



OpenShift Container Platform 4.20

Backup and restore

Backing up and restoring your OpenShift Container Platform cluster

OpenShift Container Platform 4.20 Backup and restore

Backing up and restoring your OpenShift Container Platform cluster

Legal Notice

Copyright © Red Hat.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat Software Collections is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This document provides instructions for backing up your cluster's data and for recovering from various disaster scenarios.

Table of Contents

CHAPTER 1. BACKUP AND RESTORE	12
1.1. CONTROL PLANE BACKUP AND RESTORE OPERATIONS	12
1.2. APPLICATION BACKUP AND RESTORE OPERATIONS	12
1.2.1. OADP requirements	12
1.2.2. Backing up and restoring applications	13
CHAPTER 2. SHUTTING DOWN THE CLUSTER GRACEFULLY	14
2.1. PREREQUISITES	14
2.2. SHUTTING DOWN THE CLUSTER	14
2.3. ADDITIONAL RESOURCES	16
CHAPTER 3. RESTARTING THE CLUSTER GRACEFULLY	17
3.1. PREREQUISITES	17
3.2. RESTARTING THE CLUSTER	17
CHAPTER 4. HIBERNATING AN OPENSIFT CONTAINER PLATFORM CLUSTER	21
4.1. ABOUT CLUSTER HIBERNATION	21
4.2. PREREQUISITES	21
4.3. HIBERNATING A CLUSTER	21
4.4. RESUMING A HIBERNATED CLUSTER	23
CHAPTER 5. OADP APPLICATION BACKUP AND RESTORE	26
5.1. INTRODUCTION TO OPENSIFT API FOR DATA PROTECTION	26
5.1.1. OpenShift API for Data Protection APIs	26
5.1.1.1. Support for OpenShift API for Data Protection	26
5.1.1.1.1. Unsupported versions of the OADP Operator	27
5.2. OADP RELEASE NOTES	28
5.2.1. OADP 1.5 release notes	28
5.2.1.1. OADP 1.5.3 release notes	28
5.2.1.2. OADP 1.5.2 release notes	28
5.2.1.2.1. Resolved issues	28
5.2.1.3. OADP 1.5.1 release notes	29
5.2.1.3.1. New features	29
5.2.1.3.2. Resolved issues	30
5.2.1.3.3. Known issues	31
5.2.1.3.4. Deprecated features	33
5.2.1.4. OADP 1.5.0 release notes	33
5.2.1.4.1. New features	33
5.2.1.4.2. Resolved issues	35
5.2.1.4.3. Known issues	38
5.2.1.4.4. Deprecated features	38
5.2.1.4.5. Technology Preview	38
5.2.1.5. Upgrading OADP 1.4 to 1.5	39
5.2.1.5.1. Changes from OADP 1.4 to 1.5	39
5.2.1.5.2. Backing up the DPA configuration	39
5.2.1.5.3. Upgrading the OADP Operator	40
5.2.1.5.4. Converting DPA to the new version for OADP 1.5.0	40
5.2.1.5.5. Verifying the upgrade	41
5.3. OADP PERFORMANCE	42
5.3.1. OADP recommended network settings	42
5.4. OADP FEATURES AND PLUGINS	43
5.4.1. OADP features	43

