "userAssignedIdentities" : {
                  "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',
variables('identityName'))]" : {}
                }
              },
              "dependsOn" : [
                "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')
[copyIndex()], '-nic'))]"
              ],
              "properties" : {
                "hardwareProfile" : {
                  "vmSize" : "[parameters('nodeVMSize')]"
                },
                "osProfile" : {
                  "computerName" : "[variables('vmNames')[copyIndex()]]",
                  "adminUsername" : "capi",
                  "adminPassword" : "NotActuallyApplied!",
                  "customData" : "[parameters('workerIgnition')]",
                  "linuxConfiguration" : {
                    "disablePasswordAuthentication" : false
                  }
                },
                "storageProfile" : {
                  "imageReference": {
                    "id": "[resourceId('Microsoft.Compute/galleries/images', variables('galleryName'),
variables('imageName'))]"
                  },
                  "osDisk" : {
                    "name": "[concat(variables('vmNames')[copyIndex()],'_OSDisk')]",
                    "osType" : "Linux",
                    "createOption" : "FromImage",
                    "managedDisk": {
                      "storageAccountType": "Premium_LRS"
                    },
                    "diskSizeGB": 128
                  }
                },
                "networkProfile" : {
                  "networkInterfaces" : [
                    {
                      "id" : "[resourceId('Microsoft.Network/networkInterfaces',
concat(variables('vmNames')[copyIndex()], '-nic'))]",
                      "properties": {
                        "primary": true
                      }
                    }
                  ]
                }
              }
            }
          ]
        }
      }
    }
  ]
}
```

4.2.17. Installing the OpenShift CLI

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.

> **IMPORTANT**
>
> If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.18. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux
You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the OpenShift Container Platform downloads page on the Red Hat Customer Portal.
2. Select the architecture from the **Product Variant** drop-down list.
3. Select the appropriate version from the **Version** drop-down list.
4. Click **Download Now** next to the **OpenShift v4.18 Linux Clients** entry and save the file.
5. Unpack the archive:

   ```
   $ tar xvf <file>
   ```

6. Place the **oc** binary in a directory that is on your **PATH**.
   To check your **PATH**, execute the following command:

   ```
   $ echo $PATH
   ```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

  ```
  $ oc <command>
  ```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the OpenShift Container Platform downloads page on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.18 Windows Client** entry and save the file.
4. Unzip the archive with a ZIP program.
5. Move the **oc** binary to a directory that is on your **PATH**.
   To check your **PATH**, open the command prompt and execute the following command:

   ```
   C:\> path
   ```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

  ```
  C:\> oc <command>
  ```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the OpenShift Container Platform downloads page on the Red Hat Customer Portal.
2. Select the appropriate version from the **Version** drop-down list.
3. Click **Download Now** next to the **OpenShift v4.18 macOS Clients** entry and save the file.

   > **NOTE**
   >
   > For macOS arm64, choose the **OpenShift v4.18 macOS arm64 Client** entry.

4. Unpack and unzip the archive.
5. Move the **oc** binary to a directory on your PATH.
   To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- Verify your installation by using an **oc** command:

```
$ oc <command>
```

4.2.18. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

4.2.19. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.31.3
master-1  Ready     master  63m  v1.31.3
master-2  Ready     master  64m  v1.31.3
```

The output lists all of the machines that you created.

NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME        AGE     REQUESTOR                                                                   CONDITION
csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

  ```
  $ oc adm certificate approve <csr_name> 1
  ```

  1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

  ```
  $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
  {{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
  ```

  NOTE

  Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

   ```
   $ oc get csr
   ```

   Example output

   ```
   NAME        AGE     REQUESTOR                                                  CONDITION
   csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
   Pending
   csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
   Pending
   ...
   ```

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

   - To approve them individually, run the following command for each valid CSR:

     ```
     $ oc adm certificate approve <csr_name> 1
     ```

     1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

  ```
  $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
  {{end}}{{end}}' | xargs oc adm certificate approve
  ```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

   ```
   $ oc get nodes
   ```

   Example output

   ```
   NAME      STATUS   ROLES   AGE  VERSION
   master-0  Ready    master  73m  v1.31.3
   master-1  Ready    master  73m  v1.31.3
   master-2  Ready    master  74m  v1.31.3
   worker-0  Ready    worker  11m  v1.31.3
   worker-1  Ready    worker  11m  v1.31.3
   ```

   > **NOTE**
   >
   > It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- Certificate Signing Requests

4.2.20. Adding the Ingress DNS records

If you removed the DNS Zone configuration when creating Kubernetes manifests and generating Ignition configs, you must manually create DNS records that point at the Ingress load balancer. You can create either a wildcard ***.apps.{baseDomain}.** or specific records. You can use A, CNAME, and other records per your requirements.

Prerequisites

- You deployed an OpenShift Container Platform cluster on Microsoft Azure by using infrastructure that you provisioned.
- Install the OpenShift CLI (**oc**).
- Install or update the Azure CLI.

Procedure

1. Confirm the Ingress router has created a load balancer and populated the **EXTERNAL-IP** field:

   ```
   $ oc -n openshift-ingress get service router-default
   ```

   Example output

```
NAME           TYPE          CLUSTER-IP     EXTERNAL-IP    PORT(S)                AGE
router-default   LoadBalancer   172.30.20.10   35.130.120.110
80:32288/TCP,443:31215/TCP   20
```

2. Export the Ingress router IP as a variable:

```
$ export PUBLIC_IP_ROUTER=`oc -n openshift-ingress get service router-default --no-
headers | awk '{print $4}'`
```

3. Add a ***.apps** record to the public DNS zone.

   a. If you are adding this cluster to a new public zone, run:

   ```
   $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
   z ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER} --ttl 300
   ```

   b. If you are adding this cluster to an already existing public zone, run:

   ```
   $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
   z ${BASE_DOMAIN} -n *.apps.${CLUSTER_NAME} -a ${PUBLIC_IP_ROUTER} --ttl 300
   ```

4. Add a ***.apps** record to the private DNS zone:

   a. Create a ***.apps** record by using the following command:

   ```
   $ az network private-dns record-set a create -g ${RESOURCE_GROUP} -z
   ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps --ttl 300
   ```

   b. Add the ***.apps** record to the private DNS zone by using the following command:

   ```
   $ az network private-dns record-set a add-record -g ${RESOURCE_GROUP} -z
   ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER}
   ```

If you prefer to add explicit domains instead of using a wildcard, you can create entries for each of the cluster's current routes:

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}{"\n"}{end}
{end}' routes
```

Example output

```
oauth-openshift.apps.cluster.basedomain.com
console-openshift-console.apps.cluster.basedomain.com
downloads-openshift-console.apps.cluster.basedomain.com
alertmanager-main-openshift-monitoring.apps.cluster.basedomain.com
prometheus-k8s-openshift-monitoring.apps.cluster.basedomain.com
```

4.2.21. Completing an Azure installation on user-provisioned infrastructure

After you start the OpenShift Container Platform installation on Microsoft Azure user-provisioned infrastructure, you can monitor the cluster events until the cluster is ready.

Prerequisites

- Deploy the bootstrap machine for an OpenShift Container Platform cluster on user-provisioned Azure infrastructure.
- Install the **oc** CLI and log in.

Procedure

- Complete the cluster installation:

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

Example output

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

4.2.22. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.18, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

4.3. INSTALLING A CLUSTER ON AZURE USING ARM TEMPLATES

In OpenShift Container Platform version 4.18, you can install a cluster on Microsoft Azure by using infrastructure that you provide.

Several Azure Resource Manager (ARM) templates are provided to assist in completing these steps or to help model your own.

IMPORTANT

The steps for performing a user-provisioned infrastructure installation are provided as an example only. Installing a cluster with infrastructure you provide requires knowledge of the cloud provider and the installation process of OpenShift Container Platform. Several ARM templates are provided to assist in completing these steps or to help model your own. You are also free to create the required resources through other methods; the templates are just an example.

4.3.1. Prerequisites

- You reviewed details about the OpenShift Container Platform installation and update processes.

- You read the documentation on selecting a cluster installation method and preparing it for users.

- You configured an Azure account to host the cluster.

- You downloaded the Azure CLI and installed it on your computer. See Install the Azure CLI in the Azure documentation. The following documentation was last tested using version **2.49.0** of the Azure CLI. Azure CLI commands might perform differently based on the version you use.

- If the cloud identity and access management (IAM) APIs are not accessible in your environment, or if you do not want to store an administrator-level credential secret in the **kube-system** namespace, see Alternatives to storing administrator-level secrets in the kube-system project .

- If you use a firewall and plan to use the Telemetry service, you configured the firewall to allow the sites that your cluster requires access to.

  NOTE

  Be sure to also review this site list if you are configuring a proxy.

4.3.2. Internet access for OpenShift Container Platform

In OpenShift Container Platform 4.18, you require access to the internet to install your cluster.

You must have internet access to:

- Access OpenShift Cluster Manager to download the installation program and perform subscription management. If the cluster has internet access and you do not disable Telemetry, that service automatically entitles your cluster.

- Access Quay.io to obtain the packages that are required to install your cluster.

- Obtain the packages that are required to perform cluster updates.

IMPORTANT

If your cluster cannot have direct internet access, you can perform a restricted network installation on some types of infrastructure that you provision. During that process, you download the required content and use it to populate a mirror registry with the installation packages. With some installation types, the environment that you install your cluster in will not require internet access. Before you update the cluster, you update the content of the mirror registry.

4.3.3. Configuring your Azure project

Before you can install OpenShift Container Platform, you must configure an Azure project to host it.

IMPORTANT

All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see Resolve reserved resource name errors in the Azure documentation.

4.3.3.1. Azure account limits

The OpenShift Container Platform cluster uses a number of Microsoft Azure components, and the default Azure subscription and service limits, quotas, and constraints affect your ability to install OpenShift Container Platform clusters.

IMPORTANT

Default limits vary by offer category types, such as Free Trial and Pay-As-You-Go, and by series, such as Dv2, F, and G. For example, the default for Enterprise Agreement subscriptions is 350 cores.

Check the limits for your subscription type and if necessary, increase quota limits for your account before you install a default cluster on Azure.

The following table summarizes the Azure components whose limits can impact your ability to install and run OpenShift Container Platform clusters.

Component	Number of components required by default	Default Azure limit	Description

Component	Number of components required by default	Default Azure limit	Description
vCPU	40	20 per region	A default cluster requires 40 vCPUs, so you must increase the account limit. By default, each cluster creates the following instances: ● One bootstrap machine, which is removed after installation ● Three control plane machines ● Three compute machines Because the bootstrap machine uses **Standard_D4s_v3** machines, which use 4 vCPUs, the control plane machines use **Standard_D8s_v3** virtual machines, which use 8 vCPUs, and the worker machines use **Standard_D4s_v3** virtual machines, which use 4 vCPUs, a default cluster requires 40 vCPUs. The bootstrap node VM, which uses 4 vCPUs, is used only during installation. To deploy more worker nodes, enable autoscaling, deploy large workloads, or use a different instance type, you must further increase the vCPU limit for your account to ensure that your cluster can deploy the machines that you require.
OS Disk	7		Each cluster machine must have a minimum of 100 GB of storage and 300 IOPS. NOTE Faster storage is recommended for production clusters and clusters with intensive workloads. For more information about optimizing storage for performance, see the page titled "Optimizing storage" in the "Scalability and performance" section.
VNet	1	1000 per region	Each default cluster requires one Virtual Network (VNet), which contains two subnets.
Network interfaces	7	65,536 per region	Each default cluster requires seven network interfaces. If you create more machines or your deployed workloads create load balancers, your cluster uses more network interfaces.

<table>
<tr><th>Component</th><th>Number of components required by default</th><th>Default Azure limit</th><th>Description</th></tr>
<tr><td>Network security groups</td><td>2</td><td>5000</td><td>Each cluster creates network security groups for each subnet in the VNet. The default cluster creates network security groups for the control plane and for the compute node subnets:
<table>
<tr><td><b>controlplane</b></td><td>Allows the control plane machines to be reached on port 6443 from anywhere</td></tr>
<tr><td><b>node</b></td><td>Allows worker nodes to be reached from the internet on ports 80 and 443</td></tr>
</table></td></tr>
<tr><td>Network load balancers</td><td>3</td><td>1000 per region</td><td>Each cluster creates the following load balancers:
<table>
<tr><td><b>default</b></td><td>Public IP address that load balances requests to ports 80 and 443 across worker machines</td></tr>
<tr><td><b>internal</b></td><td>Private IP address that load balances requests to ports 6443 and 22623 across control plane machines</td></tr>
<tr><td><b>external</b></td><td>Public IP address that load balances requests to port 6443 across control plane machines</td></tr>
</table>
If your applications create more Kubernetes <b>LoadBalancer</b> service objects, your cluster uses more load balancers.</td></tr>
<tr><td>Public IP addresses</td><td>3</td><td></td><td>Each of the two public load balancers uses a public IP address. The bootstrap machine also uses a public IP address so that you can SSH into the machine to troubleshoot issues during installation. The IP address for the bootstrap node is used only during installation.</td></tr>
<tr><td>Private IP addresses</td><td>7</td><td></td><td>The internal load balancer, each of the three control plane machines, and each of the three worker machines each use a private IP address.</td></tr>
</table>

Component	Number of components required by default	Default Azure limit	Description
Spot VM vCPUs (optional)	0 If you configure spot VMs, your cluster must have two spot VM vCPUs for every compute node.	20 per region	This is an optional component. To use spot VMs, you must increase the Azure default limit to at least twice the number of compute nodes in your cluster. **NOTE** Using spot VMs for control plane nodes is not recommended.

To increase an account limit, file a support request on the Azure portal. For more information, see Request a quota limit increase for Azure Deployment Environments resources .

Additional resources

- Optimizing storage

4.3.3.2. Configuring a public DNS zone in Azure

To install OpenShift Container Platform, the Microsoft Azure account you use must have a dedicated public hosted DNS zone in your account. This zone must be authoritative for the domain. This service provides cluster DNS resolution and name lookup for external connections to the cluster.

Procedure

1. Identify your domain, or subdomain, and registrar. You can transfer an existing domain and registrar or obtain a new one through Azure or another source.
   - To purchase a new domain through Azure, see Buy a custom domain name for Azure App Service.
   - If you are using an existing domain and registrar, migrate its DNS to Azure. For more information, see Migrate an active DNS name to Azure App Service in the Azure documentation.
2. Configure DNS for your domain, which includes creating a public hosted zone for your domain or subdomain, extracting the new authoritative name servers, and updating the registrar records for the name servers that your domain uses. For more information, see Tutorial: Host your domain in Azure DNS.
   Use an appropriate root domain, such as **openshiftcorp.com**, or subdomain, such as **clusters.openshiftcorp.com**.
3. If you use a subdomain, follow your organization's procedures to add its delegation records to the parent domain.

You can view Azure's DNS solution by visiting this example for creating DNS zones .

4.3.3.3. Certificate signing requests management

Because your cluster has limited access to automatic machine management when you use infrastructure that you provision, you must provide a mechanism for approving cluster certificate signing requests (CSRs) after installation. The **kube-controller-manager** only approves the kubelet client CSRs. The **machine-approver** cannot guarantee the validity of a serving certificate that is requested by using kubelet credentials because it cannot confirm that the correct machine issued the request. You must determine and implement a method of verifying the validity of the kubelet serving certificate requests and approving them.

4.3.3.4. Recording the subscription and tenant IDs

The installation program requires the subscription and tenant IDs that are associated with your Azure account. You can use the Azure CLI to gather this information.

Prerequisites

- You have installed or updated the Azure CLI.

Procedure

1. Log in to the Azure CLI by running the following command:

```
$ az login
```

2. Ensure that you are using the right subscription:

   a. View a list of available subscriptions by running the following command:

```
$ az account list --refresh
```

Example output

```
[
  {
    "cloudName": "AzureCloud",
    "id": "8xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "isDefault": true,
    "name": "Subscription Name 1",
    "state": "Enabled",
    "tenantId": "6xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "user": {
      "name": "you@example.com",
      "type": "user"
    }
  },
  {
    "cloudName": "AzureCloud",
    "id": "9xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "isDefault": false,
    "name": "Subscription Name 2",
    "state": "Enabled",
    "tenantId": "7xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "user": {
      "name": "you2@example.com",
      "type": "user"
    }
  }
]
```

b. View the details of the active account, and confirm that this is the subscription you want to use, by running the following command:

```
$ az account show
```

Example output

```
{
  "environmentName": "AzureCloud",
  "id": "8xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
  "isDefault": true,
  "name": "Subscription Name 1",
  "state": "Enabled",
  "tenantId": "6xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

3. If you are not using the right subscription:

a. Change the active subscription by running the following command:

```
$ az account set -s <subscription_id>
```

b. Verify that you are using the subscription you need by running the following command:

```
$ az account show
```

Example output

```
{
  "environmentName": "AzureCloud",
  "id": "9xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
  "isDefault": true,
  "name": "Subscription Name 2",
  "state": "Enabled",
  "tenantId": "7xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
  "user": {
    "name": "you2@example.com",
    "type": "user"
  }
}
```

4. Record the **id** and **tenantId** parameter values from the output. You require these values to install an OpenShift Container Platform cluster.

4.3.3.5. Supported identities to access Azure resources

An OpenShift Container Platform cluster requires an Azure identity to create and manage Azure resources. You need one of the following types of identities to complete the installation:

- A service principal
- A system-assigned managed identity
- A user-assigned managed identity

For more information on Azure identities, see Managed identity types.

4.3.3.6. Required Azure permissions for user-provisioned infrastructure

The installation program requires access to an Azure service principal or managed identity with the necessary permissions to deploy the cluster and to maintain its daily operation. These permissions must be granted to the Azure subscription that is associated with the identity.

The following options are available to you:

- You can assign the identity the **Contributor** and **User Access Administrator** roles. Assigning these roles is the quickest way to grant all of the required permissions.
  For more information about assigning roles, see the Azure documentation for managing access to Azure resources using the Azure portal.
- If your organization's security policies require a more restrictive set of permissions, you can create a custom role with the necessary permissions.

The following permissions are required for creating an OpenShift Container Platform cluster on Microsoft Azure.