5.4.2. OADP plugins	44
5.4.3. About OADP Velero plugins	45
5.4.3.1. Default Velero cloud provider plugins	45
5.4.3.2. Custom Velero plugins	46
5.4.4. Supported architectures for OADP	46
5.4.5. OADP support for IBM Power and IBM Z	47
5.4.5.1. OADP support for target backup locations using IBM Power	47
5.4.5.2. OADP testing and support for target backup locations using IBM Z	47
5.4.5.2.1. Known issue of OADP using IBM Power(R) and IBM Z(R) platforms	48
5.4.6. OADP and FIPS	48
5.4.7. Avoiding the Velero plugin panic error	48
5.4.8. Workaround for OpenShift ADP Controller segmentation fault	49
5.5. OADP USE CASES	49
5.5.1. Backup using OpenShift API for Data Protection and Red Hat OpenShift Data Foundation (ODF)	49
5.5.1.1. Backing up an application using OADP and ODF	49
5.5.2. OpenShift API for Data Protection (OADP) restore use case	54
5.5.2.1. Restoring an application to a different namespace using OADP	54
5.5.3. Including a self-signed CA certificate during backup	55
5.5.3.1. Backing up an application and its self-signed CA certificate	55
5.5.4. Using the legacy-aws Velero plugin	60
5.5.4.1. Using the legacy-aws Velero plugin in the DataProtectionApplication CR	60
5.5.5. Backing up workloads on OADP with OpenShift Container Platform	63
5.5.5.1. Performing a backup with OADP and OpenShift Container Platform	63
5.5.5.2. Cleaning up a cluster after a backup with OADP and ROSA STS	65
5.6. INSTALLING OADP	67
5.6.1. About installing OADP	67
5.6.1.1. AWS S3 compatible backup storage providers	68
5.6.1.1.1. Certified backup storage providers	68
5.6.1.1.2. Unsupported backup storage providers	69
5.6.1.1.3. Backup storage providers with known limitations	69
5.6.1.2. Configuring Multicloud Object Gateway (MCG) for disaster recovery on OpenShift Data Foundation	69
5.6.1.3. About OADP update channels	70
5.6.1.4. Installation of OADP on multiple namespaces	71
5.6.1.5. OADP support for backup data immutability	71
5.6.1.6. Velero CPU and memory requirements based on collected data	72
5.6.1.6.1. CPU and memory requirement for configurations	72
5.6.1.6.2. NodeAgent CPU for large usage	73
5.6.2. Installing the OADP Operator	73
5.6.2.1. Installing the OADP Operator	74
5.6.2.2. OADP-Velero-OpenShift Container Platform version relationship	74
5.7. CONFIGURING OADP WITH AWS S3 COMPATIBLE STORAGE	75
5.7.1. Configuring the OpenShift API for Data Protection with AWS S3 compatible storage	75
5.7.1.1. About Amazon Simple Storage Service, Identity and Access Management, and GovCloud	75
5.7.1.2. Configuring Amazon Web Services	76
5.7.1.3. About backup and snapshot locations and their secrets	78
5.7.1.3.1. Backup locations	78
5.7.1.3.2. Snapshot locations	79
5.7.1.3.3. Secrets	79
5.7.1.3.4. Creating a default Secret	79
5.7.1.3.5. Creating profiles for different credentials	80
5.7.1.3.6. Creating an OADP SSE-C encryption key for additional data security	81
5.7.1.3.6.1. Downloading a file with an SSE-C encryption key for files backed up by Velero	84

5.7.1.4. Installing the Data Protection Application	84
5.7.1.4.1. Setting Velero CPU and memory resource allocations	88
5.7.1.4.2. Enabling self-signed CA certificates	89
5.7.1.4.3. Using CA certificates with the velero command aliased for Velero deployment	90
5.7.1.4.4. Configuring node agents and node labels	91
5.7.1.5. Configuring the backup storage location with a MD5 checksum algorithm	91
5.7.1.6. Configuring the DPA with client burst and QPS settings	93
5.7.1.7. Configuring node agent load affinity	94
5.7.1.8. Node agent load affinity guidelines	95
5.7.1.9. Configuring node agent load concurrency	96
5.7.1.10. Configuring the node agent as a non-root and non-privileged user	97
5.7.1.11. Configuring repository maintenance	99
5.7.1.12. Configuring Velero load affinity	100
5.7.1.13. Overriding the imagePullPolicy setting in the DPA	101
5.7.1.14. Enabling CSI in the DataProtectionApplication CR	103
5.7.1.15. Disabling the node agent in DataProtectionApplication	103
5.8. CONFIGURING OADP WITH IBM CLOUD	104
5.8.1. Configuring the OpenShift API for Data Protection with IBM Cloud	104
5.8.1.1. Configuring the COS instance	104
5.8.1.2. Creating a default Secret	107
5.8.1.3. Creating secrets for different credentials	107
5.8.1.4. Installing the Data Protection Application	108
5.8.1.5. Setting Velero CPU and memory resource allocations	111
5.8.1.6. Configuring node agents and node labels	112
5.8.1.7. Configuring the DPA with client burst and QPS settings	112
5.8.1.8. Configuring node agent load affinity	114
5.8.1.9. Node agent load affinity guidelines	115
5.8.1.10. Configuring node agent load concurrency	115
5.8.1.11. Configuring repository maintenance	116
5.8.1.12. Configuring Velero load affinity	118
5.8.1.13. Overriding the imagePullPolicy setting in the DPA	119
5.8.1.14. Configuring the DPA with more than one BSL	120
5.8.1.15. Disabling the node agent in DataProtectionApplication	122
5.9. CONFIGURING OADP WITH AZURE	123
5.9.1. Configuring the OpenShift API for Data Protection with Microsoft Azure	123
5.9.1.1. Configuring Microsoft Azure	123
5.9.1.2. About backup and snapshot locations and their secrets	124
5.9.1.2.1. Backup locations	124
5.9.1.2.2. Snapshot locations	124
5.9.1.2.3. Secrets	124
5.9.1.3. About authenticating OADP with Azure	125
5.9.1.4. Using a service principal or a storage account access key	125
5.9.1.5. Using OADP with Azure Security Token Service authentication	127
5.9.1.6. Setting Velero CPU and memory resource allocations	131
5.9.1.7. Enabling self-signed CA certificates	132
5.9.1.8. Using CA certificates with the velero command aliased for Velero deployment	133
5.9.1.9. Installing the Data Protection Application	134
5.9.1.10. Configuring the DPA with client burst and QPS settings	137
5.9.1.11. Configuring node agents and node labels	139
5.9.1.12. Configuring node agent load affinity	139
5.9.1.13. Node agent load affinity guidelines	141
5.9.1.14. Configuring node agent load concurrency	141
5.9.1.15. Configuring the node agent as a non-root and non-privileged user	142

5.9.1.16. Configuring repository maintenance	144
5.9.1.17. Configuring Velero load affinity	146
5.9.1.18. Overriding the imagePullPolicy setting in the DPA	147
5.9.1.18.1. Enabling CSI in the DataProtectionApplication CR	148
5.9.1.18.2. Disabling the node agent in DataProtectionApplication	148
5.10. CONFIGURING OADP WITH GOOGLE CLOUD	149
5.10.1. Configuring the OpenShift API for Data Protection with Google Cloud	149
5.10.1.1. Configuring Google Cloud	150
5.10.1.2. About backup and snapshot locations and their secrets	151
5.10.1.2.1. Backup locations	152
5.10.1.2.2. Snapshot locations	152
5.10.1.2.3. Secrets	152
5.10.1.2.4. Creating a default Secret	152
5.10.1.2.5. Creating secrets for different credentials	153
5.10.1.3. Configuring the Data Protection Application	154
5.10.1.3.1. Setting Velero CPU and memory resource allocations	154
5.10.1.3.2. Enabling self-signed CA certificates	155
5.10.1.3.3. Using CA certificates with the velero command aliased for Velero deployment	156
5.10.1.4. Google workload identity federation cloud authentication	157
5.10.1.4.1. Google workload identity federation known issues	159
5.10.1.5. Installing the Data Protection Application	159
5.10.1.6. Configuring the DPA with client burst and QPS settings	163
5.10.1.7. Configuring node agents and node labels	164
5.10.1.8. Configuring node agent load affinity	164
5.10.1.9. Node agent load affinity guidelines	166
5.10.1.10. Configuring node agent load concurrency	166
5.10.1.11. Configuring the node agent as a non-root and non-privileged user	167
5.10.1.12. Configuring repository maintenance	170
5.10.1.13. Configuring Velero load affinity	171
5.10.1.14. Overriding the imagePullPolicy setting in the DPA	172
5.10.1.14.1. Enabling CSI in the DataProtectionApplication CR	173
5.10.1.14.2. Disabling the node agent in DataProtectionApplication	174
5.11. CONFIGURING OADP WITH MCG	175
5.11.1. Configuring the OpenShift API for Data Protection with Multicloud Object Gateway	175
5.11.1.1. Retrieving Multicloud Object Gateway credentials	175
5.11.1.2. About backup and snapshot locations and their secrets	176
5.11.1.2.1. Backup locations	176
5.11.1.2.2. Snapshot locations	176
5.11.1.2.3. Secrets	176
5.11.1.2.4. Creating a default Secret	177
5.11.1.2.5. Creating secrets for different credentials	177
5.11.1.3. Configuring the Data Protection Application	179
5.11.1.3.1. Setting Velero CPU and memory resource allocations	179
5.11.1.3.2. Enabling self-signed CA certificates	180
5.11.1.3.3. Using CA certificates with the velero command aliased for Velero deployment	180
5.11.1.4. Installing the Data Protection Application	182
5.11.1.5. Configuring the DPA with client burst and QPS settings	185
5.11.1.6. Configuring node agents and node labels	186
5.11.1.7. Configuring node agent load affinity	187
5.11.1.8. Node agent load affinity guidelines	188
5.11.1.9. Configuring node agent load concurrency	189
5.11.1.10. Configuring the node agent as a non-root and non-privileged user	190
5.11.1.11. Configuring repository maintenance	192