Example 4.28. Required permissions for creating authorization resources

- **Microsoft.Authorization/policies/audit/action**
- **Microsoft.Authorization/policies/auditIfNotExists/action**
- **Microsoft.Authorization/roleAssignments/read**
- **Microsoft.Authorization/roleAssignments/write**

Example 4.29. Required permissions for creating compute resources

- **Microsoft.Compute/images/read**
- **Microsoft.Compute/images/write**
- **Microsoft.Compute/images/delete**
- **Microsoft.Compute/availabilitySets/read**
- **Microsoft.Compute/disks/beginGetAccess/action**
- **Microsoft.Compute/disks/delete**
- **Microsoft.Compute/disks/read**

- **Microsoft.Compute/disks/write**
- **Microsoft.Compute/galleries/images/read**
- **Microsoft.Compute/galleries/images/versions/read**
- **Microsoft.Compute/galleries/images/versions/write**
- **Microsoft.Compute/galleries/images/write**
- **Microsoft.Compute/galleries/read**
- **Microsoft.Compute/galleries/write**
- **Microsoft.Compute/snapshots/read**
- **Microsoft.Compute/snapshots/write**
- **Microsoft.Compute/snapshots/delete**
- **Microsoft.Compute/virtualMachines/delete**
- **Microsoft.Compute/virtualMachines/powerOff/action**
- **Microsoft.Compute/virtualMachines/read**
- **Microsoft.Compute/virtualMachines/write**
- **Microsoft.Compute/virtualMachines/deallocate/action**

Example 4.30. Required permissions for creating identity management resources

- **Microsoft.ManagedIdentity/userAssignedIdentities/assign/action**
- **Microsoft.ManagedIdentity/userAssignedIdentities/read**
- **Microsoft.ManagedIdentity/userAssignedIdentities/write**

Example 4.31. Required permissions for creating network resources

- **Microsoft.Network/dnsZones/A/write**
- **Microsoft.Network/dnsZones/CNAME/write**
- **Microsoft.Network/dnszones/CNAME/read**
- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/loadBalancers/backendAddressPools/join/action**
- **Microsoft.Network/loadBalancers/backendAddressPools/read**
- **Microsoft.Network/loadBalancers/backendAddressPools/write**
- **Microsoft.Network/loadBalancers/read**

- **Microsoft.Network/loadBalancers/write**
- **Microsoft.Network/networkInterfaces/delete**
- **Microsoft.Network/networkInterfaces/join/action**
- **Microsoft.Network/networkInterfaces/read**
- **Microsoft.Network/networkInterfaces/write**
- **Microsoft.Network/networkSecurityGroups/join/action**
- **Microsoft.Network/networkSecurityGroups/read**
- **Microsoft.Network/networkSecurityGroups/securityRules/delete**
- **Microsoft.Network/networkSecurityGroups/securityRules/read**
- **Microsoft.Network/networkSecurityGroups/securityRules/write**
- **Microsoft.Network/networkSecurityGroups/write**
- **Microsoft.Network/privateDnsZones/A/read**
- **Microsoft.Network/privateDnsZones/A/write**
- **Microsoft.Network/privateDnsZones/A/delete**
- **Microsoft.Network/privateDnsZones/SOA/read**
- **Microsoft.Network/privateDnsZones/read**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/read**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/write**
- **Microsoft.Network/privateDnsZones/write**
- **Microsoft.Network/publicIPAddresses/delete**
- **Microsoft.Network/publicIPAddresses/join/action**
- **Microsoft.Network/publicIPAddresses/read**
- **Microsoft.Network/publicIPAddresses/write**
- **Microsoft.Network/virtualNetworks/join/action**
- **Microsoft.Network/virtualNetworks/read**
- **Microsoft.Network/virtualNetworks/subnets/join/action**
- **Microsoft.Network/virtualNetworks/subnets/read**
- **Microsoft.Network/virtualNetworks/subnets/write**
- **Microsoft.Network/virtualNetworks/write**

Example 4.32. Required permissions for checking the health of resources

- **Microsoft.Resourcehealth/healthevent/Activated/action**
- **Microsoft.Resourcehealth/healthevent/InProgress/action**
- **Microsoft.Resourcehealth/healthevent/Pending/action**
- **Microsoft.Resourcehealth/healthevent/Resolved/action**
- **Microsoft.Resourcehealth/healthevent/Updated/action**

Example 4.33. Required permissions for creating a resource group

- **Microsoft.Resources/subscriptions/resourceGroups/read**
- **Microsoft.Resources/subscriptions/resourcegroups/write**

Example 4.34. Required permissions for creating resource tags

- **Microsoft.Resources/tags/write**

Example 4.35. Required permissions for creating storage resources

- **Microsoft.Storage/storageAccounts/blobServices/read**
- **Microsoft.Storage/storageAccounts/blobServices/containers/write**
- **Microsoft.Storage/storageAccounts/fileServices/read**
- **Microsoft.Storage/storageAccounts/fileServices/shares/read**
- **Microsoft.Storage/storageAccounts/fileServices/shares/write**
- **Microsoft.Storage/storageAccounts/fileServices/shares/delete**
- **Microsoft.Storage/storageAccounts/listKeys/action**
- **Microsoft.Storage/storageAccounts/read**
- **Microsoft.Storage/storageAccounts/write**

Example 4.36. Required permissions for creating deployments

- **Microsoft.Resources/deployments/read**
- **Microsoft.Resources/deployments/write**
- **Microsoft.Resources/deployments/validate/action**
- **Microsoft.Resources/deployments/operationstatuses/read**

Example 4.37. Optional permissions for creating compute resources

- **Microsoft.Compute/availabilitySets/delete**
- **Microsoft.Compute/availabilitySets/write**

Example 4.38. Optional permissions for creating marketplace virtual machine resources

- **Microsoft.MarketplaceOrdering/offertypes/publishers/offers/plans/agreements/read**
- **Microsoft.MarketplaceOrdering/offertypes/publishers/offers/plans/agreements/write**

Example 4.39. Optional permissions for enabling user-managed encryption

- **Microsoft.Compute/diskEncryptionSets/read**
- **Microsoft.Compute/diskEncryptionSets/write**
- **Microsoft.Compute/diskEncryptionSets/delete**
- **Microsoft.KeyVault/vaults/read**
- **Microsoft.KeyVault/vaults/write**
- **Microsoft.KeyVault/vaults/delete**
- **Microsoft.KeyVault/vaults/deploy/action**
- **Microsoft.KeyVault/vaults/keys/read**
- **Microsoft.KeyVault/vaults/keys/write**
- **Microsoft.Features/providers/features/register/action**

The following permissions are required for deleting an OpenShift Container Platform cluster on Microsoft Azure.

Example 4.40. Required permissions for deleting authorization resources

- **Microsoft.Authorization/roleAssignments/delete**

Example 4.41. Required permissions for deleting compute resources

- **Microsoft.Compute/disks/delete**
- **Microsoft.Compute/galleries/delete**
- **Microsoft.Compute/galleries/images/delete**
- **Microsoft.Compute/galleries/images/versions/delete**
- **Microsoft.Compute/virtualMachines/delete**

- **Microsoft.Compute/images/delete**

Example 4.42. Required permissions for deleting identity management resources

- **Microsoft.ManagedIdentity/userAssignedIdentities/delete**

Example 4.43. Required permissions for deleting network resources

- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/dnsZones/A/read**
- **Microsoft.Network/dnsZones/A/delete**
- **Microsoft.Network/dnsZones/CNAME/read**
- **Microsoft.Network/dnsZones/CNAME/delete**
- **Microsoft.Network/loadBalancers/delete**
- **Microsoft.Network/networkInterfaces/delete**
- **Microsoft.Network/networkSecurityGroups/delete**
- **Microsoft.Network/privateDnsZones/read**
- **Microsoft.Network/privateDnsZones/A/read**
- **Microsoft.Network/privateDnsZones/delete**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/delete**
- **Microsoft.Network/publicIPAddresses/delete**
- **Microsoft.Network/virtualNetworks/delete**

Example 4.44. Required permissions for checking the health of resources

- **Microsoft.Resourcehealth/healthevent/Activated/action**
- **Microsoft.Resourcehealth/healthevent/Resolved/action**
- **Microsoft.Resourcehealth/healthevent/Updated/action**

Example 4.45. Required permissions for deleting a resource group

- **Microsoft.Resources/subscriptions/resourcegroups/delete**

Example 4.46. Required permissions for deleting storage resources

- **Microsoft.Storage/storageAccounts/delete**
- **Microsoft.Storage/storageAccounts/listKeys/action**

NOTE

To install OpenShift Container Platform on Azure, you must scope the permissions related to resource group creation to your subscription. After the resource group is created, you can scope the rest of the permissions to the created resource group. If the public DNS zone is present in a different resource group, then the network DNS zone related permissions must always be applied to your subscription.

You can scope all the permissions to your subscription when deleting an OpenShift Container Platform cluster.

4.3.3.7. Using Azure managed identities

The installation program requires an Azure identity to complete the installation. You can use either a system-assigned or user-assigned managed identity.

If you are unable to use a managed identity, you can use a service principal.

Procedure

1. If you are using a system-assigned managed identity, enable it on the virtual machine that you will run the installation program from.
2. If you are using a user-assigned managed identity:
   a. Assign it to the virtual machine that you will run the installation program from.
   b. Record its client ID. You require this value when installing the cluster.
   For more information about viewing the details of a user-assigned managed identity, see List user-assigned managed identities in the Azure documentation.
3. Verify that the required permissions are assigned to the managed identity.

4.3.3.8. Creating a service principal

The installation program requires an Azure identity to complete the installation. You can use a service principal.

If you are unable to use a service principal, you can use a managed identity.

Prerequisites

- You have installed or updated the Azure CLI.
- You have an Azure subscription ID.
- If you are not assigning the **Contributor** and **User Administrator Access** roles to the service principal, you have created a custom role with the required Azure permissions.

Procedure

1. Create the service principal for your account by running the following command:

   ```
   $ az ad sp create-for-rbac --role <role_name> \ ①
        --name <service_principal> \ ②
        --scopes /subscriptions/<subscription_id> ③
   ```

   ① Defines the role name. You can use the **Contributor** role, or you can specify a custom role which contains the necessary permissions.

   ② Defines the service principal name.

   ③ Specifies the subscription ID.

   ### Example output

   ```
   Creating 'Contributor' role assignment under scope '/subscriptions/<subscription_id>'
   The output includes credentials that you must protect. Be sure that you do not
   include these credentials in your code or check the credentials into your source
   control. For more information, see https://aka.ms/azadsp-cli
   {
     "appId": "axxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
     "displayName": <service_principal>",
     "password": "00000000-0000-0000-0000-000000000000",
     "tenantId": "8xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
   }
   ```

   Record the values of the **appId** and **password** parameters from the output. You require these values when installing the cluster.

2. If you assigned the **Contributor** role to your service principal, assign the **User Administrator Access** role by running the following command:

   ```
   $ az role assignment create --role "User Access Administrator" \
     --assignee-object-id $(az ad sp show --id <appId> --query id -o tsv) ①
     --scope /subscriptions/<subscription_id> ②
   ```

   ① Specify the **appId** parameter value for your service principal.

   ② Specifies the subscription ID.

Additional resources

- For more information about CCO modes, see About the Cloud Credential Operator.

4.3.3.9. Supported Azure regions

The installation program dynamically generates the list of available Microsoft Azure regions based on your subscription.

Supported Azure public regions

- **australiacentral** (Australia Central)

- **australiaeast** (Australia East)
- **australiasoutheast** (Australia South East)
- **brazilsouth** (Brazil South)
- **canadacentral** (Canada Central)
- **canadaeast** (Canada East)
- **centralindia** (Central India)
- **centralus** (Central US)
- **chilecentral** (Chile Central)
- **eastasia** (East Asia)
- **eastus** (East US)
- **eastus2** (East US 2)
- **francecentral** (France Central)
- **germanywestcentral** (Germany West Central)
- **indonesiacentral** (Indonesia Central)
- **israelcentral** (Israel Central)
- **italynorth** (Italy North)
- **japaneast** (Japan East)
- **japanwest** (Japan West)
- **koreacentral** (Korea Central)
- **koreasouth** (Korea South)
- **malaysiawest** (Malaysia West)
- **mexicocentral** (Mexico Central)
- **newzealandnorth** (New Zealand North)
- **northcentralus** (North Central US)
- **northeurope** (North Europe)
- **norwayeast** (Norway East)
- **polandcentral** (Poland Central)
- **qatarcentral** (Qatar Central)
- **southafricanorth** (South Africa North)

- **southcentralus** (South Central US)
- **southeastasia** (Southeast Asia)
- **southindia** (South India)
- **spaincentral** (Spain Central)
- **swedencentral** (Sweden Central)
- **switzerlandnorth** (Switzerland North)
- **uaenorth** (UAE North)
- **uksouth** (UK South)
- **ukwest** (UK West)
- **westcentralus** (West Central US)
- **westeurope** (West Europe)
- **westindia** (West India)
- **westus** (West US)
- **westus2** (West US 2)
- **westus3** (West US 3)

Supported Azure Government regions
Support for the following Microsoft Azure Government (MAG) regions was added in OpenShift Container Platform version 4.6:

- **usgovtexas** (US Gov Texas)
- **usgovvirginia** (US Gov Virginia)

You can reference all available MAG regions in the Azure documentation. Other provided MAG regions are expected to work with OpenShift Container Platform, but have not been tested.

4.3.4. Requirements for a cluster with user-provisioned infrastructure

For a cluster that contains user-provisioned infrastructure, you must deploy all of the required machines.

This section describes the requirements for deploying OpenShift Container Platform on user-provisioned infrastructure.

4.3.4.1. Required machines for cluster installation

The smallest OpenShift Container Platform clusters require the following hosts:

Table 4.6. Minimum required hosts

Hosts	Description
One temporary bootstrap machine	The cluster requires the bootstrap machine to deploy the OpenShift Container Platform cluster on the three control plane machines. You can remove the bootstrap machine after you install the cluster.
Three control plane machines	The control plane machines run the Kubernetes and OpenShift Container Platform services that form the control plane.
At least two compute machines, which are also known as worker machines.	The workloads requested by OpenShift Container Platform users run on the compute machines.

IMPORTANT

To maintain high availability of your cluster, use separate physical hosts for these cluster machines.

The bootstrap and control plane machines must use Red Hat Enterprise Linux CoreOS (RHCOS) as the operating system. However, the compute machines can choose between Red Hat Enterprise Linux CoreOS (RHCOS), Red Hat Enterprise Linux (RHEL) 8.6 and later.

Note that RHCOS is based on Red Hat Enterprise Linux (RHEL) 9.2 and inherits all of its hardware certifications and requirements. See Red Hat Enterprise Linux technology capabilities and limits .

4.3.4.2. Minimum resource requirements for cluster installation

Each cluster machine must meet the following minimum requirements:

Table 4.7. Minimum resource requirements

Machine	Operating System	vCPU [1]	Virtual RAM	Storage	Input/Output Per Second (IOPS)[2]
Bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS, RHEL 8.6 and later [3]	2	8 GB	100 GB	300

1. One vCPU is equivalent to one physical core when simultaneous multithreading (SMT), or Hyper-Threading, is not enabled. When enabled, use the following formula to calculate the corresponding ratio: (threads per core × cores) × sockets = vCPUs.

2. OpenShift Container Platform and Kubernetes are sensitive to disk performance, and faster storage is recommended, particularly for etcd on the control plane nodes which require a 10 ms p99 fsync duration. Note that on many cloud platforms, storage size and IOPS scale together, so you might need to over-allocate storage volume to obtain sufficient performance.

3. As with all user-provisioned installations, if you choose to use RHEL compute machines in your cluster, you take responsibility for all operating system life cycle management and maintenance, including performing system updates, applying patches, and completing all other required tasks. Use of RHEL 7 compute machines is deprecated and has been removed in OpenShift Container Platform 4.10 and later.

NOTE

For OpenShift Container Platform version 4.18, RHCOS is based on RHEL version 9.4, which updates the micro-architecture requirements. The following list contains the minimum instruction set architectures (ISA) that each architecture requires:

- x86-64 architecture requires x86-64-v2 ISA
- ARM64 architecture requires ARMv8.0-A ISA
- IBM Power architecture requires Power 9 ISA
- s390x architecture requires z14 ISA

For more information, see Architectures (RHEL documentation).

IMPORTANT

You are required to use Azure virtual machines that have the **premiumIO** parameter set to **true**.

If an instance type for your platform meets the minimum requirements for cluster machines, it is supported to use in OpenShift Container Platform.

Additional resources

- Optimizing storage

4.3.4.3. Tested instance types for Azure

The following Microsoft Azure instance types have been tested with OpenShift Container Platform.

Example 4.47. Machine types based on 64-bit x86 architecture

- **standardBasv2Family**
- **standardBSFamily**
- **standardBsv2Family**
- **standardDADSv5Family**
- **standardDASv4Family**

- **standardDASv5Family**
- **standardDCACCV5Family**
- **standardDCADCCV5Family**
- **standardDCADSv5Family**
- **standardDCASv5Family**
- **standardDCSv3Family**
- **standardDCSv2Family**
- **standardDDCSv3Family**
- **standardDDSv4Family**
- **standardDDSv5Family**
- **standardDLDSv5Family**
- **standardDLSv5Family**
- **standardDSFamily**
- **standardDSv2Family**
- **standardDSv2PromoFamily**
- **standardDSv3Family**
- **standardDSv4Family**
- **standardDSv5Family**
- **standardEADSv5Family**
- **standardEASv4Family**
- **standardEASv5Family**
- **standardEBDSv5Family**
- **standardEBSv5Family**
- **standardECACCV5Family**
- **standardECADCCV5Family**
- **standardECADSv5Family**
- **standardECASv5Family**
- **standardEDSv4Family**
- **standardEDSv5Family**

- **standardEIADSv5Family**
- **standardEIASv4Family**
- **standardEIASv5Family**
- **standardEIBDSv5Family**
- **standardEIBSv5Family**
- **standardEIDSv5Family**
- **standardEISv3Family**
- **standardEISv5Family**
- **standardESv3Family**
- **standardESv4Family**
- **standardESv5Family**
- **standardFXMDVSFamily**
- **standardFSFamily**
- **standardFSv2Family**
- **standardGSFamily**
- **standardHBrsv2Family**
- **standardHBSFamily**
- **standardHBv4Family**
- **standardHCSFamily**
- **standardHXFamily**
- **standardLASv3Family**
- **standardLSFamily**
- **standardLSv2Family**
- **standardLSv3Family**
- **standardMDSHighMemoryv3Family**
- **standardMDSMediumMemoryv2Family**
- **standardMDSMediumMemoryv3Family**
- **standardMIDSHighMemoryv3Family**
- **standardMIDSMediumMemoryv2Family**

- **standardMISHighMemoryv3Family**
- **standardMISMediumMemoryv2Family**
- **standardMSFamily**
- **standardMSHighMemoryv3Family**
- **standardMSMediumMemoryv2Family**
- **standardMSMediumMemoryv3Family**
- **StandardNCADSA100v4Family**
- **Standard NCASv3_T4 Family**
- **standardNCSv3Family**
- **standardNDSv2Family**
- **StandardNGADSV620v1Family**
- **standardNPSFamily**
- **StandardNVADSA10v5Family**
- **standardNVSv3Family**
- **standardXEISv4Family**

4.3.4.4. Tested instance types for Azure on 64-bit ARM infrastructures

The following Microsoft Azure ARM64 instance types have been tested with OpenShift Container Platform.

Example 4.48. Machine types based on 64-bit ARM architecture

- **standardBpsv2Family**
- **standardDPSv5Family**
- **standardDPDSv5Family**
- **standardDPLDSv5Family**
- **standardDPLSv5Family**
- **standardEPSv5Family**
- **standardEPDSv5Family**
- **StandardDpdsv6Family**
- **StandardDpldsv6Famil**
- **StandardDplsv6Family**

- **StandardDpsv6Family**
- **StandardEpdsv6Family**
- **StandardEpsv6Family**

4.3.5. Using the Azure Marketplace offering

Using the Azure Marketplace offering lets you deploy an OpenShift Container Platform cluster, which is billed on pay-per-use basis (hourly, per core) through Azure, while still being supported directly by Red Hat.

To deploy an OpenShift Container Platform cluster using the Azure Marketplace offering, you must first obtain the Azure Marketplace image. The installation program uses this image to deploy worker or control plane nodes. When obtaining your image, consider the following:

- While the images are the same, the Azure Marketplace publisher is different depending on your region. If you are located in North America, specify **redhat** as the publisher. If you are located in EMEA, specify **redhat-limited** as the publisher.
- The offer includes a **rh-ocp-worker** SKU and a **rh-ocp-worker-gen1** SKU. The **rh-ocp-worker** SKU represents a Hyper-V generation version 2 VM image. The default instance types used in OpenShift Container Platform are version 2 compatible. If you plan to use an instance type that is only version 1 compatible, use the image associated with the **rh-ocp-worker-gen1** SKU. The **rh-ocp-worker-gen1** SKU represents a Hyper-V version 1 VM image.

IMPORTANT

Installing images with the Azure marketplace is not supported on clusters with 64-bit ARM instances.

You should only modify the RHCOS image for compute machines to use an Azure Marketplace image. Control plane machines and infrastructure nodes do not require an OpenShift Container Platform subscription and use the public RHCOS default image by default, which does not incur subscription costs on your Azure bill. Therefore, you should not modify the cluster default boot image or the control plane boot images. Applying the Azure Marketplace image to them will incur additional licensing costs that cannot be recovered.

Prerequisites

- You have installed the Azure CLI client **(az)**.
- Your Azure account is entitled for the offer and you have logged into this account with the Azure CLI client.

Procedure

1. Display all of the available OpenShift Container Platform images by running one of the following commands:
   - North America:

     ```
     $ az vm image list --all --offer rh-ocp-worker --publisher redhat -o table
     ```

Example output

```
Offer         Publisher      Sku               Urn                                               Version
------------  -------------  ----------------  --------------------------------------------------------
----  -----------------
rh-ocp-worker  RedHat         rh-ocp-worker      RedHat:rh-ocp-worker:rh-ocp-
worker:4.15.2024072409            4.15.2024072409
rh-ocp-worker  RedHat         rh-ocp-worker-gen1  RedHat:rh-ocp-worker:rh-ocp-worker-
gen1:4.15.2024072409         4.15.2024072409
```

- EMEA:

```
$  az vm image list --all --offer rh-ocp-worker --publisher redhat-limited -o table
```

Example output

```
Offer         Publisher      Sku               Urn
Version
------------  -------------  ----------------  --------------------------------------------------------
----          -----------------
rh-ocp-worker  redhat-limited  rh-ocp-worker      redhat-limited:rh-ocp-worker:rh-ocp-
worker:4.15.2024072409             4.15.2024072409
rh-ocp-worker  redhat-limited  rh-ocp-worker-gen1  redhat-limited:rh-ocp-worker:rh-ocp-
worker-gen1:4.15.2024072409         4.15.2024072409
```

NOTE

Use the latest image that is available for compute and control plane nodes. If required, your VMs are automatically upgraded as part of the installation process.

2. Inspect the image for your offer by running one of the following commands:

   - North America:

   ```
   $ az vm image show --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
   ```

   - EMEA:

   ```
   $ az vm image show --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
   ```

3. Review the terms of the offer by running one of the following commands:

   - North America:

   ```
   $ az vm image terms show --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
   ```

   - EMEA:

   ```
   $ az vm image terms show --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
   ```

4. Accept the terms of the offering by running one of the following commands:

   - North America:

```
$ az vm image terms accept --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
```

- EMEA:

```
$ az vm image terms accept --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
```

5. Record the image details of your offer. If you use the Azure Resource Manager (ARM) template to deploy your compute nodes:

   a. Update **storageProfile.imageReference** by deleting the **id** parameter and adding the **offer**, **publisher**, **sku**, and **version** parameters by using the values from your offer.

   b. Specify a **plan** for the virtual machines (VMs).

   Example 06_workers.json ARM template with an updated storageProfile.imageReference object and a specified plan

```
...
  "plan" : {
    "name": "rh-ocp-worker",
    "product": "rh-ocp-worker",
    "publisher": "redhat"
  },
  "dependsOn" : [
    "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')
[copyIndex()], '-nic'))]"
  ],
  "properties" : {
...
  "storageProfile": {
    "imageReference": {
    "offer": "rh-ocp-worker",
    "publisher": "redhat",
    "sku": "rh-ocp-worker",
    "version": "413.92.2023101700"
    }
    ...
   }
...
  }
```

4.3.6. Obtaining the installation program

Before you install OpenShift Container Platform, download the installation file on the host you are using for installation.

Prerequisites

- You have a computer that runs Linux or macOS, with 500 MB of local disk space.

Procedure

1. Go to the Cluster Type page on the Red Hat Hybrid Cloud Console. If you have a Red Hat account, log in with your credentials. If you do not, create an account.

TIP

You can also download the binaries for a specific OpenShift Container Platform release .

2. Select your infrastructure provider from the **Run it yourself** section of the page.

3. Select your host operating system and architecture from the dropdown menus under **OpenShift Installer** and click **Download Installer**.

4. Place the downloaded file in the directory where you want to store the installation configuration files.

IMPORTANT

- The installation program creates several files on the computer that you use to install your cluster. You must keep the installation program and the files that the installation program creates after you finish installing the cluster. Both of the files are required to delete the cluster.

- Deleting the files created by the installation program does not remove your cluster, even if the cluster failed during installation. To remove your cluster, complete the OpenShift Container Platform uninstallation procedures for your specific cloud provider.

5. Extract the installation program. For example, on a computer that uses a Linux operating system, run the following command:

```
$ tar -xvf openshift-install-linux.tar.gz
```

6. Download your installation pull secret from Red Hat OpenShift Cluster Manager . This pull secret allows you to authenticate with the services that are provided by the included authorities, including Quay.io, which serves the container images for OpenShift Container Platform components.

TIP

Alternatively, you can retrieve the installation program from the Red Hat Customer Portal, where you can specify a version of the installation program to download. However, you must have an active subscription to access this page.

4.3.7. Generating a key pair for cluster node SSH access

During an OpenShift Container Platform installation, you can provide an SSH public key to the installation program. The key is passed to the Red Hat Enterprise Linux CoreOS (RHCOS) nodes through their Ignition config files and is used to authenticate SSH access to the nodes. The key is added to the **~/.ssh/authorized_keys** list for the **core** user on each node, which enables password-less authentication.

After the key is passed to the nodes, you can use the key pair to SSH in to the RHCOS nodes as the user **core**. To access the nodes through SSH, the private key identity must be managed by SSH for your local user.

If you want to SSH in to your cluster nodes to perform installation debugging or disaster recovery, you must provide the SSH public key during the installation process. The **./openshift-install gather** command also requires the SSH public key to be in place on the cluster nodes.

IMPORTANT

Do not skip this procedure in production environments, where disaster recovery and debugging is required.

NOTE

You must use a local key, not one that you configured with platform-specific approaches.

Procedure

1. If you do not have an existing SSH key pair on your local machine to use for authentication onto your cluster nodes, create one. For example, on a computer that uses a Linux operating system, run the following command:

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 Specify the path and file name, such as **~/.ssh/id_ed25519**, of the new SSH key. If you have an existing key pair, ensure your public key is in the your **~/.ssh** directory.

NOTE

If you plan to install an OpenShift Container Platform cluster that uses the RHEL cryptographic libraries that have been submitted to NIST for FIPS 140-2/140-3 Validation on only the **x86_64**, **ppc64le**, and **s390x** architectures, do not create a key that uses the **ed25519** algorithm. Instead, create a key that uses the **rsa** or **ecdsa** algorithm.

2. View the public SSH key:

```
$ cat <path>/<file_name>.pub
```

For example, run the following to view the **~/.ssh/id_ed25519.pub** public key:

```
$ cat ~/.ssh/id_ed25519.pub
```

3. Add the SSH private key identity to the SSH agent for your local user, if it has not already been added. SSH agent management of the key is required for password-less SSH authentication onto your cluster nodes, or if you want to use the **./openshift-install gather** command.

NOTE

On some distributions, default SSH private key identities such as **~/.ssh/id_rsa** and **~/.ssh/id_dsa** are managed automatically.

a. If the **ssh-agent** process is not already running for your local user, start it as a background task:

```
$ eval "$(ssh-agent -s)"
```

Example output

```
Agent pid 31874
```

> **NOTE**
>
> If your cluster is in FIPS mode, only use FIPS-compliant algorithms to generate the SSH key. The key must be either RSA or ECDSA.

4. Add your SSH private key to the **ssh-agent**:

```
$ ssh-add <path>/<file_name> (1)
```

(1) Specify the path and file name for your SSH private key, such as **~/.ssh/id_ed25519**

Example output

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

Next steps

- When you install OpenShift Container Platform, provide the SSH public key to the installation program. If you install a cluster on infrastructure that you provision, you must provide the key to the installation program.

4.3.8. Creating the installation files for Azure

To install OpenShift Container Platform on Microsoft Azure using user-provisioned infrastructure, you must generate the files that the installation program needs to deploy your cluster and modify them so that the cluster creates only the machines that it will use. You generate and customize the **install-config.yaml** file, Kubernetes manifests, and Ignition config files. You also have the option to first set up a separate **var** partition during the preparation phases of installation.

4.3.8.1. Optional: Creating a separate /var partition

It is recommended that disk partitioning for OpenShift Container Platform be left to the installer. However, there are cases where you might want to create separate partitions in a part of the filesystem that you expect to grow.

OpenShift Container Platform supports the addition of a single partition to attach storage to either the **/var** partition or a subdirectory of **/var**. For example:

- **/var/lib/containers**: Holds container-related content that can grow as more images and containers are added to a system.
- **/var/lib/etcd**: Holds data that you might want to keep separate for purposes such as performance optimization of etcd storage.
- **/var**: Holds data that you might want to keep separate for purposes such as auditing.

Storing the contents of a **/var** directory separately makes it easier to grow storage for those areas as needed and reinstall OpenShift Container Platform at a later date and keep that data intact. With this method, you will not have to pull all your containers again, nor will you have to copy massive log files when you update systems.

Because **/var** must be in place before a fresh installation of Red Hat Enterprise Linux CoreOS (RHCOS), the following procedure sets up the separate **/var** partition by creating a machine config manifest that is inserted during the **openshift-install** preparation phases of an OpenShift Container Platform installation.

> IMPORTANT
>
> If you follow the steps to create a separate **/var** partition in this procedure, it is not necessary to create the Kubernetes manifest and Ignition config files again as described later in this section.

Procedure

1. Create a directory to hold the OpenShift Container Platform installation files:

```
$ mkdir $HOME/clusterconfig
```

2. Run **openshift-install** to create a set of files in the **manifest** and **openshift** subdirectories. Answer the system questions as you are prompted:

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

Example output

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. Optional: Confirm that the installation program created manifests in the **clusterconfig/openshift** directory:

```
$ ls $HOME/clusterconfig/openshift/
```

Example output

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. Create a Butane config that configures the additional partition. For example, name the file **$HOME/clusterconfig/98-var-partition.bu**, change the disk device name to the name of the storage device on the **worker** systems, and set the storage size as appropriate. This example places the **/var** directory on a separate partition:

```
variant: openshift
version: 4.18.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

1 The storage device name of the disk that you want to partition.

2 When adding a data partition to the boot disk, a minimum value of 25000 MiB (Mebibytes) is recommended. The root file system is automatically resized to fill all available space up to the specified offset. If no value is specified, or if the specified value is smaller than the recommended minimum, the resulting root file system will be too small, and future reinstalls of RHCOS might overwrite the beginning of the data partition.

3 The size of the data partition in mebibytes.

4 The **prjquota** mount option must be enabled for filesystems used for container storage.

NOTE

When creating a separate **/var** partition, you cannot use different instance types for worker nodes, if the different instance types do not have the same device name.

5. Create a manifest from the Butane config and save it to the **clusterconfig/openshift** directory. For example, run the following command:

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

6. Run **openshift-install** again to create Ignition configs from a set of files in the **manifest** and **openshift** subdirectories:

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

Now you can use the Ignition config files as input to the installation procedures to install Red Hat Enterprise Linux CoreOS (RHCOS) systems.

4.3.8.2. Creating the installation configuration file

You can customize the OpenShift Container Platform cluster you install on Microsoft Azure.

Prerequisites

- You have the OpenShift Container Platform installation program and the pull secret for your cluster.
- You have an Azure subscription ID and tenant ID.
- If you are installing the cluster using a service principal, you have its application ID and password.
- If you are installing the cluster using a system-assigned managed identity, you have enabled it on the virtual machine that you will run the installation program from.
- If you are installing the cluster using a user-assigned managed identity, you have met these prerequisites:
  - You have its client ID.
  - You have assigned it to the virtual machine that you will run the installation program from.

Procedure

1. Optional: If you have run the installation program on this computer before, and want to use an alternative service principal or managed identity, go to the ~/**.azure**/ directory and delete the **osServicePrincipal.json** configuration file.
   Deleting this file prevents the installation program from automatically reusing subscription and authentication values from a previous installation.

2. Create the **install-config.yaml** file.

   a. Change to the directory that contains the installation program and run the following command:

   ```
   $ ./openshift-install create install-config --dir <installation_directory> 1
   ```

   1 For **<installation_directory>**, specify the directory name to store the files that the installation program creates.

   When specifying the directory:

   - Verify that the directory has the **execute** permission. This permission is required to run Terraform binaries under the installation directory.
   - Use an empty directory. Some installation assets, such as bootstrap X.509 certificates, have short expiration intervals, therefore you must not reuse an installation directory. If you want to reuse individual files from another cluster installation, you can copy them into your directory. However, the file names for the installation assets might change between releases. Use caution when copying installation files from an earlier OpenShift Container Platform version.

b. At the prompts, provide the configuration details for your cloud:

  i. Optional: Select an SSH key to use to access your cluster machines.

  NOTE

  For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.

  ii. Select **azure** as the platform to target.
  If the installation program cannot locate the **osServicePrincipal.json** configuration file from a previous installation, you are prompted for Azure subscription and authentication values.

  iii. Enter the following Azure parameter values for your subscription:

  - **azure subscription id** Enter the subscription ID to use for the cluster.

  - **azure tenant id** Enter the tenant ID.

  iv. Depending on the Azure identity you are using to deploy the cluster, do one of the following when prompted for the **azure service principal client id**

  - If you are using a service principal, enter its application ID.

  - If you are using a system-assigned managed identity, leave this value blank.

  - If you are using a user-assigned managed identity, specify its client ID.

  v. Depending on the Azure identity you are using to deploy the cluster, do one of the following when prompted for the **azure service principal client secret**

  - If you are using a service principal, enter its password.

  - If you are using a system-assigned managed identity, leave this value blank.

  - If you are using a user-assigned managed identity, leave this value blank.

  vi. Select the region to deploy the cluster to.

  vii. Select the base domain to deploy the cluster to. The base domain corresponds to the Azure DNS Zone that you created for your cluster.

  viii. Enter a descriptive name for your cluster.

  IMPORTANT

  All Azure resources that are available through public endpoints are subject to resource name restrictions, and you cannot create resources that use certain terms. For a list of terms that Azure restricts, see Resolve reserved resource name errors in the Azure documentation.

3. Modify the **install-config.yaml** file. You can find more information about the available parameters in the "Installation configuration parameters" section.

NOTE

If you are installing a three-node cluster, be sure to set the **compute.replicas** parameter to **0**. This ensures that the cluster's control planes are schedulable. For more information, see "Installing a three-node cluster on Azure".

4. Back up the **install-config.yaml** file so that you can use it to install multiple clusters.

IMPORTANT

The **install-config.yaml** file is consumed during the installation process. If you want to reuse the file, you must back it up now.

If previously not detected, the installation program creates an **osServicePrincipal.json** configuration file and stores this file in the ~**/.azure**/ directory on your computer. This ensures that the installation program can load the profile when it is creating an OpenShift Container Platform cluster on the target platform.

4.3.8.3. Configuring the cluster-wide proxy during installation

Production environments can deny direct access to the internet and instead have an HTTP or HTTPS proxy available. You can configure a new OpenShift Container Platform cluster to use a proxy by configuring the proxy settings in the **install-config.yaml** file.

Prerequisites

- You have an existing **install-config.yaml** file.
- You reviewed the sites that your cluster requires access to and determined whether any of them need to bypass the proxy. By default, all cluster egress traffic is proxied, including calls to hosting cloud provider APIs. You added sites to the **Proxy** object's **spec.noProxy** field to bypass the proxy if necessary.

NOTE

The **Proxy** object **status.noProxy** field is populated with the values of the **networking.machineNetwork[].cidr**, **networking.clusterNetwork[].cidr**, and **networking.serviceNetwork[]** fields from your installation configuration.

For installations on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, and Red Hat OpenStack Platform (RHOSP), the **Proxy** object **status.noProxy** field is also populated with the instance metadata endpoint (**169.254.169.254**).

Procedure

1. Edit your **install-config.yaml** file and add the proxy settings. For example:

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
```

```
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. A proxy URL to use for creating HTTP connections outside the cluster. The URL scheme must be **http**.

2. A proxy URL to use for creating HTTPS connections outside the cluster.

3. A comma-separated list of destination domain names, IP addresses, or other network CIDRs to exclude from proxying. Preface a domain with **.** to match subdomains only. For example, **.y.com** matches **x.y.com**, but not **y.com**. Use * to bypass the proxy for all destinations.

4. If provided, the installation program generates a config map that is named **user-ca-bundle** in the **openshift-config** namespace that contains one or more additional CA certificates that are required for proxying HTTPS connections. The Cluster Network Operator then creates a **trusted-ca-bundle** config map that merges these contents with the Red Hat Enterprise Linux CoreOS (RHCOS) trust bundle, and this config map is referenced in the **trustedCA** field of the **Proxy** object. The **additionalTrustBundle** field is required unless the proxy's identity certificate is signed by an authority from the RHCOS trust bundle.

5. Optional: The policy to determine the configuration of the **Proxy** object to reference the **user-ca-bundle** config map in the **trustedCA** field. The allowed values are **Proxyonly** and **Always**. Use **Proxyonly** to reference the **user-ca-bundle** config map only when **http/https** proxy is configured. Use **Always** to always reference the **user-ca-bundle** config map. The default value is **Proxyonly**.

NOTE

The installation program does not support the proxy **readinessEndpoints** field.

NOTE

If the installer times out, restart and then complete the deployment by using the **wait-for** command of the installer. For example:

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. Save the file and reference it when installing OpenShift Container Platform.

The installation program creates a cluster-wide proxy that is named **cluster** that uses the proxy settings in the provided **install-config.yaml** file. If no proxy settings are provided, a **cluster Proxy** object is still created, but it will have a nil **spec**.

NOTE

Only the **Proxy** object named **cluster** is supported, and no additional proxies can be created.

4.3.8.4. Exporting common variables for ARM templates

You must export a common set of variables that are used with the provided Azure Resource Manager (ARM) templates used to assist in completing a user-provided infrastructure install on Microsoft Azure.

NOTE

Specific ARM templates can also require additional exported variables, which are detailed in their related procedures.

Prerequisites

- Obtain the OpenShift Container Platform installation program and the pull secret for your cluster.

Procedure

1. Export common variables found in the **install-config.yaml** to be used by the provided ARM templates:

```
$ export CLUSTER_NAME=<cluster_name> 1
$ export AZURE_REGION=<azure_region> 2
$ export SSH_KEY=<ssh_key> 3
$ export BASE_DOMAIN=<base_domain> 4
$ export BASE_DOMAIN_RESOURCE_GROUP=<base_domain_resource_group> 5
```

1 The value of the **.metadata.name** attribute from the **install-config.yaml** file.

2 The region to deploy the cluster into, for example **centralus**. This is the value of the **.platform.azure.region** attribute from the **install-config.yaml** file.

3 The SSH RSA public key file as a string. You must enclose the SSH key in quotes since it contains spaces. This is the value of the **.sshKey** attribute from the **install-config.yaml** file.

4 The base domain to deploy the cluster to. The base domain corresponds to the public DNS zone that you created for your cluster. This is the value of the **.baseDomain** attribute from the **install-config.yaml** file.

5 The resource group where the public DNS zone exists. This is the value of the **.platform.azure.baseDomainResourceGroupName** attribute from the **install-config.yaml** file.

For example:

```
$ export CLUSTER_NAME=test-cluster
$ export AZURE_REGION=centralus
$ export SSH_KEY="ssh-rsa xxx/xxx/xxx= user@email.com"
$ export BASE_DOMAIN=example.com
$ export BASE_DOMAIN_RESOURCE_GROUP=ocp-cluster
```

2. Export the kubeadmin credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1. For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

4.3.8.5. Creating the Kubernetes manifest and Ignition config files

Because you must modify some cluster definition files and manually start the cluster machines, you must generate the Kubernetes manifest and Ignition config files that the cluster needs to configure the machines.

The installation configuration file transforms into the Kubernetes manifests. The manifests wrap into the Ignition configuration files, which are later used to configure the cluster machines.

IMPORTANT

- The Ignition config files that the OpenShift Container Platform installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.

- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

Prerequisites

- You obtained the OpenShift Container Platform installation program.
- You created the **install-config.yaml** installation configuration file.

Procedure

1. Change to the directory that contains the OpenShift Container Platform installation program and generate the Kubernetes manifests for the cluster:

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1. For **<installation_directory>**, specify the installation directory that contains the **install-config.yaml** file you created.

2. Remove the Kubernetes manifest files that define the control plane machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

By removing these files, you prevent the cluster from automatically generating control plane machines.

3. Remove the Kubernetes manifest files that define the control plane machine set:

```
$ rm -f <installation_directory>/openshift/99_openshift-machine-api_master-control-plane-machine-set.yaml
```

4. Remove the Kubernetes manifest files that define the worker machines:

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

IMPORTANT

If you disabled the **MachineAPI** capability when installing a cluster on user-provisioned infrastructure, you must remove the Kubernetes manifest files that define the worker machines. Otherwise, your cluster fails to install.

Because you create and manage the worker machines yourself, you do not need to initialize these machines.

WARNING

If you are installing a three-node cluster, skip the following step to allow the control plane nodes to be schedulable.

IMPORTANT

When you configure control plane nodes from the default unschedulable to schedulable, additional subscriptions are required. This is because control plane nodes then become compute nodes.

5. Check that the **mastersSchedulable** parameter in the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes manifest file is set to **false**. This setting prevents pods from being scheduled on the control plane machines:

   a. Open the **<installation_directory>/manifests/cluster-scheduler-02-config.yml** file.

   b. Locate the **mastersSchedulable** parameter and ensure that it is set to **false**.

   c. Save and exit the file.

6. Optional: If you do not want the Ingress Operator to create DNS records on your behalf, remove the **privateZone** and **publicZone** sections from the **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS configuration file:

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
    id: mycluster-100419-private-zone
  publicZone: 2
    id: example.openshift.com
status: {}
```

1 2 Remove this section completely.

If you do so, you must add ingress DNS records manually in a later step.

7. When configuring Azure on user-provisioned infrastructure, you must export some common variables defined in the manifest files to use later in the Azure Resource Manager (ARM) templates:

   a. Export the infrastructure ID by using the following command:

      ```
      $ export INFRA_ID=<infra_id> 1
      ```

      1 The OpenShift Container Platform cluster has been assigned an identifier (**INFRA_ID**) in the form of **<cluster_name>-<random_string>**. This will be used as the base name for most resources created using the provided ARM templates. This is the value of the **.status.infrastructureName** attribute from the **manifests/cluster-infrastructure-02-config.yml** file.

   b. Export the resource group by using the following command:

      ```
      $ export RESOURCE_GROUP=<resource_group> 1
      ```

      1 All resources created in this Azure deployment exists as part of a resource group. The resource group name is also based on the **INFRA_ID**, in the form of **<cluster_name>-<random_string>-rg**. This is the value of the **.status.platformStatus.azure.resourceGroupName** attribute from the **manifests/cluster-infrastructure-02-config.yml** file.

8. To create the Ignition configuration files, run the following command from the directory that contains the installation program:

   ```
   $ ./openshift-install create ignition-configs --dir <installation_directory> 1
   ```

   1 For **<installation_directory>**, specify the same installation directory.

   Ignition config files are created for the bootstrap, control plane, and compute nodes in the installation directory. The **kubeadmin-password** and **kubeconfig** files are created in the **./<installation_directory>/auth** directory:

   ```
   .
   ├── auth
   │   ├── kubeadmin-password
   │   └── kubeconfig
   ├── bootstrap.ign
   ```

```
├── master.ign
├── metadata.json
└── worker.ign
```

4.3.9. Creating the Azure resource group

You must create a Microsoft Azure resource group and an identity for that resource group. These are both used during the installation of your OpenShift Container Platform cluster on Azure.

Procedure

1. Create the resource group in a supported Azure region:

   ```
   $ az group create --name ${RESOURCE_GROUP} --location ${AZURE_REGION}
   ```

2. Create an Azure identity for the resource group:

   ```
   $ az identity create -g ${RESOURCE_GROUP} -n ${INFRA_ID}-identity
   ```

   This is used to grant the required access to Operators in your cluster. For example, this allows the Ingress Operator to create a public IP and its load balancer. You must assign the Azure identity to a role.

3. Grant the Contributor role to the Azure identity:

   a. Export the following variables required by the Azure role assignment:

   ```
   $ export PRINCIPAL_ID=`az identity show -g ${RESOURCE_GROUP} -n ${INFRA_ID}-identity --query principalId --out tsv`
   ```

   ```
   $ export RESOURCE_GROUP_ID=`az group show -g ${RESOURCE_GROUP} --query id --out tsv`
   ```

   b. Assign the Contributor role to the identity:

   ```
   $ az role assignment create --assignee "${PRINCIPAL_ID}" --role 'Contributor' --scope "${RESOURCE_GROUP_ID}"
   ```

   NOTE

   If you want to assign a custom role with all the required permissions to the identity, run the following command:

   ```
   $ az role assignment create --assignee "${PRINCIPAL_ID}" --role <custom_role> \ ①
   --scope "${RESOURCE_GROUP_ID}"
   ```

   ① Specifies the custom role name.

4.3.10. Uploading the RHCOS cluster image and bootstrap Ignition config file

The Azure client does not support deployments based on files existing locally. You must copy and store the RHCOS virtual hard disk (VHD) cluster image and bootstrap Ignition config file in a storage container so they are accessible during deployment.

Prerequisites

- Generate the Ignition config files for your cluster.

Procedure

1. Create an Azure storage account to store the VHD cluster image:

```
$ az storage account create -g ${RESOURCE_GROUP} --location ${AZURE_REGION} --
name ${CLUSTER_NAME}sa --kind Storage --sku Standard_LRS
```

WARNING

The Azure storage account name must be between 3 and 24 characters in length and use numbers and lower-case letters only. If your **CLUSTER_NAME** variable does not follow these restrictions, you must manually define the Azure storage account name. For more information on Azure storage account name restrictions, see Resolve errors for storage account names in the Azure documentation.

2. Export the storage account key as an environment variable:

```
$ export ACCOUNT_KEY=`az storage account keys list -g ${RESOURCE_GROUP} --
account-name ${CLUSTER_NAME}sa --query "[0].value" -o tsv`
```

3. Export the URL of the RHCOS VHD to an environment variable:

```
$ export VHD_URL=`openshift-install coreos print-stream-json | jq -r '.architectures.
<architecture>."rhel-coreos-extensions"."azure-disk".url'`
```

where:

<architecture>

Specifies the architecture, valid values include **x86_64** or **aarch64**.

IMPORTANT

The RHCOS images might not change with every release of OpenShift Container Platform. You must specify an image with the highest version that is less than or equal to the OpenShift Container Platform version that you install. Use the image version that matches your OpenShift Container Platform version if it is available.

4. Create the storage container for the VHD:

```
$ az storage container create --name vhd --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY}
```

5. Copy the local VHD to a blob:

```
$ az storage blob copy start --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} --destination-blob "rhcos.vhd" --destination-container vhd --source-uri "${VHD_URL}"
```

6. Create a blob storage container and upload the generated **bootstrap.ign** file:

```
$ az storage container create --name files --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY}
```

```
$ az storage blob upload --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} -c "files" -f "<installation_directory>/bootstrap.ign" -n "bootstrap.ign"
```

4.3.11. Example for creating DNS zones

DNS records are required for clusters that use user-provisioned infrastructure. You should choose the DNS strategy that fits your scenario.

For this example, Azure's DNS solution is used, so you will create a new public DNS zone for external (internet) visibility and a private DNS zone for internal cluster resolution.

> NOTE
>
> The public DNS zone is not required to exist in the same resource group as the cluster deployment and might already exist in your organization for the desired base domain. If that is the case, you can skip creating the public DNS zone; be sure the installation config you generated earlier reflects that scenario.

Procedure

1. Create the new public DNS zone in the resource group exported in the **BASE_DOMAIN_RESOURCE_GROUP** environment variable:

```
$ az network dns zone create -g ${BASE_DOMAIN_RESOURCE_GROUP} -n ${CLUSTER_NAME}.${BASE_DOMAIN}
```

You can skip this step if you are using a public DNS zone that already exists.

2. Create the private DNS zone in the same resource group as the rest of this deployment:

```
$ az network private-dns zone create -g ${RESOURCE_GROUP} -n ${CLUSTER_NAME}.${BASE_DOMAIN}
```

You can learn more about configuring a public DNS zone in Azure by visiting that section.

4.3.12. Creating a VNet in Azure

You must create a virtual network (VNet) in Microsoft Azure for your OpenShift Container Platform cluster to use. You can customize the VNet to meet your requirements. One way to create the VNet is to modify the provided Azure Resource Manager (ARM) template.

NOTE

If you do not use the provided ARM template to create your Azure infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Procedure

1. Copy the template from the **ARM template for the VNet**section of this topic and save it as **01_vnet.json** in your cluster's installation directory. This template describes the VNet that your cluster requires.

2. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/01_vnet.json" \
  --parameters baseName="${INFRA_ID}" 1
```

1 The base name to be used in resource names; this is usually the cluster's infrastructure ID.

3. Link the VNet template to the private DNS zone:

```
$ az network private-dns link vnet create -g ${RESOURCE_GROUP} -z
${CLUSTER_NAME}.${BASE_DOMAIN} -n ${INFRA_ID}-network-link -v "${INFRA_ID}-vnet"
-e false
```

4.3.12.1. ARM template for the VNet

You can use the following Azure Resource Manager (ARM) template to deploy the VNet that you need for your OpenShift Container Platform cluster:

Example 4.49. 01_vnet.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
  "virtualNetworkName" : "[concat(parameters('baseName'), '-vnet')]",
  "addressPrefix" : "10.0.0.0/16",
  "masterSubnetName" : "[concat(parameters('baseName'), '-master-subnet')]",
  "masterSubnetPrefix" : "10.0.0.0/24",
  "nodeSubnetName" : "[concat(parameters('baseName'), '-worker-subnet')]",
  "nodeSubnetPrefix" : "10.0.1.0/24",
  "clusterNsgName" : "[concat(parameters('baseName'), '-nsg')]"
 },
 "resources" : [
  {
   "apiVersion" : "2018-12-01",
   "type" : "Microsoft.Network/virtualNetworks",
   "name" : "[variables('virtualNetworkName')]",
   "location" : "[variables('location')]",
   "dependsOn" : [
    "[concat('Microsoft.Network/networkSecurityGroups/', variables('clusterNsgName'))]"
   ],
   "properties" : {
    "addressSpace" : {
     "addressPrefixes" : [
      "[variables('addressPrefix')]"
     ]
    },
    "subnets" : [
     {
      "name" : "[variables('masterSubnetName')]",
      "properties" : {
       "addressPrefix" : "[variables('masterSubnetPrefix')]",
       "serviceEndpoints": [],
       "networkSecurityGroup" : {
        "id" : "[resourceId('Microsoft.Network/networkSecurityGroups',
variables('clusterNsgName'))]"
       }
      }
     },
     {
      "name" : "[variables('nodeSubnetName')]",
      "properties" : {
       "addressPrefix" : "[variables('nodeSubnetPrefix')]",
       "serviceEndpoints": [],
       "networkSecurityGroup" : {
        "id" : "[resourceId('Microsoft.Network/networkSecurityGroups',
variables('clusterNsgName'))]"
       }
      }
     }
    ]
   }
  },
  {
   "type" : "Microsoft.Network/networkSecurityGroups",
   "name" : "[variables('clusterNsgName')]",
   "apiVersion" : "2018-10-01",
   "location" : "[variables('location')]",
   "properties" : {
    "securityRules" : [
          {
            "name" : "apiserver_in",
            "properties" : {
              "protocol" : "Tcp",
              "sourcePortRange" : "*",
              "destinationPortRange" : "6443",
              "sourceAddressPrefix" : "*",
              "destinationAddressPrefix" : "*",
              "access" : "Allow",
              "priority" : 101,
              "direction" : "Inbound"
            }
          }
        ]
      }
    }
  ]
}
```

4.3.13. Deploying the RHCOS cluster image for the Azure infrastructure

You must use a valid Red Hat Enterprise Linux CoreOS (RHCOS) image for Microsoft Azure for your OpenShift Container Platform nodes.

Prerequisites

- Store the RHCOS virtual hard disk (VHD) cluster image in an Azure storage container.
- Store the bootstrap Ignition config file in an Azure storage container.

Procedure

1. Copy the template from the **ARM template for image storage**section of this topic and save it as **02_storage.json** in your cluster's installation directory. This template describes the image storage that your cluster requires.

2. Export the RHCOS VHD blob URL as a variable:

```
$ export VHD_BLOB_URL=`az storage blob url --account-name ${CLUSTER_NAME}sa --
account-key ${ACCOUNT_KEY} -c vhd -n "rhcos.vhd" -o tsv`
```

3. Deploy the cluster image:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/02_storage.json" \
  --parameters vhdBlobURL="${VHD_BLOB_URL}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameters storageAccount="${CLUSTER_NAME}sa" \ 3
  --parameters architecture="<architecture>" 4
```

1 The blob URL of the RHCOS VHD to be used to create master and worker machines.

2 The base name to be used in resource names; this is usually the cluster's infrastructure ID.

3 The name of your Azure storage account.

4 Specify the system architecture. Valid values are **x64** (default) or **Arm64**.

4.3.13.1. ARM template for image storage

You can use the following Azure Resource Manager (ARM) template to deploy the stored Red Hat Enterprise Linux CoreOS (RHCOS) image that you need for your OpenShift Container Platform cluster:

Example 4.50. **02_storage.json** ARM template

```
{
  "$schema": "https://schema.management.azure.com/schemas/2019-04-
01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "architecture": {
      "type": "string",
      "metadata": {
        "description": "The architecture of the Virtual Machines"
      },
      "defaultValue": "x64",
      "allowedValues": [
        "Arm64",
        "x64"
      ]
    },
    "baseName": {
      "type": "string",
      "minLength": 1,
      "metadata": {
        "description": "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "storageAccount": {
      "type": "string",
      "metadata": {
        "description": "The Storage Account name"
      }
    },
    "vhdBlobURL": {
      "type": "string",
      "metadata": {
        "description": "URL pointing to the blob where the VHD to be used to create master and
worker machines is located"
      }
    }
  },
  "variables": {
    "location": "[resourceGroup().location]",
    "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
    "imageName": "[parameters('baseName')]",
    "imageNameGen2": "[concat(parameters('baseName'), '-gen2')]",
    "imageRelease": "1.0.0"
  },
  "resources": [
    {
      "apiVersion": "2021-10-01",
      "type": "Microsoft.Compute/galleries",
      "name": "[variables('galleryName')]",
      "location": "[variables('location')]",
      "resources": [
        {
          "apiVersion": "2021-10-01",
          "type": "images",
          "name": "[variables('imageName')]",
          "location": "[variables('location')]",
          "dependsOn": [
            "[variables('galleryName')]"
          ],
          "properties": {
            "architecture": "[parameters('architecture')]",
            "hyperVGeneration": "V1",
            "identifier": {
              "offer": "rhcos",
              "publisher": "RedHat",
              "sku": "basic"
            },
            "osState": "Generalized",
            "osType": "Linux"
          },
          "resources": [
            {
              "apiVersion": "2021-10-01",
              "type": "versions",
              "name": "[variables('imageRelease')]",
              "location": "[variables('location')]",
              "dependsOn": [
                "[variables('imageName')]"
              ],
              "properties": {
                "publishingProfile": {
                  "storageAccountType": "Standard_LRS",
                  "targetRegions": [
                    {
                      "name": "[variables('location')]",
                      "regionalReplicaCount": "1"
                    }
                  ]
                },
                "storageProfile": {
                  "osDiskImage": {
                    "source": {
                      "id": "[resourceId('Microsoft.Storage/storageAccounts',
parameters('storageAccount'))]",
                      "uri": "[parameters('vhdBlobURL')]"
                    }
                  }
                }
              }
            }
          ]
        },
        {
          "apiVersion": "2021-10-01",
          "type": "images",
          "name": "[variables('imageNameGen2')]",
          "location": "[variables('location')]",
          "dependsOn": [
            "[variables('galleryName')]"
          ],
          "properties": {
            "architecture": "[parameters('architecture')]",
            "hyperVGeneration": "V2",
            "identifier": {
              "offer": "rhcos-gen2",
              "publisher": "RedHat-gen2",
              "sku": "gen2"
            },
            "osState": "Generalized",
            "osType": "Linux"
          },
          "resources": [
            {
              "apiVersion": "2021-10-01",
              "type": "versions",
              "name": "[variables('imageRelease')]",
              "location": "[variables('location')]",
              "dependsOn": [
                "[variables('imageNameGen2')]"
              ],
              "properties": {
                "publishingProfile": {
                  "storageAccountType": "Standard_LRS",
                  "targetRegions": [
                    {
                      "name": "[variables('location')]",
                      "regionalReplicaCount": "1"
                    }
                  ]
                },
                "storageProfile": {
                  "osDiskImage": {
                    "source": {
                      "id": "[resourceId('Microsoft.Storage/storageAccounts',
parameters('storageAccount'))]",
                      "uri": "[parameters('vhdBlobURL')]"
                    }
                  }
                }
              }
            }
          ]
        }
      ]
```