5.11.1.12. Configuring Velero load affinity	193
5.11.1.13. Overriding the imagePullPolicy setting in the DPA	194
5.11.1.13.1. Enabling CSI in the DataProtectionApplication CR	195
5.11.1.13.2. Disabling the node agent in DataProtectionApplication	196
5.12. CONFIGURING OADP WITH ODF	197
5.12.1. Configuring the OpenShift API for Data Protection with OpenShift Data Foundation	197
5.12.1.1. About backup and snapshot locations and their secrets	197
5.12.1.1.1. Backup locations	197
5.12.1.1.2. Snapshot locations	198
5.12.1.1.3. Secrets	198
5.12.1.1.4. Creating a default Secret	198
5.12.1.1.5. Creating secrets for different credentials	199
5.12.1.2. Configuring the Data Protection Application	200
5.12.1.2.1. Setting Velero CPU and memory resource allocations	200
5.12.1.2.1.1. Adjusting Ceph CPU and memory requirements based on collected data	201
5.12.1.2.1.1.1. CPU and memory requirement for configurations	201
5.12.1.2.2. Enabling self-signed CA certificates	202
5.12.1.2.3. Using CA certificates with the velero command aliased for Velero deployment	202
5.12.1.3. Installing the Data Protection Application	204
5.12.1.4. Configuring the DPA with client burst and QPS settings	207
5.12.1.5. Configuring node agents and node labels	208
5.12.1.6. Configuring node agent load affinity	209
5.12.1.7. Node agent load affinity guidelines	210
5.12.1.8. Configuring node agent load concurrency	211
5.12.1.9. Configuring the node agent as a non-root and non-privileged user	212
5.12.1.10. Configuring repository maintenance	214
5.12.1.11. Configuring Velero load affinity	215
5.12.1.12. Overriding the imagePullPolicy setting in the DPA	216
5.12.1.12.1. Creating an Object Bucket Claim for disaster recovery on OpenShift Data Foundation	217
5.12.1.12.2. Enabling CSI in the DataProtectionApplication CR	218
5.12.1.12.3. Disabling the node agent in DataProtectionApplication	218
5.13. CONFIGURING OADP WITH OPENSIFT VIRTUALIZATION	219
5.13.1. Configuring the OpenShift API for Data Protection with OpenShift Virtualization	220
5.13.1.1. Installing and configuring OADP with OpenShift Virtualization	220
5.13.1.2. Installing the Data Protection Application	221
5.13.1.3. Backing up a single VM	224
5.13.1.4. Restoring a single VM	225
5.13.1.5. Restoring a single VM from a backup of multiple VMs	226
5.13.1.6. Configuring the DPA with client burst and QPS settings	227
5.13.1.7. Configuring the node agent as a non-root and non-privileged user	228
5.13.1.8. Configuring node agents and node labels	231
5.13.1.9. Configuring node agent load affinity	231
5.13.1.10. Node agent load affinity guidelines	232
5.13.1.11. Configuring node agent load concurrency	233
5.13.1.12. Configuring repository maintenance	234
5.13.1.13. Configuring Velero load affinity	235
5.13.1.14. Overriding the imagePullPolicy setting in the DPA	236
5.13.1.15. About incremental back up support	238
5.14. CONFIGURING OADP WITH MULTIPLE BACKUP STORAGE LOCATIONS	238
5.14.1. Configuring the OpenShift API for Data Protection (OADP) with more than one Backup Storage Location	239
5.14.1.1. Configuring the DPA with more than one BSL	239
5.14.1.2. Configuring two backup BSLs with different cloud credentials	241