```
        }
      ]
    }
```

4.3.14. Networking requirements for user-provisioned infrastructure

All the Red Hat Enterprise Linux CoreOS (RHCOS) machines require networking to be configured in **initramfs** during boot to fetch their Ignition config files.

4.3.14.1. Network connectivity requirements

You must configure the network connectivity between machines to allow OpenShift Container Platform cluster components to communicate. Each machine must be able to resolve the hostnames of all other machines in the cluster.

This section provides details about the ports that are required.

IMPORTANT

In connected OpenShift Container Platform environments, all nodes are required to have internet access to pull images for platform containers and provide telemetry data to Red Hat.

Table 4.8. Ports used for all-machine to all-machine communications

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	**1936**	Metrics
	9000-**9999**	Host level services, including the node exporter on ports **9100**-**9101** and the Cluster Version Operator on port**9099**.
	10250-**10259**	The default ports that Kubernetes reserves
UDP	**4789**	VXLAN
	6081	Geneve
	9000-**9999**	Host level services, including the node exporter on ports **9100**-**9101**.
	500	IPsec IKE packets
	4500	IPsec NAT-T packets

Protocol	Port	Description
	123	Network Time Protocol (NTP) on UDP port **123** If an external NTP time server is configured, you must open UDP port **123**.
TCP/UDP	**30000**-**32767**	Kubernetes node port
ESP	N/A	IPsec Encapsulating Security Payload (ESP)

Table 4.9. Ports used for all-machine to control plane communications

Protocol	Port	Description
TCP	**6443**	Kubernetes API

Table 4.10. Ports used for control plane machine to control plane machine communications

Protocol	Port	Description
TCP	**2379**-**2380**	etcd server and peer ports

4.3.15. Creating networking and load balancing components in Azure

You must configure networking and load balancing in Microsoft Azure for your OpenShift Container Platform cluster to use. One way to create these components is to modify the provided Azure Resource Manager (ARM) template.

NOTE

If you do not use the provided ARM template to create your Azure infrastructure, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Create and configure a VNet and associated subnets in Azure.

Procedure

1. Copy the template from the **ARM template for the network and load balancers**section of this topic and save it as **03_infra.json** in your cluster's installation directory. This template describes the networking and load balancing objects that your cluster requires.

2. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/03_infra.json" \
--parameters privateDNSZoneName="${CLUSTER_NAME}.${BASE_DOMAIN}" \ 1
--parameters baseName="${INFRA_ID}" 2
```

1. The name of the private DNS zone.

2. The base name to be used in resource names; this is usually the cluster’s infrastructure ID.

3. Create an **api** DNS record in the public zone for the API public load balancer. The **${BASE_DOMAIN_RESOURCE_GROUP}** variable must point to the resource group where the public DNS zone exists.

   a. Export the following variable:

   ```
   $ export PUBLIC_IP=`az network public-ip list -g ${RESOURCE_GROUP} --query "[?
   name=='${INFRA_ID}-master-pip'] | [0].ipAddress" -o tsv`
   ```

   b. Create the **api** DNS record in a new public zone:

   ```
   $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
   z ${CLUSTER_NAME}.${BASE_DOMAIN} -n api -a ${PUBLIC_IP} --ttl 60
   ```

   If you are adding the cluster to an existing public zone, you can create the **api** DNS record in it instead:

   ```
   $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
   z ${BASE_DOMAIN} -n api.${CLUSTER_NAME} -a ${PUBLIC_IP} --ttl 60
   ```

4.3.15.1. ARM template for the network and load balancers

You can use the following Azure Resource Manager (ARM) template to deploy the networking objects and load balancers that you need for your OpenShift Container Platform cluster:

Example 4.51. 03_infra.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vnetBaseName": {
      "type": "string",
      "defaultValue": "",
      "metadata" : {
        "description" : "The specific customer vnet's base name (optional)"
      }
    },
```

```
    "privateDNSZoneName" : {
      "type" : "string",
      "metadata" : {
        "description" : "Name of the private DNS zone"
      }
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "masterSubnetName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-master-subnet')]",
    "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
    "masterPublicIpAddressName" : "[concat(parameters('baseName'), '-master-pip')]",
    "masterPublicIpAddressID" : "[resourceId('Microsoft.Network/publicIPAddresses',
variables('masterPublicIpAddressName'))]",
    "masterLoadBalancerName" : "[parameters('baseName')]",
    "masterLoadBalancerID" : "[resourceId('Microsoft.Network/loadBalancers',
variables('masterLoadBalancerName'))]",
    "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal-lb')]",
    "internalLoadBalancerID" : "[resourceId('Microsoft.Network/loadBalancers',
variables('internalLoadBalancerName'))]",
    "skuName": "Standard"
  },
  "resources" : [
    {
      "apiVersion" : "2018-12-01",
      "type" : "Microsoft.Network/publicIPAddresses",
      "name" : "[variables('masterPublicIpAddressName')]",
      "location" : "[variables('location')]",
      "sku": {
        "name": "[variables('skuName')]"
      },
      "properties" : {
        "publicIPAllocationMethod" : "Static",
        "dnsSettings" : {
          "domainNameLabel" : "[variables('masterPublicIpAddressName')]"
        }
      }
    },
    {
      "apiVersion" : "2018-12-01",
      "type" : "Microsoft.Network/loadBalancers",
      "name" : "[variables('masterLoadBalancerName')]",
      "location" : "[variables('location')]",
      "sku": {
        "name": "[variables('skuName')]"
      },
      "dependsOn" : [
        "[concat('Microsoft.Network/publicIPAddresses/', variables('masterPublicIpAddressName'))]"
      ],
      "properties" : {
```

```
    "frontendIPConfigurations" : [
     {
      "name" : "public-lb-ip-v4",
      "properties" : {
       "publicIPAddress" : {
        "id" : "[variables('masterPublicIpAddressID')]"
       }
      }
     }
    ],
    "backendAddressPools" : [
     {
      "name" : "[variables('masterLoadBalancerName')]"
     }
    ],
    "loadBalancingRules" : [
     {
      "name" : "api-internal",
      "properties" : {
       "frontendIPConfiguration" : {
        "id" :"[concat(variables('masterLoadBalancerID'), '/frontendIPConfigurations/public-lb-ip-
v4')]"
       },
       "backendAddressPool" : {
        "id" : "[concat(variables('masterLoadBalancerID'), '/backendAddressPools/',
variables('masterLoadBalancerName'))]"
       },
       "protocol" : "Tcp",
       "loadDistribution" : "Default",
       "idleTimeoutInMinutes" : 30,
       "frontendPort" : 6443,
       "backendPort" : 6443,
       "probe" : {
        "id" : "[concat(variables('masterLoadBalancerID'), '/probes/api-internal-probe')]"
       }
      }
     }
    ],
    "probes" : [
     {
      "name" : "api-internal-probe",
      "properties" : {
       "protocol" : "Https",
       "port" : 6443,
       "requestPath": "/readyz",
       "intervalInSeconds" : 10,
       "numberOfProbes" : 3
      }
     }
    ]
   }
  },
  {
   "apiVersion" : "2018-12-01",
   "type" : "Microsoft.Network/loadBalancers",
   "name" : "[variables('internalLoadBalancerName')]",
   "location" : "[variables('location')]",
   "sku": {
    "name": "[variables('skuName')]"
   },
   "properties" : {
    "frontendIPConfigurations" : [
     {
      "name" : "internal-lb-ip",
      "properties" : {
       "privateIPAllocationMethod" : "Dynamic",
       "subnet" : {
        "id" : "[variables('masterSubnetRef')]"
       },
       "privateIPAddressVersion" : "IPv4"
      }
     }
    ],
    "backendAddressPools" : [
     {
      "name" : "internal-lb-backend"
     }
    ],
    "loadBalancingRules" : [
     {
      "name" : "api-internal",
      "properties" : {
       "frontendIPConfiguration" : {
        "id" : "[concat(variables('internalLoadBalancerID'), '/frontendIPConfigurations/internal-lb-
ip')]"
       },
       "frontendPort" : 6443,
       "backendPort" : 6443,
       "enableFloatingIP" : false,
       "idleTimeoutInMinutes" : 30,
       "protocol" : "Tcp",
       "enableTcpReset" : false,
       "loadDistribution" : "Default",
       "backendAddressPool" : {
        "id" : "[concat(variables('internalLoadBalancerID'), '/backendAddressPools/internal-lb-
backend')]"
       },
       "probe" : {
        "id" : "[concat(variables('internalLoadBalancerID'), '/probes/api-internal-probe')]"
       }
      }
     },
     {
      "name" : "sint",
      "properties" : {
       "frontendIPConfiguration" : {
        "id" : "[concat(variables('internalLoadBalancerID'), '/frontendIPConfigurations/internal-lb-
ip')]"
       },
       "frontendPort" : 22623,
       "backendPort" : 22623,
       "enableFloatingIP" : false,
              "idleTimeoutInMinutes" : 30,
              "protocol" : "Tcp",
              "enableTcpReset" : false,
              "loadDistribution" : "Default",
              "backendAddressPool" : {
                "id" : "[concat(variables('internalLoadBalancerID'), '/backendAddressPools/internal-lb-backend')]"
              },
              "probe" : {
                "id" : "[concat(variables('internalLoadBalancerID'), '/probes/sint-probe')]"
              }
            }
          }
        ],
        "probes" : [
          {
            "name" : "api-internal-probe",
            "properties" : {
              "protocol" : "Https",
              "port" : 6443,
              "requestPath": "/readyz",
              "intervalInSeconds" : 10,
              "numberOfProbes" : 3
            }
          },
          {
            "name" : "sint-probe",
            "properties" : {
              "protocol" : "Https",
              "port" : 22623,
              "requestPath": "/healthz",
              "intervalInSeconds" : 10,
              "numberOfProbes" : 3
            }
          }
        ]
      }
    },
    {
      "apiVersion": "2018-09-01",
      "type": "Microsoft.Network/privateDnsZones/A",
      "name": "[concat(parameters('privateDNSZoneName'), '/api')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[concat('Microsoft.Network/loadBalancers/', variables('internalLoadBalancerName'))]"
      ],
      "properties": {
        "ttl": 60,
        "aRecords": [
          {
            "ipv4Address": "[reference(variables('internalLoadBalancerName')).frontendIPConfigurations[0].properties.privateIPAddress]"
          }
        ]
      }
    },
    {
      "apiVersion": "2018-09-01",
      "type": "Microsoft.Network/privateDnsZones/A",
      "name": "[concat(parameters('privateDNSZoneName'), '/api-int')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[concat('Microsoft.Network/loadBalancers/', variables('internalLoadBalancerName'))]"
      ],
      "properties": {
        "ttl": 60,
        "aRecords": [
          {
            "ipv4Address": "
[reference(variables('internalLoadBalancerName')).frontendIPConfigurations[0].properties.privateIP
Address]"
          }
        ]
      }
    }
  ]
}
```

4.3.16. Creating the bootstrap machine in Azure

You must create the bootstrap machine in Microsoft Azure to use during OpenShift Container Platform cluster initialization. One way to create this machine is to modify the provided Azure Resource Manager (ARM) template.

> **NOTE**
>
> If you do not use the provided ARM template to create your bootstrap machine, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, you might have to contact Red Hat support with your installation logs.

Prerequisites

- Create and configure networking and load balancers in Azure.
- Create the Azure identity and grant the appropriate roles.

Procedure

1. Copy the template from the **ARM template for the bootstrap machine**section of this topic and save it as **04_bootstrap.json** in your cluster's installation directory. This template describes the bootstrap machine that your cluster requires.

2. Export the bootstrap URL variable:

   ```
   $ bootstrap_url_expiry=`date -u -d "10 hours" '+%Y-%m-%dT%H:%MZ'`
   ```

```
$ export BOOTSTRAP_URL=`az storage blob generate-sas -c 'files' -n 'bootstrap.ign' --https-only --full-uri --permissions r --expiry $bootstrap_url_expiry --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} -o tsv`
```

3. Export the bootstrap ignition variable:

```
$ export BOOTSTRAP_IGNITION=`jq -rcnM --arg v "3.2.0" --arg url ${BOOTSTRAP_URL} '{ignition:{version:$v,config:{replace:{source:$url}}}}' | base64 | tr -d '\n'`
```

4. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/04_bootstrap.json" \
  --parameters bootstrapIgnition="${BOOTSTRAP_IGNITION}" \ ①
  --parameters baseName="${INFRA_ID}" \ ②
  --parameter bootstrapVMSize="Standard_D4s_v3" ③
```

① The bootstrap Ignition content for the bootstrap cluster.

② The base name to be used in resource names; this is usually the cluster's infrastructure ID.

③ Optional: Specify the size of the bootstrap VM. Use a VM size compatible with your specified architecture. If this value is not defined, the default value from the template is set.

4.3.16.1. ARM template for the bootstrap machine

You can use the following Azure Resource Manager (ARM) template to deploy the bootstrap machine that you need for your OpenShift Container Platform cluster:

Example 4.52. 04_bootstrap.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vnetBaseName": {
      "type": "string",
      "defaultValue": "",
      "metadata" : {
        "description" : "The specific customer vnet's base name (optional)"
      }
    },
    "bootstrapIgnition" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Bootstrap ignition content for the bootstrap cluster"
      }
    },
    "sshKeyData" : {
      "type" : "securestring",
      "defaultValue" : "Unused",
      "metadata" : {
        "description" : "Unused"
      }
    },
    "bootstrapVMSize" : {
      "type" : "string",
      "defaultValue" : "Standard_D4s_v3",
      "metadata" : {
        "description" : "The size of the Bootstrap Virtual Machine"
      }
    },
    "hyperVGen": {
      "type": "string",
      "metadata": {
        "description": "VM generation image to use"
      },
      "defaultValue": "V2",
      "allowedValues": [
        "V1",
        "V2"
      ]
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "masterSubnetName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-master-subnet')]",
    "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
    "masterLoadBalancerName" : "[parameters('baseName')]",
    "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal-lb')]",
    "sshKeyPath" : "/home/core/.ssh/authorized_keys",
    "identityName" : "[concat(parameters('baseName'), '-identity')]",
    "vmName" : "[concat(parameters('baseName'), '-bootstrap')]",
    "nicName" : "[concat(variables('vmName'), '-nic')]",
    "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
    "imageName" : "[concat(parameters('baseName'), if(equals(parameters('hyperVGen'), 'V2'), '-
gen2', ''))]",
    "clusterNsgName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-nsg')]",
    "sshPublicIpAddressName" : "[concat(variables('vmName'), '-ssh-pip')]"
  },
  "resources" : [
    {
```

```
   "apiVersion" : "2018-12-01",
   "type" : "Microsoft.Network/publicIPAddresses",
   "name" : "[variables('sshPublicIpAddressName')]",
   "location" : "[variables('location')]",
   "sku": {
    "name": "Standard"
   },
   "properties" : {
    "publicIPAllocationMethod" : "Static",
    "dnsSettings" : {
     "domainNameLabel" : "[variables('sshPublicIpAddressName')]"
    }
   }
  },
  {
   "apiVersion" : "2018-06-01",
   "type" : "Microsoft.Network/networkInterfaces",
   "name" : "[variables('nicName')]",
   "location" : "[variables('location')]",
   "dependsOn" : [
    "[resourceId('Microsoft.Network/publicIPAddresses', variables('sshPublicIpAddressName'))]"
   ],
   "properties" : {
    "ipConfigurations" : [
     {
      "name" : "pipConfig",
      "properties" : {
       "privateIPAllocationMethod" : "Dynamic",
       "publicIPAddress": {
        "id": "[resourceId('Microsoft.Network/publicIPAddresses',
variables('sshPublicIpAddressName'))]"
       },
       "subnet" : {
        "id" : "[variables('masterSubnetRef')]"
       },
       "loadBalancerBackendAddressPools" : [
        {
         "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('masterLoadBalancerName'), '/backendAddressPools/',
variables('masterLoadBalancerName'))]"
        },
        {
         "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('internalLoadBalancerName'), '/backendAddressPools/internal-lb-backend')]"
        }
       ]
      }
     }
    ]
   }
  },
  {
   "apiVersion" : "2018-06-01",
   "type" : "Microsoft.Compute/virtualMachines",
      "name" : "[variables('vmName')]",
      "location" : "[variables('location')]",
      "identity" : {
        "type" : "userAssigned",
        "userAssignedIdentities" : {
          "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',
variables('identityName'))]" : {}
        }
      },
      "dependsOn" : [
        "[concat('Microsoft.Network/networkInterfaces/', variables('nicName'))]"
      ],
      "properties" : {
        "hardwareProfile" : {
          "vmSize" : "[parameters('bootstrapVMSize')]"
        },
        "osProfile" : {
          "computerName" : "[variables('vmName')]",
          "adminUsername" : "core",
          "adminPassword" : "NotActuallyApplied!",
          "customData" : "[parameters('bootstrapIgnition')]",
          "linuxConfiguration" : {
            "disablePasswordAuthentication" : false
          }
        },
        "storageProfile" : {
          "imageReference": {
            "id": "[resourceId('Microsoft.Compute/galleries/images', variables('galleryName'),
variables('imageName'))]"
          },
          "osDisk" : {
            "name": "[concat(variables('vmName'),'_OSDisk')]",
            "osType" : "Linux",
            "createOption" : "FromImage",
            "managedDisk": {
              "storageAccountType": "Premium_LRS"
            },
            "diskSizeGB" : 100
          }
        },
        "networkProfile" : {
          "networkInterfaces" : [
            {
              "id" : "[resourceId('Microsoft.Network/networkInterfaces', variables('nicName'))]"
            }
          ]
        }
      }
    },
    {
      "apiVersion" : "2018-06-01",
      "type": "Microsoft.Network/networkSecurityGroups/securityRules",
      "name" : "[concat(variables('clusterNsgName'), '/bootstrap_ssh_in')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[resourceId('Microsoft.Compute/virtualMachines', variables('vmName'))]"
```

```
          ],
          "properties": {
            "protocol" : "Tcp",
            "sourcePortRange" : "*",
            "destinationPortRange" : "22",
            "sourceAddressPrefix" : "*",
            "destinationAddressPrefix" : "*",
            "access" : "Allow",
            "priority" : 100,
            "direction" : "Inbound"
          }
        }
      ]
    }
```

4.3.17. Creating the control plane machines in Azure

You must create the control plane machines in Microsoft Azure for your cluster to use. One way to create these machines is to modify the provided Azure Resource Manager (ARM) template.

NOTE

By default, Microsoft Azure places control plane machines and compute machines in a pre-set availability zone. You can manually set an availability zone for a compute node or control plane node. To do this, modify a vendor's Azure Resource Manager (ARM) template by specifying each of your availability zones in the **zones** parameter of the virtual machine resource.

If you do not use the provided ARM template to create your control plane machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, consider contacting Red Hat support with your installation logs.

Prerequisites

- Create the bootstrap machine.

Procedure

1. Copy the template from the **ARM template for control plane machines**section of this topic and save it as **05_masters.json** in your cluster's installation directory. This template describes the control plane machines that your cluster requires.

2. Export the following variable needed by the control plane machine deployment:

```
$ export MASTER_IGNITION=`cat <installation_directory>/master.ign | base64 | tr -d '\n'`
```

3. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/05_masters.json" \
  --parameters masterIgnition="${MASTER_IGNITION}" \ 1
```

```
--parameters baseName="${INFRA_ID}" \ 2
--parameters masterVMSize="Standard_D8s_v3" 3
```

1. The Ignition content for the control plane nodes.
2. The base name to be used in resource names; this is usually the cluster’s infrastructure ID.
3. Optional: Specify the size of the Control Plane VM. Use a VM size compatible with your specified architecture. If this value is not defined, the default value from the template is set.

4.3.17.1. ARM template for control plane machines

You can use the following Azure Resource Manager (ARM) template to deploy the control plane machines that you need for your OpenShift Container Platform cluster:

Example 4.53. 05_masters.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vnetBaseName": {
      "type": "string",
      "defaultValue": "",
      "metadata" : {
        "description" : "The specific customer vnet's base name (optional)"
      }
    },
    "masterIgnition" : {
      "type" : "string",
      "metadata" : {
        "description" : "Ignition content for the master nodes"
      }
    },
    "numberOfMasters" : {
      "type" : "int",
      "defaultValue" : 3,
      "minValue" : 2,
      "maxValue" : 30,
      "metadata" : {
        "description" : "Number of OpenShift masters to deploy"
      }
    },
    "sshKeyData" : {
      "type" : "securestring",
      "defaultValue" : "Unused",
      "metadata" : {
        "description" : "Unused"
      }
    },
    "privateDNSZoneName" : {
      "type" : "string",
      "defaultValue" : "",
      "metadata" : {
        "description" : "unused"
      }
    },
    "masterVMSize" : {
      "type" : "string",
      "defaultValue" : "Standard_D8s_v3",
      "metadata" : {
        "description" : "The size of the Master Virtual Machines"
      }
    },
    "diskSizeGB" : {
      "type" : "int",
      "defaultValue" : 1024,
      "metadata" : {
        "description" : "Size of the Master VM OS disk, in GB"
      }
    },
    "hyperVGen": {
      "type": "string",
      "metadata": {
        "description": "VM generation image to use"
      },
      "defaultValue": "V2",
      "allowedValues": [
        "V1",
        "V2"
      ]
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "masterSubnetName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-master-subnet')]",
    "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
    "masterLoadBalancerName" : "[parameters('baseName')]",
    "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal-lb')]",
    "sshKeyPath" : "/home/core/.ssh/authorized_keys",
    "identityName" : "[concat(parameters('baseName'), '-identity')]",
    "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
    "imageName" : "[concat(parameters('baseName'), if(equals(parameters('hyperVGen'), 'V2'), '-
gen2', ''))]",
    "copy" : [
      {
        "name" : "vmNames",
        "count" :  "[parameters('numberOfMasters')]",
        "input" : "[concat(parameters('baseName'), '-master-', copyIndex('vmNames'))]"
      }
    ]
  },
  "resources" : [
    {
      "apiVersion" : "2018-06-01",
      "type" : "Microsoft.Network/networkInterfaces",
      "copy" : {
        "name" : "nicCopy",
        "count" : "[length(variables('vmNames'))]"
      },
      "name" : "[concat(variables('vmNames')[copyIndex()], '-nic')]",
      "location" : "[variables('location')]",
      "properties" : {
        "ipConfigurations" : [
          {
            "name" : "pipConfig",
            "properties" : {
              "privateIPAllocationMethod" : "Dynamic",
              "subnet" : {
                "id" : "[variables('masterSubnetRef')]"
              },
              "loadBalancerBackendAddressPools" : [
                {
                  "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('masterLoadBalancerName'), '/backendAddressPools/',
variables('masterLoadBalancerName'))]"
                },
                {
                  "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('internalLoadBalancerName'), '/backendAddressPools/internal-lb-backend')]"
                }
              ]
            }
          }
        ]
      }
    },
    {
      "apiVersion" : "2018-06-01",
      "type" : "Microsoft.Compute/virtualMachines",
      "copy" : {
        "name" : "vmCopy",
        "count" : "[length(variables('vmNames'))]"
      },
      "name" : "[variables('vmNames')[copyIndex()]]",
      "location" : "[variables('location')]",
      "identity" : {
        "type" : "userAssigned",
        "userAssignedIdentities" : {
          "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',
variables('identityName'))]" : {}
        }
      },
      "dependsOn" : [
        "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')[copyIndex()], '-
nic'))]"
      ],
      "properties" : {
        "hardwareProfile" : {
          "vmSize" : "[parameters('masterVMSize')]"
        },
        "osProfile" : {
          "computerName" : "[variables('vmNames')[copyIndex()]]",
          "adminUsername" : "core",
          "adminPassword" : "NotActuallyApplied!",
          "customData" : "[parameters('masterIgnition')]",
          "linuxConfiguration" : {
            "disablePasswordAuthentication" : false
          }
        },
        "storageProfile" : {
          "imageReference": {
            "id": "[resourceId('Microsoft.Compute/galleries/images', variables('galleryName'),
variables('imageName'))]"
          },
          "osDisk" : {
            "name": "[concat(variables('vmNames')[copyIndex()], '_OSDisk')]",
            "osType" : "Linux",
            "createOption" : "FromImage",
            "caching": "ReadOnly",
            "writeAcceleratorEnabled": false,
            "managedDisk": {
              "storageAccountType": "Premium_LRS"
            },
            "diskSizeGB" : "[parameters('diskSizeGB')]"
          }
        },
        "networkProfile" : {
          "networkInterfaces" : [
            {
              "id" : "[resourceId('Microsoft.Network/networkInterfaces', concat(variables('vmNames')
[copyIndex()], '-nic'))]",
              "properties": {
                "primary": false
              }
            }
          ]
        }
      }
    }
  ]
}
```

4.3.18. Wait for bootstrap completion and remove bootstrap resources in Azure

After you create all of the required infrastructure in Microsoft Azure, wait for the bootstrap process to complete on the machines that you provisioned by using the Ignition config files that you generated with the installation program.

Prerequisites

- Create the control plane machines.

Procedure

1. Change to the directory that contains the installation program and run the following command:

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
    --log-level info 2
```

1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2 To view different installation details, specify **warn**, **debug**, or **error** instead of **info**.

If the command exits without a **FATAL** warning, your production control plane has initialized.

2. Delete the bootstrap resources:

```
$ az network nsg rule delete -g ${RESOURCE_GROUP} --nsg-name ${INFRA_ID}-nsg --name bootstrap_ssh_in
$ az vm stop -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
$ az vm deallocate -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
$ az vm delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap --yes
$ az disk delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap_OSDisk --no-wait --yes
$ az network nic delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-nic --no-wait
$ az storage blob delete --account-key ${ACCOUNT_KEY} --account-name ${CLUSTER_NAME}sa --container-name files --name bootstrap.ign
$ az network public-ip delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-ssh-pip
```

NOTE

If you do not delete the bootstrap server, installation may not succeed due to API traffic being routed to the bootstrap server.

4.3.19. Creating additional worker machines in Azure

You can create worker machines in Microsoft Azure for your cluster to use by launching individual instances discretely or by automated processes outside the cluster, such as auto scaling groups. You can also take advantage of the built-in cluster scaling mechanisms and the machine API in OpenShift Container Platform.

NOTE

If you are installing a three-node cluster, skip this step. A three-node cluster consists of three control plane machines, which also act as compute machines.

In this example, you manually launch one instance by using the Azure Resource Manager (ARM) template. Additional instances can be launched by including additional resources of type **06_workers.json** in the file.

NOTE

By default, Microsoft Azure places control plane machines and compute machines in a pre-set availability zone. You can manually set an availability zone for a compute node or control plane node. To do this, modify a vendor's ARM template by specifying each of your availability zones in the **zones** parameter of the virtual machine resource.

If you do not use the provided ARM template to create your control plane machines, you must review the provided information and manually create the infrastructure. If your cluster does not initialize correctly, consider contacting Red Hat support with your installation logs.

Procedure

1. Copy the template from the **ARM template for worker machines** section of this topic and save it as **06_workers.json** in your cluster's installation directory. This template describes the worker machines that your cluster requires.

2. Export the following variable needed by the worker machine deployment:

```
$ export WORKER_IGNITION=`cat <installation_directory>/worker.ign | base64 | tr -d '\n'`
```

3. Create the deployment by using the **az** CLI:

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/06_workers.json" \
  --parameters workerIgnition="${WORKER_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameters nodeVMSize="Standard_D4s_v3" 3
```

1 The Ignition content for the worker nodes.

2 The base name to be used in resource names; this is usually the cluster's infrastructure ID.

3 Optional: Specify the size of the compute node VM. Use a VM size compatible with your specified architecture. If this value is not defined, the default value from the template is set.

4.3.19.1. ARM template for worker machines

You can use the following Azure Resource Manager (ARM) template to deploy the worker machines that you need for your OpenShift Container Platform cluster:

Example 4.54. 06_workers.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vnetBaseName": {
      "type": "string",
      "defaultValue": "",
      "metadata" : {
        "description" : "The specific customer vnet's base name (optional)"
      }
    },
    "workerIgnition" : {
      "type" : "string",
      "metadata" : {
        "description" : "Ignition content for the worker nodes"
      }
    },
    "numberOfNodes" : {
      "type" : "int",
      "defaultValue" : 3,
      "minValue" : 2,
      "maxValue" : 30,
      "metadata" : {
        "description" : "Number of OpenShift compute nodes to deploy"
      }
    },
    "sshKeyData" : {
      "type" : "securestring",
      "defaultValue" : "Unused",
      "metadata" : {
        "description" : "Unused"
      }
    },
    "nodeVMSize" : {
      "type" : "string",
      "defaultValue" : "Standard_D4s_v3",
      "metadata" : {
        "description" : "The size of the each Node Virtual Machine"
      }
    },
    "hyperVGen": {
      "type": "string",
      "metadata": {
        "description": "VM generation image to use"
      },
      "defaultValue": "V2",
      "allowedValues": [
        "V1",
    "V2"
   ]
  }
 },
 "variables" : {
  "location" : "[resourceGroup().location]",
  "virtualNetworkName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
  "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
  "nodeSubnetName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-worker-subnet')]",
  "nodeSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('nodeSubnetName'))]",
  "infraLoadBalancerName" : "[parameters('baseName')]",
  "sshKeyPath" : "/home/capi/.ssh/authorized_keys",
  "identityName" : "[concat(parameters('baseName'), '-identity')]",
  "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
  "imageName" : "[concat(parameters('baseName'), if(equals(parameters('hyperVGen'), 'V2'), '-
gen2', ''))]",
  "copy" : [
   {
    "name" : "vmNames",
    "count" :  "[parameters('numberOfNodes')]",
    "input" : "[concat(parameters('baseName'), '-worker-', variables('location'), '-',
copyIndex('vmNames', 1))]"
   }
  ]
 },
 "resources" : [
  {
   "apiVersion" : "2019-05-01",
   "name" : "[concat('node', copyIndex())]",
   "type" : "Microsoft.Resources/deployments",
   "copy" : {
    "name" : "nodeCopy",
    "count" : "[length(variables('vmNames'))]"
   },
   "properties" : {
    "mode" : "Incremental",
    "template" : {
     "$schema" : "http://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
     "contentVersion" : "1.0.0.0",
     "resources" : [
      {
       "apiVersion" : "2018-06-01",
       "type" : "Microsoft.Network/networkInterfaces",
       "name" : "[concat(variables('vmNames')[copyIndex()], '-nic')]",
       "location" : "[variables('location')]",
       "properties" : {
        "ipConfigurations" : [
         {
          "name" : "pipConfig",
          "properties" : {
           "privateIPAllocationMethod" : "Dynamic",
                      "subnet" : {
                        "id" : "[variables('nodeSubnetRef')]"
                      }
                    }
                  }
                ]
              }
            },
            {
              "apiVersion" : "2018-06-01",
              "type" : "Microsoft.Compute/virtualMachines",
              "name" : "[variables('vmNames')[copyIndex()]]",
              "location" : "[variables('location')]",
              "tags" : {
                "kubernetes.io-cluster-ffranzupi": "owned"
              },
              "identity" : {
                "type" : "userAssigned",
                "userAssignedIdentities" : {
                  "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',
variables('identityName'))]" : {}
                }
              },
              "dependsOn" : [
                "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')
[copyIndex()], '-nic'))]"
              ],
              "properties" : {
                "hardwareProfile" : {
                  "vmSize" : "[parameters('nodeVMSize')]"
                },
                "osProfile" : {
                  "computerName" : "[variables('vmNames')[copyIndex()]]",
                  "adminUsername" : "capi",
                  "adminPassword" : "NotActuallyApplied!",
                  "customData" : "[parameters('workerIgnition')]",
                  "linuxConfiguration" : {
                    "disablePasswordAuthentication" : false
                  }
                },
                "storageProfile" : {
                  "imageReference": {
                    "id": "[resourceId('Microsoft.Compute/galleries/images', variables('galleryName'),
variables('imageName'))]"
                  },
                  "osDisk" : {
                    "name": "[concat(variables('vmNames')[copyIndex()],'_OSDisk')]",
                    "osType" : "Linux",
                    "createOption" : "FromImage",
                    "managedDisk": {
                      "storageAccountType": "Premium_LRS"
                    },
                    "diskSizeGB": 128
                  }
                },
                "networkProfile" : {
                  "networkInterfaces" : [
                    {
                      "id" : "[resourceId('Microsoft.Network/networkInterfaces',
concat(variables('vmNames')[copyIndex()], '-nic'))]",
                      "properties": {
                        "primary": true
                      }
                    }
                  ]
                }
              }
            }
          ]
        }
      }
    }
  ]
}
```

4.3.20. Installing the OpenShift CLI

You can install the OpenShift CLI (**oc**) to interact with OpenShift Container Platform from a command-line interface. You can install **oc** on Linux, Windows, or macOS.

IMPORTANT

If you installed an earlier version of **oc**, you cannot use it to complete all of the commands in OpenShift Container Platform 4.18. Download and install the new version of **oc**.

Installing the OpenShift CLI on Linux

You can install the OpenShift CLI (**oc**) binary on Linux by using the following procedure.

Procedure

1. Navigate to the OpenShift Container Platform downloads page on the Red Hat Customer Portal.

2. Select the architecture from the **Product Variant** drop-down list.

3. Select the appropriate version from the **Version** drop-down list.

4. Click **Download Now** next to the **OpenShift v4.18 Linux Clients** entry and save the file.

5. Unpack the archive:

```
$ tar xvf <file>
```

6. Place the **oc** binary in a directory that is on your **PATH**.
   To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
$ oc <command>
```

Installing the OpenShift CLI on Windows

You can install the OpenShift CLI (**oc**) binary on Windows by using the following procedure.

Procedure

1. Navigate to the OpenShift Container Platform downloads page on the Red Hat Customer Portal.

2. Select the appropriate version from the **Version** drop-down list.

3. Click **Download Now** next to the **OpenShift v4.18 Windows Client** entry and save the file.

4. Unzip the archive with a ZIP program.

5. Move the **oc** binary to a directory that is on your **PATH**.
   To check your **PATH**, open the command prompt and execute the following command:

```
C:\> path
```

Verification

- After you install the OpenShift CLI, it is available using the **oc** command:

```
C:\> oc <command>
```

Installing the OpenShift CLI on macOS

You can install the OpenShift CLI (**oc**) binary on macOS by using the following procedure.

Procedure

1. Navigate to the OpenShift Container Platform downloads page on the Red Hat Customer Portal.

2. Select the appropriate version from the **Version** drop-down list.

3. Click **Download Now** next to the **OpenShift v4.18 macOS Clients** entry and save the file.

   NOTE

   For macOS arm64, choose the **OpenShift v4.18 macOS arm64 Client** entry.

4. Unpack and unzip the archive.

5. Move the **oc** binary to a directory on your PATH.
   To check your **PATH**, open a terminal and execute the following command:

```
$ echo $PATH
```

Verification

- Verify your installation by using an **oc** command:

```
$ oc <command>
```

4.3.21. Logging in to the cluster by using the CLI

You can log in to your cluster as a default system user by exporting the cluster **kubeconfig** file. The **kubeconfig** file contains information about the cluster that is used by the CLI to connect a client to the correct cluster and API server. The file is specific to a cluster and is created during OpenShift Container Platform installation.

Prerequisites

- You deployed an OpenShift Container Platform cluster.
- You installed the **oc** CLI.

Procedure

1. Export the **kubeadmin** credentials:

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2. Verify you can run **oc** commands successfully using the exported configuration:

```
$ oc whoami
```

Example output

```
system:admin
```

4.3.22. Approving the certificate signing requests for your machines

When you add machines to a cluster, two pending certificate signing requests (CSRs) are generated for each machine that you added. You must confirm that these CSRs are approved or, if necessary, approve them yourself. The client requests must be approved first, followed by the server requests.

Prerequisites

- You added machines to your cluster.

Procedure

1. Confirm that the cluster recognizes the machines:

```
$ oc get nodes
```

Example output

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.31.3
master-1  Ready     master  63m  v1.31.3
master-2  Ready     master  64m  v1.31.3
```

The output lists all of the machines that you created.

NOTE

The preceding output might not include the compute nodes, also known as worker nodes, until some CSRs are approved.

2. Review the pending CSRs and ensure that you see the client requests with the **Pending** or **Approved** status for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME        AGE     REQUESTOR                                                                   CONDITION
csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

In this example, two machines are joining the cluster. You might see more approved CSRs in the list.

3. If the CSRs were not approved, after all of the pending CSRs for the machines you added are in **Pending** status, approve the CSRs for your cluster machines:

NOTE

Because the CSRs rotate automatically, approve your CSRs within an hour of adding the machines to the cluster. If you do not approve them within an hour, the certificates will rotate, and more than two certificates will be present for each node. You must approve all of these certificates. After the client CSR is approved, the Kubelet creates a secondary CSR for the serving certificate, which requires manual approval. Then, subsequent serving certificate renewal requests are automatically approved by the **machine-approver** if the Kubelet requests a new certificate with identical parameters.

NOTE

For clusters running on platforms that are not machine API enabled, such as bare metal and other user-provisioned infrastructure, you must implement a method of automatically approving the kubelet serving certificate requests (CSRs). If a request is not approved, then the **oc exec**, **oc rsh**, and **oc logs** commands cannot succeed, because a serving certificate is required when the API server connects to the kubelet. Any operation that contacts the Kubelet endpoint requires this certificate approval to be in place. The method must watch for new CSRs, confirm that the CSR was submitted by the **node-bootstrapper** service account in the **system:node** or **system:admin** groups, and confirm the identity of the node.

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

NOTE

Some Operators might not become available until some CSRs are approved.