5.15. CONFIGURING OADP WITH MULTIPLE VOLUME SNAPSHOT LOCATIONS	245
5.15.1. Configuring the OpenShift API for Data Protection (OADP) with more than one Volume Snapshot Location	245
5.15.1.1. Configuring the DPA with more than one VSL	245
5.16. UNINSTALLING OADP	246
5.16.1. Uninstalling the OpenShift API for Data Protection	246
5.17. OADP BACKING UP	246
5.17.1. Backing up applications	246
5.17.1.1. Previewing resources before running backup and restore	247
5.17.1.2. Known issues	248
5.17.2. Creating a Backup CR	248
5.17.3. Backing up persistent volumes with CSI snapshots	250
5.17.3.1. Backing up persistent volumes with CSI snapshots	250
5.17.4. Backing up applications with File System Backup: Kopia or Restic	251
5.17.4.1. Backing up applications with File System Backup	252
5.17.5. Creating backup hooks	252
5.17.6. Scheduling backups using Schedule CR	254
5.17.7. Deleting backups	255
5.17.7.1. Deleting a backup by creating a DeleteBackupRequest CR	256
5.17.7.2. Deleting a backup by using the Velero CLI	256
5.17.7.3. About Kopia repository maintenance	256
5.17.7.3.1. Kopia maintenance in OADP	257
5.17.7.4. Deleting a backup repository	257
5.17.8. About Kopia	258
5.17.8.1. OADP integration with Kopia	258
5.18. OADP RESTORING	259
5.18.1. Restoring applications	259
5.18.1.1. Previewing resources before running backup and restore	259
5.18.1.2. Creating a Restore CR	260
5.18.1.3. Creating restore hooks	262
5.19. OADP SELF-SERVICE	264
5.19.1. OADP Self-Service	265
5.19.1.1. About OADP Self-Service	265
5.19.1.2. What namespace-scoped backup and restore means	265
5.19.1.3. OADP Self-Service custom resources	266
5.19.1.4. How OADP Self-Service works	266
5.19.1.5. OADP Self-Service prerequisites	267
5.19.1.6. OADP Self-Service namespace permissions	267
5.19.1.6.1. Example RBAC YAML for backup operation	268
5.19.1.7. OADP Self-Service limitations	268
5.19.1.8. OADP Self-Service backup and restore phases	269
5.19.1.9. About NonAdminBackupStorageLocation CR	270
5.19.2. OADP Self-Service cluster admin use cases	271
5.19.2.1. Enabling and disabling OADP Self-Service	271
5.19.2.2. Enabling NonAdminBackupStorageLocation administrator approval workflow	272
5.19.2.3. Approving a NonAdminBackupStorageLocation request	273
5.19.2.4. Rejecting a NonAdminBackupStorageLocation request	275
5.19.2.5. OADP Self-Service administrator DPA spec enforcement	275
5.19.2.6. Self-Service administrator spec enforcement for NABSL	276
5.19.2.7. Self-Service administrator spec enforcement for NAB	277
5.19.2.8. Self-Service administrator spec enforcement for NAR	278
5.19.3. OADP Self-Service namespace admin use cases	278
5.19.3.1. Creating a NonAdminBackupStorageLocation CR	279