4. Now that your client requests are approved, you must review the server requests for each machine that you added to the cluster:

```
$ oc get csr
```

Example output

```
NAME        AGE     REQUESTOR                                                       CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. If the remaining CSRs are not approved, and are in the **Pending** status, approve the CSRs for your cluster machines:

- To approve them individually, run the following command for each valid CSR:

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- To approve all pending CSRs, run the following command:

  ```
  $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
  {{end}}{{end}}' | xargs oc adm certificate approve
  ```

6. After all client and server CSRs have been approved, the machines have the **Ready** status. Verify this by running the following command:

   ```
   $ oc get nodes
   ```

   Example output

   ```
   NAME      STATUS   ROLES   AGE  VERSION
   master-0  Ready    master  73m  v1.31.3
   master-1  Ready    master  73m  v1.31.3
   master-2  Ready    master  74m  v1.31.3
   worker-0  Ready    worker  11m  v1.31.3
   worker-1  Ready    worker  11m  v1.31.3
   ```

   NOTE

   It can take a few minutes after approval of the server CSRs for the machines to transition to the **Ready** status.

Additional information

- Certificate Signing Requests

4.3.23. Adding the Ingress DNS records

If you removed the DNS Zone configuration when creating Kubernetes manifests and generating Ignition configs, you must manually create DNS records that point at the Ingress load balancer. You can create either a wildcard ***.apps.{baseDomain}.** or specific records. You can use A, CNAME, and other records per your requirements.

Prerequisites

- You deployed an OpenShift Container Platform cluster on Microsoft Azure by using infrastructure that you provisioned.
- Install the OpenShift CLI (**oc**).
- Install or update the Azure CLI.

Procedure

1. Confirm the Ingress router has created a load balancer and populated the **EXTERNAL-IP** field:

   ```
   $ oc -n openshift-ingress get service router-default
   ```

   Example output

```
NAME           TYPE         CLUSTER-IP      EXTERNAL-IP     PORT(S)                AGE
router-default   LoadBalancer   172.30.20.10   35.130.120.110
80:32288/TCP,443:31215/TCP   20
```

2. Export the Ingress router IP as a variable:

```
$ export PUBLIC_IP_ROUTER=`oc -n openshift-ingress get service router-default --no-
headers | awk '{print $4}'`
```

3. Add a ***.apps** record to the public DNS zone.

    a. If you are adding this cluster to a new public zone, run:

    ```
    $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
    z ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER} --ttl 300
    ```

    b. If you are adding this cluster to an already existing public zone, run:

    ```
    $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
    z ${BASE_DOMAIN} -n *.apps.${CLUSTER_NAME} -a ${PUBLIC_IP_ROUTER} --ttl 300
    ```

4. Add a ***.apps** record to the private DNS zone:

    a. Create a ***.apps** record by using the following command:

    ```
    $ az network private-dns record-set a create -g ${RESOURCE_GROUP} -z
    ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps --ttl 300
    ```

    b. Add the ***.apps** record to the private DNS zone by using the following command:

    ```
    $ az network private-dns record-set a add-record -g ${RESOURCE_GROUP} -z
    ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER}
    ```

If you prefer to add explicit domains instead of using a wildcard, you can create entries for each of the cluster's current routes:

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}{"\n"}{end}
{end}' routes
```

Example output

```
oauth-openshift.apps.cluster.basedomain.com
console-openshift-console.apps.cluster.basedomain.com
downloads-openshift-console.apps.cluster.basedomain.com
alertmanager-main-openshift-monitoring.apps.cluster.basedomain.com
prometheus-k8s-openshift-monitoring.apps.cluster.basedomain.com
```

4.3.24. Completing an Azure installation on user-provisioned infrastructure

After you start the OpenShift Container Platform installation on Microsoft Azure user-provisioned infrastructure, you can monitor the cluster events until the cluster is ready.

Prerequisites

- Deploy the bootstrap machine for an OpenShift Container Platform cluster on user-provisioned Azure infrastructure.
- Install the **oc** CLI and log in.

Procedure

- Complete the cluster installation:

  ```
  $ ./openshift-install --dir <installation_directory> wait-for install-complete 1
  ```

  #### Example output

  ```
  INFO Waiting up to 30m0s for the cluster to initialize...
  ```

  1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

IMPORTANT

- The Ignition config files that the installation program generates contain certificates that expire after 24 hours, which are then renewed at that time. If the cluster is shut down before renewing the certificates and the cluster is later restarted after the 24 hours have elapsed, the cluster automatically recovers the expired certificates. The exception is that you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. See the documentation for *Recovering from expired control plane certificates* for more information.
- It is recommended that you use Ignition config files within 12 hours after they are generated because the 24-hour certificate rotates from 16 to 22 hours after the cluster is installed. By using the Ignition config files within 12 hours, you can avoid installation failure if the certificate update runs during installation.

4.3.25. Telemetry access for OpenShift Container Platform

In OpenShift Container Platform 4.18, the Telemetry service, which runs by default to provide metrics about cluster health and the success of updates, requires internet access. If your cluster is connected to the internet, Telemetry runs automatically, and your cluster is registered to OpenShift Cluster Manager.

After you confirm that your OpenShift Cluster Manager inventory is correct, either maintained automatically by Telemetry or manually by using OpenShift Cluster Manager, use subscription watch to track your OpenShift Container Platform subscriptions at the account or multi-cluster level.

Additional resources

- See About remote health monitoring for more information about the Telemetry service

CHAPTER 5. INSTALLING A THREE-NODE CLUSTER ON AZURE

In OpenShift Container Platform version 4.18, you can install a three-node cluster on Microsoft Azure. A three-node cluster consists of three control plane machines, which also act as compute machines. This type of cluster provides a smaller, more resource efficient cluster, for cluster administrators and developers to use for testing, development, and production.

You can install a three-node cluster using either installer-provisioned or user-provisioned infrastructure.

NOTE

Deploying a three-node cluster using an Azure Marketplace image is not supported.

5.1. CONFIGURING A THREE-NODE CLUSTER

You configure a three-node cluster by setting the number of worker nodes to **0** in the **install-config.yaml** file before deploying the cluster. Setting the number of worker nodes to **0** ensures that the control plane machines are schedulable. This allows application workloads to be scheduled to run from the control plane nodes.

NOTE

Because application workloads run from control plane nodes, additional subscriptions are required, as the control plane nodes are considered to be compute nodes.

Prerequisites

- You have an existing **install-config.yaml** file.

Procedure

1. Set the number of compute replicas to **0** in your **install-config.yaml** file, as shown in the following **compute** stanza:

   Example install-config.yaml file for a three-node cluster

   ```
   apiVersion: v1
   baseDomain: example.com
   compute:
   - name: worker
     platform: {}
     replicas: 0
   # ...
   ```

2. If you are deploying a cluster with user-provisioned infrastructure:

   - After you create the Kubernetes manifest files, make sure that the **spec.mastersSchedulable** parameter is set to **true** in **cluster-scheduler-02-config.yml** file. You can locate this file in **<installation_directory>/manifests**. For more information, see "Creating the Kubernetes manifest and Ignition config files" in "Installing a cluster on Azure using ARM templates".

- Do not create additional worker nodes.

Example cluster-scheduler-02-config.yml file for a three-node cluster

```
apiVersion: config.openshift.io/v1
kind: Scheduler
metadata:
  creationTimestamp: null
  name: cluster
spec:
  mastersSchedulable: true
  policy:
    name: ""
status: {}
```

5.2. NEXT STEPS

- Installing a cluster on Azure with customizations
- Installing a cluster on Azure using ARM templates

CHAPTER 6. UNINSTALLING A CLUSTER ON AZURE

You can remove a cluster that you deployed to Microsoft Azure.

6.1. REMOVING A CLUSTER THAT USES INSTALLER-PROVISIONED INFRASTRUCTURE

You can remove a cluster that uses installer-provisioned infrastructure from your cloud.

NOTE

After uninstallation, check your cloud provider for any resources not removed properly, especially with User Provisioned Infrastructure (UPI) clusters. There might be resources that the installer did not create or that the installer is unable to access.

Prerequisites

- You have a copy of the installation program that you used to deploy the cluster.
- You have the files that the installation program generated when you created your cluster.

Procedure

1. From the directory that contains the installation program on the computer that you used to install the cluster, run the following command:

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1 For **<installation_directory>**, specify the path to the directory that you stored the installation files in.

2 To view different details, specify **warn**, **debug**, or **error** instead of **info**.

NOTE

You must specify the directory that contains the cluster definition files for your cluster. The installation program requires the **metadata.json** file in this directory to delete the cluster.

2. Optional: Delete the **<installation_directory>** directory and the OpenShift Container Platform installation program.

6.2. DELETING MICROSOFT AZURE RESOURCES WITH THE CLOUD CREDENTIAL OPERATOR UTILITY

After uninstalling an OpenShift Container Platform cluster that uses short-term credentials managed outside the cluster, you can use the CCO utility (**ccoctl**) to remove the Microsoft Azure (Azure) resources that **ccoctl** created during installation.

Prerequisites

- Extract and prepare the **ccoctl** binary.
- Uninstall an OpenShift Container Platform cluster on Azure that uses short-term credentials.

Procedure

- Delete the Azure resources that **ccoctl** created by running the following command:

```
$ ccoctl azure delete \
  --name=<name> \ 1
  --region=<azure_region> \ 2
  --subscription-id=<azure_subscription_id> \ 3
  --delete-oidc-resource-group
```

1 **<name>** matches the name that was originally used to create and tag the cloud resources.

2 **<azure_region>** is the Azure region in which to delete cloud resources.

3 **<azure_subscription_id>** is the Azure subscription ID for which to delete cloud resources.

Verification

- To verify that the resources are deleted, query Azure. For more information, refer to Azure documentation.

CHAPTER 7. INSTALLATION CONFIGURATION PARAMETERS FOR AZURE

Before you deploy an OpenShift Container Platform cluster on Microsoft Azure, you provide parameters to customize your cluster and the platform that hosts it. When you create the **install-config.yaml** file, you provide values for the required parameters through the command line. You can then modify the **install-config.yaml** file to customize your cluster further.

7.1. AVAILABLE INSTALLATION CONFIGURATION PARAMETERS FOR AZURE

The following tables specify the required, optional, and Azure-specific installation configuration parameters that you can set as part of the installation process.

NOTE

After installation, you cannot modify these parameters in the **install-config.yaml** file.

7.1.1. Required configuration parameters

Required installation configuration parameters are described in the following table:

Table 7.1. Required parameters

Parameter	Description	Values
apiVersion:	The API version for the **install-config.yaml** content. The current version is **v1**. The installation program may also support older API versions.	String
baseDomain:	The base domain of your cloud provider. The base domain is used to create routes to your OpenShift Container Platform cluster components. The full DNS name for your cluster is a combination of the **baseDomain** and **metadata.name** parameter values that uses the **<metadata.name>.<baseDomain>** format.	A fully-qualified domain or subdomain name, such as **example.com**.
metadata:	Kubernetes resource **ObjectMeta**, from which only the **name** parameter is consumed.	Object

Parameter	Description	Values
metadata: name:	The name of the cluster. DNS records for the cluster are all subdomains of **{{.metadata.name}}.{{.baseDomain}}**.	String of lowercase letters, hyphens (**-**), and periods (**.**), such as **dev**.
platform:	The configuration for the specific platform upon which to perform the installation: **aws**, **baremetal**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, or **{}**. For additional information about **platform.<platform>** parameters, consult the table for your specific platform that follows.	Object

pullSecret:	Get a pull secret from Red Hat OpenShift Cluster Manager to authenticate downloading container images for OpenShift Container Platform components from services such as Quay.io.	{
		"auths":{
		"cloud.openshift.com":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		},
		"quay.io":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		}
		}
		}

7.1.2. Network configuration parameters

You can customize your installation configuration based on the requirements of your existing network infrastructure. For example, you can expand the IP address block for the cluster network or provide different IP address blocks than the defaults.

Only IPv4 addresses are supported.

Table 7.2. Network parameters

Parameter	Description	Values

Parameter	Description	Values
`networking:`	The configuration for the cluster network.	Object NOTE You cannot modify parameters specified by the **networking** object after installation.

`networking:`
`  networkType:`
	The Red Hat OpenShift Networking network plugin to install.	**OVNKubernetes**. **OVNKubernetes** is a CNI plugin for Linux networks and hybrid networks that contain both Linux and Windows servers. The default value is **OVNKubernetes**.

`networking:`
`  clusterNetwork:`
	The IP address blocks for pods.
	The default value is **10.128.0.0/14** with a host prefix of **/23**.
	If you specify multiple IP address blocks, the blocks must not overlap.
		An array of objects. For example:
		`networking:`
		`  clusterNetwork:`
		`  - cidr: 10.128.0.0/14`
		`    hostPrefix: 23`

`networking:`
`  clusterNetwork:`
`    cidr:`
	Required if you use **networking.clusterNetwork**. An IP address block.
	An IPv4 network.
		An IP address block in Classless Inter-Domain Routing (CIDR) notation. The prefix length for an IPv4 block is between **0** and **32**.

`networking:`
`  clusterNetwork:`
`    hostPrefix:`
	The subnet prefix length to assign to each individual node. For example, if **hostPrefix** is set to **23** then each node is assigned a **/23** subnet out of the given **cidr**. A **hostPrefix** value of **23** provides 510 (2^(32 - 23) - 2) pod IP addresses.	A subnet prefix.
		The default value is **23**.

`networking:`
`  serviceNetwork:`
	The IP address block for services. The default value is **172.30.0.0/16**.
	The OVN-Kubernetes network plugins supports only a single IP address block for the service network.
		An array with an IP address block in CIDR format. For example:
		`networking:`
		`  serviceNetwork:`
		`   - 172.30.0.0/16`

Parameter	Description	Values

`networking:`
`  machineNetwork:`
	The IP address blocks for machines.
	If you specify multiple IP address blocks, the blocks must not overlap.
		An array of objects. For example:
		`networking:`
		`  machineNetwork:`
		`  - cidr: 10.0.0.0/16`

`networking:`
`  machineNetwork:`
`    cidr:`
	Required if you use **networking.machineNetwork**. An IP address block. The default value is **10.0.0.0/16** for all platforms other than libvirt and IBM Power® Virtual Server. For libvirt, the default value is **192.168.126.0/24**. For IBM Power® Virtual Server, the default value is **192.168.0.0/24**.	An IP network block in CIDR notation.
		For example, **10.0.0.0/16**.
		NOTE
		Set the **networking.machineNetwork** to match the CIDR that the preferred NIC resides in.

`networking:`
`  ovnKubernetesConfig:`
`    ipv4:`
`      internalJoinSubnet:`
	Configures the IPv4 join subnet that is used internally by **ovn-kubernetes**. This subnet must not overlap with any other subnet that OpenShift Container Platform is using, including the node network. The size of the subnet must be larger than the number of nodes. You cannot change the value after installation.	An IP network block in CIDR notation. The default value is **100.64.0.0/16**.

7.1.3. Optional configuration parameters

Optional installation configuration parameters are described in the following table:

Table 7.3. Optional parameters

Parameter	Description	Values
`additionalTrustBundle:`	A PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store. This trust bundle may also be used when a proxy has been configured.	String

Parameter	Description	Values
capabilities:	Controls the installation of optional core cluster components. You can reduce the footprint of your OpenShift Container Platform cluster by disabling optional components. For more information, see the "Cluster capabilities" page in *Installing*.	String array
capabilities: baselineCapabilitySet:	Selects an initial set of optional capabilities to enable. Valid values are **None**, **v4.11**, **v4.12** and **vCurrent**. The default value is **vCurrent**.	String
capabilities: additionalEnabledCapabilities:	Extends the set of optional capabilities beyond what you specify in **baselineCapabilitySet**. You may specify multiple capabilities in this parameter.	String array
cpuPartitioningMode:	Enables workload partitioning, which isolates OpenShift Container Platform services, cluster management workloads, and infrastructure pods to run on a reserved set of CPUs. Workload partitioning can only be enabled during installation and cannot be disabled after installation. While this field enables workload partitioning, it does not configure workloads to use specific CPUs. For more information, see the *Workload partitioning* page in the *Scalability and Performance* section.	**None** or **AllNodes**. **None** is the default value.
compute:	The configuration for the machines that comprise the compute nodes.	Array of **MachinePool** objects.

Parameter	Description	Values
compute: architecture:	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are **amd64** and **arm64**. Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see *Supported installation methods for different platforms* in *Selecting a cluster installation method and preparing it for users*.	String
compute: hyperthreading:	Whether to enable or disable simultaneous multithreading, or **hyperthreading**, on compute machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	**Enabled** or **Disabled**
compute: name:	Required if you use **compute**. The name of the machine pool.	**worker**
compute: platform:	Required if you use **compute**. Use this parameter to specify the cloud provider to host the worker machines. This parameter value must match the **controlPlane.platform** parameter value.	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, or **{}**

Parameter	Description	Values
compute: replicas:	The number of compute machines, which are also known as worker machines, to provision.	A positive integer greater than or equal to **2**. The default value is **3**.
featureSet:	Enables the cluster for a feature set. A feature set is a collection of OpenShift Container Platform features that are not enabled by default. For more information about enabling a feature set during installation, see "Enabling features using feature gates".	String. The name of the feature set to enable, such as **TechPreviewNoUpgrade**.
controlPlane:	The configuration for the machines that comprise the control plane.	Array of **MachinePool** objects.
controlPlane: architecture:	Determines the instruction set architecture of the machines in the pool. Currently, clusters with varied architectures are not supported. All pools must specify the same architecture. Valid values are **amd64** and **arm64**. Not all installation options support the 64-bit ARM architecture. To verify if your installation option is supported on your platform, see *Supported installation methods for different platforms* in *Selecting a cluster installation method and preparing it for users*.	String
controlPlane: hyperthreading:	Whether to enable or disable simultaneous multithreading, or **hyperthreading**, on control plane machines. By default, simultaneous multithreading is enabled to increase the performance of your machines' cores. IMPORTANT If you disable simultaneous multithreading, ensure that your capacity planning accounts for the dramatically decreased machine performance.	**Enabled** or **Disabled**

Parameter	Description	Values
controlPlane: name:	Required if you use **controlPlane**. The name of the machine pool.	**master**
controlPlane: platform:	Required if you use **controlPlane**. Use this parameter to specify the cloud provider that hosts the control plane machines. This parameter value must match the **compute.platform** parameter value.	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, or **{}**
controlPlane: replicas:	The number of control plane machines to provision.	Supported values are **3**, or **1** when deploying single-node OpenShift.
credentialsMode:	The Cloud Credential Operator (CCO) mode. If no mode is specified, the CCO dynamically tries to determine the capabilities of the provided credentials, with a preference for mint mode on the platforms where multiple modes are supported. NOTE Not all CCO modes are supported for all cloud providers. For more information about CCO modes, see the "Managing cloud provider credentials" entry in the *Authentication and authorization* content.	**Mint**, **Passthrough**, **Manual** or an empty string (**""**).

Parameter	Description	Values
`fips:`	Enable or disable FIPS mode. The default is **false** (disabled). If FIPS mode is enabled, the Red Hat Enterprise Linux CoreOS (RHCOS) machines that OpenShift Container Platform runs on bypass the default Kubernetes cryptography suite and use the cryptography modules that are provided with RHCOS instead. **IMPORTANT** To enable FIPS mode for your cluster, you must run the installation program from a Red Hat Enterprise Linux (RHEL) computer configured to operate in FIPS mode. For more information about configuring FIPS mode on RHEL, see Switching RHEL to FIPS mode. When running Red Hat Enterprise Linux (RHEL) or Red Hat Enterprise Linux CoreOS (RHCOS) booted in FIPS mode, OpenShift Container Platform core components use the RHEL cryptographic libraries that have been submitted to NIST for FIPS 140-2/140-3 Validation on only the x86_64, ppc64le, and s390x architectures. **NOTE** If you are using Azure File storage, you cannot enable FIPS mode.	**false** or **true**

Parameter	Description	Values
imageContentSources:	Sources and repositories for the release-image content.	Array of objects. Includes a **source** and, optionally, **mirrors**, as described in the following rows of this table.
imageContentSources: source:	Required if you use **imageContentSources**. Specify the repository that users refer to, for example, in image pull specifications.	String
imageContentSources: mirrors:	Specify one or more repositories that may also contain the same images.	Array of strings
publish:	How to publish or expose the user-facing endpoints of your cluster, such as the Kubernetes API, OpenShift routes.	**Internal**, **External**, or **Mixed**. To deploy a private cluster, which cannot be accessed from the internet, set **publish** to **Internal**. The default value is **External**. To deploy a cluster where the API and the ingress server have different publishing strategies, set **publish** to **Mixed** and use the **operatorPublishingStrategy** parameter.
sshKey:	The SSH key to authenticate access to your cluster machines. NOTE For production OpenShift Container Platform clusters on which you want to perform installation debugging or disaster recovery, specify an SSH key that your **ssh-agent** process uses.	For example, **sshKey: ssh-ed25519 AAAA..**.