5.19.3.2. Creating a NonAdminBackup CR	282
5.19.3.3. Deleting a NonAdminBackup CR	286
5.19.3.4. Creating a NonAdminRestore CR	287
5.19.3.5. About NonAdminDownloadRequest CR	289
5.19.3.6. Reviewing NAB and NAR logs	290
5.19.4. OADP Self-Service troubleshooting	293
5.19.4.1. Resolving error NonAdminBackupStorageLocation not found in the namespace	293
5.19.4.2. Resolving error NonAdminBackupStorageLocation cannot be set as default	294
5.20. OADP AND ROSA	294
5.20.1. Backing up applications on ROSA clusters using OADP	294
5.20.1.1. Preparing AWS credentials for OADP	295
5.20.1.2. Installing the OADP Operator and providing the IAM role	298
5.20.1.3. Updating the IAM role ARN in the OADP Operator subscription	303
5.20.1.4. Example: Backing up workload on OADP ROSA STS, with an optional cleanup	306
5.20.1.4.1. Performing a backup with OADP and OpenShift Container Platform	306
5.20.1.4.2. Cleaning up a cluster after a backup with OADP and ROSA STS	308
5.21. OADP AND AWS STS	310
5.21.1. Backing up applications on AWS STS using OADP	310
5.21.1.1. Preparing AWS STS credentials for OADP	310
5.21.1.1.1. Setting Velero CPU and memory resource allocations	313
5.21.1.2. Installing the OADP Operator and providing the IAM role	314
5.21.1.3. Backing up workload on OADP AWS STS, with an optional cleanup	320
5.21.1.3.1. Performing a backup with OADP and AWS STS	320
5.21.1.3.2. Cleaning up a cluster after a backup with OADP and AWS STS	322
5.22. OADP AND 3SCALE	323
5.22.1. Backing up and restoring 3scale API Management by using OADP	323
5.22.2. Backing up 3scale API Management by using OADP	323
5.22.2.1. Creating the Data Protection Application	324
5.22.2.2. Backing up the 3scale API Management operator, secret, and APIManager	325
5.22.2.3. Backing up a MySQL database	327
5.22.2.4. Backing up the back-end Redis database	330
5.22.3. Restoring 3scale API Management by using OADP	331
5.22.3.1. Restoring the 3scale API Management operator, secrets, and APIManager	332
5.22.3.2. Restoring a MySQL database	335
5.22.3.3. Restoring the back-end Redis database	338
5.22.3.4. Scaling up the 3scale API Management operator and deployment	339
5.23. OADP DATA MOVER	341
5.23.1. About the OADP Data Mover	341
5.23.1.1. Data Mover support	342
5.23.1.2. Enabling the built-in Data Mover	342
5.23.1.3. Built-in Data Mover controller and custom resource definitions (CRDs)	343
5.23.1.4. About incremental back up support	343
5.23.2. Backing up and restoring CSI snapshots data movement	344
5.23.2.1. Backing up persistent volumes with CSI snapshots	344
5.23.2.2. Restoring CSI volume snapshots	346
5.23.2.3. Deletion policy for OADP 1.3	348
5.23.2.3.1. Deletion policy guidelines for OADP 1.3	348
5.23.3. Configuring backup and restore PVCs for Data Mover	348
5.23.3.1. Configuring a backup PVC for a Data Mover backup	349
5.23.3.2. Configuring a restorePVC for a Data Mover restore	350
5.23.4. Overriding Kopio hashing, encryption, and splitter algorithms	351
5.23.4.1. Configuring the DPA to override Kopio hashing, encryption, and splitter algorithms	351
5.23.4.2. Use case for overriding Kopio hashing, encryption, and splitter algorithms	353