IMPORTANT

Setting this parameter to **Manual** enables alternatives to storing administrator-level secrets in the **kube-system** project, which require additional configuration steps. For more information, see "Alternatives to storing administrator-level secrets in the kube-system project".

7.1.4. Additional Azure configuration parameters

Additional Azure configuration parameters are described in the following table.

NOTE

By default, if you specify availability zones in the **install-config.yaml** file, the installation program distributes the control plane machines and the compute machines across these availability zones within a region. To ensure high availability for your cluster, select a region with at least three availability zones. If your region contains fewer than three availability zones, the installation program places more than one control plane machine in the available zones.

Table 7.4. Additional Azure parameters

Parameter	Description	Values

`compute:`
`  platform:`
`    azure:`
`      encryptionAtHost:`
	Enables host-level encryption for compute machines. You can enable this encryption alongside user-managed server-side encryption. This feature encrypts temporary, ephemeral, cached and un-managed disks on the VM host. This is not a prerequisite for user-managed server-side encryption.	**true** or **false**. The default is **false**.

`compute:`
`  platform:`
`    azure:`
`      osDisk:`
`        diskSizeGB:`
	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is **128**.

`compute:`
`  platform:`
`    azure:`
`      osDisk:`
`        diskType:`
	Defines the type of disk.	**standard_LRS**, **premium_LRS**, or **standardSSD_LRS**. The default is **premium_LRS**.

`compute:`
`  platform:`
`    azure:`
`      ultraSSDCapability:`
	Enables the use of Azure ultra disks for persistent storage on compute nodes. This requires that your Azure region and zone have ultra disks available.	**Enabled**, **Disabled**. The default is **Disabled**.

Parameter	Description	Values
compute: platform: azure: osDisk: diskEncryptionSet: resourceGroup:	The name of the Azure resource group that contains the disk encryption set from the installation prerequisites. This resource group should be different from the resource group where you install the cluster to avoid deleting your Azure encryption key when the cluster is destroyed. This value is only necessary if you intend to install the cluster with user-managed disk encryption.	String, for example **production_encryption_resource_group**.
compute: platform: azure: osDisk: diskEncryptionSet: name:	The name of the disk encryption set that contains the encryption key from the installation prerequisites.	String, for example **production_disk_encryption_set**.
compute: platform: azure: osDisk: diskEncryptionSet: subscriptionId:	Defines the Azure subscription of the disk encryption set where the disk encryption set resides. This secondary disk encryption set is used to encrypt compute machines.	String, in the format **00000000-0000-0000-0000-000000000000**.
compute: platform: azure: osImage: publisher:	Optional. By default, the installation program downloads and installs the Red Hat Enterprise Linux CoreOS (RHCOS) image that is used to boot compute machines. You can override the default behavior by using a custom RHCOS image that is available from the Azure Marketplace. The installation program uses this image for compute machines only.	String. The name of the image publisher.
compute: platform: azure: osImage: offer:	The name of Azure Marketplace offer that is associated with the custom RHCOS image. If you use **compute.platform.azure.osImage.publisher**, this field is required.	String. The name of the image offer.

Parameter	Description	Values

compute:
  platform:
    azure:
      osImage:
        sku:
	An instance of the Azure Marketplace offer. If you use **compute.platform.azure.osImage.publisher**, this field is required.	String. The SKU of the image offer.

compute:
  platform:
    azure:
      osImage:
        version:
	The version number of the image SKU. If you use **compute.platform.azure.osImage.publisher**, this field is required.	String. The version of the image to use.

compute:
  platform:
    azure:
      vmNetworkingType:
	Enables accelerated networking. Accelerated networking enables single root I/O virtualization (SR-IOV) to a VM, improving its networking performance. If instance type of compute machines support **Accelerated** networking, by default, the installer enables **Accelerated** networking, otherwise the default networking type is **Basic**.	**Accelerated** or **Basic**.

compute:
  platform:
    azure:
      type:
	Defines the Azure instance type for compute machines.	String

compute:
  platform:
    azure:
      zones:
	The availability zones where the installation program creates compute machines.	String list

compute:
  platform:
    azure:
      settings:
        securityType:
	Enables confidential VMs or trusted launch for compute nodes. This option is not enabled by default.	**ConfidentialVM** or **TrustedLaunch**.

Parameter	Description	Values
compute: platform: azure: settings: confidentialVM: uefiSettings: secureBoot:	Enables secure boot on compute nodes if you are using confidential VMs.	**Enabled** or **Disabled**. The default is **Disabled**.
compute: platform: azure: settings: confidentialVM: uefiSettings: virtualizedTrustedPlatformModule:	Enables the virtualized Trusted Platform Module (vTPM) feature on compute nodes if you are using confidential VMs.	**Enabled** or **Disabled**. The default is **Disabled**.
compute: platform: azure: settings: trustedLaunch: uefiSettings: secureBoot:	Enables secure boot on compute nodes if you are using trusted launch.	**Enabled** or **Disabled**. The default is **Disabled**.
compute: platform: azure: settings: trustedLaunch: uefiSettings: virtualizedTrustedPlatformModule:	Enables the vTPM feature on compute nodes if you are using trusted launch.	**Enabled** or **Disabled**. The default is **Disabled**.

Parameter	Description	Values
compute: platform: azure: osDisk: securityProfile: securityEncryptionType:	Enables the encryption of the virtual machine guest state for compute nodes. This parameter can only be used if you use Confidential VMs.	**VMGuestStateOnly** is the only supported value.
controlPlane: platform: azure: settings: securityType:	Enables confidential VMs or trusted launch for control plane nodes. This option is not enabled by default.	**ConfidentialVM** or **TrustedLaunch**.
controlPlane: platform: azure: settings: confidentialVM: uefiSettings: secureBoot:	Enables secure boot on control plane nodes if you are using confidential VMs.	**Enabled** or **Disabled**. The default is **Disabled**.
controlPlane: platform: azure: settings: confidentialVM: uefiSettings: virtualizedTrustedPlatformModule:	Enables the vTPM feature on control plane nodes if you are using confidential VMs.	**Enabled** or **Disabled**. The default is **Disabled**.

Parameter	Description	Values

controlPlane:
  platform:
    azure:
      settings:
trustedLaunch:
          uefiSettings:
secureBoot:
	Enables secure boot on control plane nodes if you are using trusted launch.	**Enabled** or **Disabled**. The default is **Disabled**.

controlPlane:
  platform:
    azure:
      settings:
trustedLaunch:
          uefiSettings:
virtualizedTrustedPlatformModule:
	Enables the vTPM feature on control plane nodes if you are using trusted launch.	**Enabled** or **Disabled**. The default is **Disabled**.

controlPlane:
  platform:
    azure:
      osDisk:
securityProfile:
securityEncryptionType:
	Enables the encryption of the virtual machine guest state for control plane nodes. This parameter can only be used if you use Confidential VMs.	**VMGuestStateOnly** is the only supported value.

controlPlane:
  platform:
    azure:
      type:
	Defines the Azure instance type for control plane machines.	String

controlPlane:
  platform:
    azure:
      zones:
	The availability zones where the installation program creates control plane machines.	String list

Parameter	Description	Values
platform: azure: defaultMachinePlatform: settings: securityType:	Enables confidential VMs or trusted launch for all nodes. This option is not enabled by default.	**ConfidentialVM** or **TrustedLaunch**.
platform: azure: defaultMachinePlatform: settings: confidentialVM: uefiSettings: secureBoot:	Enables secure boot on all nodes if you are using confidential VMs.	**Enabled** or **Disabled**. The default is **Disabled**.
platform: azure: defaultMachinePlatform: settings: confidentialVM: uefiSettings: virtualizedTrustedPlatformModule:	Enables the virtualized Trusted Platform Module (vTPM) feature on all nodes if you are using confidential VMs.	**Enabled** or **Disabled**. The default is **Disabled**.
platform: azure: defaultMachinePlatform: settings: trustedLaunch: uefiSettings: secureBoot:	Enables secure boot on all nodes if you are using trusted launch.	**Enabled** or **Disabled**. The default is **Disabled**.

Parameter	Description	Values
platform: azure: defaultMachinePlatform: settings: trustedLaunch: uefiSettings: virtualizedTrustedPlatformModule:	Enables the vTPM feature on all nodes if you are using trusted launch.	**Enabled** or **Disabled**. The default is **Disabled**.
platform: azure: defaultMachinePlatform: osDisk: securityProfile: securityEncryptionType:	Enables the encryption of the virtual machine guest state for all nodes. This parameter can only be used if you use Confidential VMs.	**VMGuestStateOnly** is the only supported value.
platform: azure: defaultMachinePlatform: encryptionAtHost:	Enables host-level encryption for compute machines. You can enable this encryption alongside user-managed server-side encryption. This feature encrypts temporary, ephemeral, cached, and un-managed disks on the VM host. This parameter is not a prerequisite for user-managed server-side encryption.	**true** or **false**. The default is **false**.
platform: azure: defaultMachinePlatform: osDisk: diskEncryptionSet: name:	The name of the disk encryption set that contains the encryption key from the installation prerequisites.	String, for example, **production_disk_encryption_set**.

Parameter	Description	Values

platform:
  azure:
    defaultMachinePlatform:
      osDisk:
        diskEncryptionSet:
          resourceGroup:
	The name of the Azure resource group that contains the disk encryption set from the installation prerequisites. To avoid deleting your Azure encryption key when the cluster is destroyed, this resource group must be different from the resource group where you install the cluster. This value is necessary only if you intend to install the cluster with user-managed disk encryption.	String, for example, **production_encryption_resource_group**.

platform:
  azure:
    defaultMachinePlatform:
      osDisk:
        diskEncryptionSet:
          subscriptionId:
	Defines the Azure subscription of the disk encryption set where the disk encryption set resides. This secondary disk encryption set is used to encrypt compute machines.	String, in the format **00000000-0000-0000-0000-000000000000**.

platform:
  azure:
    defaultMachinePlatform:
      osDisk:
        diskSizeGB:
	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is **128**.

platform:
  azure:
    defaultMachinePlatform:
      osDisk:
        diskType:
	Defines the type of disk.	**premium_LRS** or **standardSSD_LRS**. The default is **premium_LRS**.

Parameter	Description	Values
platform: azure: defaultMachinePlatform: osImage: publisher:	Optional. By default, the installation program downloads and installs the Red Hat Enterprise Linux CoreOS (RHCOS) image that is used to boot control plane and compute machines. You can override the default behavior by using a custom RHCOS image that is available from the Azure Marketplace. The installation program uses this image for both types of machines. Control plane machines do not contribute to licensing costs when using the default image, but if you apply an Azure Marketplace image for a control plane machine, usage costs will apply.	String. The name of the image publisher.
platform: azure: defaultMachinePlatform: osImage: offer:	The name of Azure Marketplace offer that is associated with the custom RHCOS image. If you use **platform.azure.defaultMachinePlatform.osImage.publisher**, this field is required.	String. The name of the image offer.
platform: azure: defaultMachinePlatform: osImage: sku:	An instance of the Azure Marketplace offer. If you use **platform.azure.defaultMachinePlatform.osImage.publisher**, this field is required.	String. The SKU of the image offer.
platform: azure: defaultMachinePlatform: osImage: version:	The version number of the image SKU. If you use **platform.azure.defaultMachinePlatform.osImage.publisher**, this field is required.	String. The version of the image to use.

Parameter	Description	Values
platform: azure: defaultMachinePlatform: type:	The Azure instance type for control plane and compute machines.	The Azure instance type.
platform: azure: defaultMachinePlatform: zones:	The availability zones where the installation program creates compute and control plane machines.	String list.
controlPlane: platform: azure: encryptionAtHost:	Enables host-level encryption for control plane machines. You can enable this encryption alongside user-managed server-side encryption. This feature encrypts temporary, ephemeral, cached and un-managed disks on the VM host. This is not a prerequisite for user-managed server-side encryption.	**true** or **false**. The default is **false**.
controlPlane: platform: azure: osDisk: diskEncryptionSet: resourceGroup:	The name of the Azure resource group that contains the disk encryption set from the installation prerequisites. This resource group should be different from the resource group where you install the cluster to avoid deleting your Azure encryption key when the cluster is destroyed. This value is only necessary if you intend to install the cluster with user-managed disk encryption.	String, for example **production_encryption_resource_group**.
controlPlane: platform: azure: osDisk: diskEncryptionSet: name:	The name of the disk encryption set that contains the encryption key from the installation prerequisites.	String, for example **production_disk_encryption_set**.

Parameter	Description	Values
controlPlane: platform: azure: osDisk: diskEncryptionSet: subscriptionId:	Defines the Azure subscription of the disk encryption set where the disk encryption set resides. This secondary disk encryption set is used to encrypt control plane machines.	String, in the format **00000000-0000-0000-0000-000000000000**.
controlPlane: platform: azure: osDisk: diskSizeGB:	The Azure disk size for the VM.	Integer that represents the size of the disk in GB. The default is **1024**.
controlPlane: platform: azure: osDisk: diskType:	Defines the type of disk.	**premium_LRS** or **standardSSD_LRS**. The default is **premium_LRS**.
controlPlane: platform: azure: osImage: publisher:	Optional. By default, the installation program downloads and installs the Red Hat Enterprise Linux CoreOS (RHCOS) image that is used to boot control plane machines. You can override the default behavior by using a custom RHCOS image that is available from the Azure Marketplace. The installation program uses this image for control plane machines only. Control plane machines do not contribute to licensing costs when using the default image, but if you apply an Azure Marketplace image for a control plane machine, usage costs will apply.	String. The name of the image publisher.
controlPlane: platform: azure: osImage: offer:	The name of Azure Marketplace offer that is associated with the custom RHCOS image. If you use **controlPlane.platform.azure.osImage.publisher**, this field is required.	String. The name of the image offer.

Parameter	Description	Values
controlPlane: platform: azure: osImage: sku:	An instance of the Azure Marketplace offer. If you use **controlPlane.platform.azure.osImage.publisher**, this field is required.	String. The SKU of the image offer.
controlPlane: platform: azure: osImage: version:	The version number of the image SKU. If you use **controlPlane.platform.azure.osImage.publisher**, this field is required.	String. The version of the image to use.
controlPlane: platform: azure: ultraSSDCapability:	Enables the use of Azure ultra disks for persistent storage on control plane machines. This requires that your Azure region and zone have ultra disks available.	**Enabled**, **Disabled**. The default is **Disabled**.
controlPlane: platform: azure: vmNetworkingType:	Enables accelerated networking. Accelerated networking enables single root I/O virtualization (SR-IOV) to a VM, improving its networking performance. If instance type of control plane machines support **Accelerated** networking, by default, the installer enables **Accelerated** networking, otherwise the default networking type is **Basic**.	**Accelerated** or **Basic**.
platform: azure: baseDomainResourceGroupName:	The name of the resource group that contains the DNS zone for your base domain.	String, for example **production_cluster**.

Parameter	Description	Values

`platform:`
`  azure:`
`    resourceGroupName:`
	The name of an already existing resource group to install your cluster to. This resource group must be empty and only used for this specific cluster; the cluster components assume ownership of all resources in the resource group. If you limit the service principal scope of the installation program to this resource group, you must ensure all other resources used by the installation program in your environment have the necessary permissions, such as the public DNS zone and virtual network. Destroying the cluster by using the installation program deletes this resource group.	String, for example **existing_resource_group**.

Parameter	Description	Values

platform:
  azure:
    outboundType:
	The outbound routing strategy used to connect your cluster to the internet. If you are using user-defined routing, you must have pre-existing networking available where the outbound routing has already been configured prior to installing a cluster. The installation program is not responsible for configuring user-defined routing. If you specify the **NatGateway** routing strategy, the installation program will only create one NAT gateway. If you specify the **NatGateway** routing strategy, your account must have the **Microsoft.Network/natGateways/read** and **Microsoft.Network/natGateways/write** permissions.
	IMPORTANT
	NatGateway is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.
	For more information about the support scope of Red Hat Technology Preview features, see Technology Preview Features Support Scope.
		LoadBalancer, **UserDefinedRouting**, or **NatGateway**. The default is **LoadBalancer**.

Parameter	Description	Values
platform: azure: region:	The name of the Azure region that hosts your cluster.	Any valid region name, such as **centralus**.
platform: azure: zone:	List of availability zones to place machines in. For high availability, specify at least two zones.	List of zones, for example **["1", "2", "3"]**.
platform: azure: customerManagedKey: keyVault: name:	Specifies the name of the key vault that contains the encryption key that is used to encrypt Azure storage.	String.
platform: azure: customerManagedKey: keyVault: keyName:	Specifies the name of the user-managed encryption key that is used to encrypt Azure storage.	String.
platform: azure: customerManagedKey: keyVault: resourceGroup:	Specifies the name of the resource group that contains the key vault and managed identity.	String.
platform: azure: customerManagedKey: keyVault: subscriptionId:	Specifies the subscription ID that is associated with the key vault.	String, in the format **00000000-0000-0000-0000-000000000000**.

Parameter	Description	Values
platform: azure: customerManagedKey: userAssignedIdentityKey:	Specifies the name of the user-assigned managed identity that resides in the resource group with the key vault and has access to the user-managed key.	String.
platform: azure: defaultMachinePlatform: ultraSSDCapability:	Enables the use of Azure ultra disks for persistent storage on control plane and compute machines. This requires that your Azure region and zone have ultra disks available.	**Enabled**, **Disabled**. The default is **Disabled**.
platform: azure: networkResourceGroupName:	The name of the resource group that contains the existing VNet that you want to deploy your cluster to. This name cannot be the same as the **platform.azure.baseDomainResourceGroupName**.	String.
platform: azure: virtualNetwork:	The name of the existing VNet that you want to deploy your cluster to.	String.
platform: azure: controlPlaneSubnet:	The name of the existing subnet in your VNet that you want to deploy your control plane machines to.	Valid CIDR, for example **10.0.0.0/16**.
platform: azure: computeSubnet:	The name of the existing subnet in your VNet that you want to deploy your compute machines to.	Valid CIDR, for example **10.0.0.0/16**.

Parameter	Description	Values
platform: azure: cloudName:	The name of the Azure cloud environment that is used to configure the Azure SDK with the appropriate Azure API endpoints. If empty, the default value **AzurePublicCloud** is used.	Any valid cloud environment, such as **AzurePublicCloud** or **AzureUSGovernmentCloud**.
platform: azure: defaultMachinePlatform: vmNetworkingType:	Enables accelerated networking. Accelerated networking enables single root I/O virtualization (SR-IOV) to a VM, improving its networking performance.	**Accelerated** or **Basic**. If instance type of control plane and compute machines support **Accelerated** networking, by default, the installer enables **Accelerated** networking, otherwise the default networking type is **Basic**.
operatorPublishingStrategy: apiserver:	Determines whether the load balancers that service the API are public or private. Set this parameter to **Internal** to prevent the API server from being accessible outside of your VNet. Set this parameter to **External** to make the API server accessible outside of your VNet. If you set this parameter, you must set the **publish** parameter to **Mixed**.	**External** or **Internal**. The default value is **External**.
operatorPublishingStrategy: ingress:	Determines whether the DNS resources that the cluster creates for ingress traffic are publicly visible. Set this parameter to **Internal** to prevent the ingress VIP from being publicly accessible. Set this parameter to **External** to make the ingress VIP publicly accessible. If you set this parameter, you must set the **publish** parameter to **Mixed**.	**External** or **Internal**. The default value is **External**.

NOTE

You cannot customize Azure Availability Zones or Use tags to organize your Azure resources with an Azure cluster.