5.23.4.3. Benchmarking Kopia hashing, encryption, and splitter algorithms	356
5.24. APIS USED WITH OADP	360
5.24.1. DataProtectionApplicationSpec type	360
5.24.2. BackupLocation type	361
5.24.3. SnapshotLocation type	362
5.24.4. ApplicationConfig type	362
5.24.5. VeleroConfig type	363
5.24.6. CustomPlugin type	364
5.24.7. ResticConfig type	364
5.24.8. PodConfig type	365
5.24.9. Features type	366
5.24.10. DataMover type	366
5.25. ADVANCED OADP FEATURES AND FUNCTIONALITIES	366
5.25.1. Working with different Kubernetes API versions on the same cluster	366
5.25.1.1. Listing the Kubernetes API group versions on a cluster	367
5.25.1.2. About Enable API Group Versions	367
5.25.1.3. Using Enable API Group Versions	368
5.25.2. Backing up data from one cluster and restoring it to another cluster	368
5.25.2.1. About backing up data from one cluster and restoring it on another cluster	368
5.25.2.1.1. Operators	368
5.25.2.1.2. Use of Velero	369
5.25.2.2. About determining which pod volumes to back up	369
5.25.2.2.1. Limitations	369
5.25.2.2.2. Backing up pod volumes by using the opt-in method	370
5.25.2.2.3. Backing up pod volumes by using the opt-out method	370
5.25.2.3. UID and GID ranges	371
5.25.2.4. Backing up data from one cluster and restoring it to another cluster	372
5.25.3. OADP storage class mapping	373
5.25.3.1. Storage class mapping	373
5.25.3.1.1. Storage class mapping with Migration Toolkit for Containers	373
5.25.3.1.2. Mapping storage classes with OADP	373
5.25.4. Additional resources	374
5.26. OADP TROUBLESHOOTING	374
5.26.1. Troubleshooting	374
5.26.2. Velero CLI tool	374
5.26.2.1. Downloading the Velero CLI tool	375
5.26.2.1.1. OADP-Velero-OpenShift Container Platform version relationship	375
5.26.2.2. Accessing the Velero binary in the Velero deployment in the cluster	376
5.26.2.3. Debugging Velero resources with the OpenShift CLI tool	376
5.26.2.4. Debugging Velero resources with the Velero CLI tool	377
5.26.3. Pods crash or restart due to lack of memory or CPU	378
5.26.3.1. Setting resource requests for a Velero pod	379
5.26.3.2. Setting resource requests for a Restic pod	379
5.26.3.3. Setting resource requests for a nodeAgent pod	380
5.26.4. Restoring workarounds for Velero backups that use admission webhooks	381
5.26.4.1. Restoring Knative resources	382
5.26.4.2. Restoring IBM AppConnect resources	382
5.26.4.3. Avoiding the Velero plugin panic error	382
5.26.4.4. Workaround for OpenShift ADP Controller segmentation fault	383
5.26.5. OADP installation issues	384
5.26.5.1. Resolving invalid directories in backup storage	384
5.26.5.2. Resolving incorrect AWS credentials	384
5.26.6. OADP Operator issues	385

5.26.6.1. Resolving silent failure of the OADP Operator	385
5.26.7. OADP timeouts	386
5.26.7.1. Implementing restic timeout	386
5.26.7.2. Implementing velero resource timeout	387
5.26.7.2.1. Implementing velero default item operation timeout	388
5.26.7.3. Implementing Data Mover timeout	388
5.26.7.4. Implementing CSI snapshot timeout	389
5.26.7.5. Implementing item operation timeout - restore	389
5.26.7.6. Implementing item operation timeout - backup	390
5.26.8. Backup and Restore CR issues	390
5.26.8.1. Troubleshooting issue where backup CR cannot retrieve volume	390
5.26.8.2. Troubleshooting issue where backup CR status remains in progress	391
5.26.8.3. Troubleshooting issue where backup CR status remains partially failed	391
5.26.9. Restic issues	392
5.26.9.1. Troubleshooting Restic permission errors for NFS data volumes	392
5.26.9.2. Troubleshooting Restic Backup CR issue that cannot be re-created after bucket is emptied	393
5.26.9.3. Troubleshooting restic restore partially failed issue on OpenShift Container Platform 4.14 due to changed PSA policy	393
5.26.10. OADP Data protection test	395
5.26.10.1. OADP DataProtectionTest CR specification fields	395
5.26.10.2. OADP DataProtectionTest CR status fields	396
5.26.10.3. Using the DataProtectionTest custom resource	397
5.26.10.4. Running a data protection test by configuring a backup storage location specification	399
5.26.10.5. Running a data protection test on an Azure object storage	401
5.26.10.6. Troubleshooting the DataProtectionTest custom resource	403
5.26.11. Using the must-gather tool	403
5.26.11.1. Using the must-gather tool	404
5.26.12. OADP monitoring	405
5.26.12.1. OADP monitoring setup	406
5.26.12.2. Creating OADP service monitor	407
5.26.12.3. Creating an alerting rule	409
5.26.12.4. List of available metrics	410
5.26.12.5. Viewing metrics using the Observe UI	413
CHAPTER 6. CONTROL PLANE BACKUP AND RESTORE	414
6.1. BACKING UP ETCD	414
6.1.1. Backing up etcd data	414
6.1.2. Additional resources	416
6.1.3. Creating automated etcd backups	416
6.1.3.1. Creating a single automated etcd backup	417
6.1.3.2. Creating recurring automated etcd backups	420
6.2. REPLACING AN UNHEALTHY ETCD MEMBER	425
6.2.1. Prerequisites	425
6.2.2. Identifying an unhealthy etcd member	425
6.2.3. Determining the state of the unhealthy etcd member	426
6.2.4. Replacing the unhealthy etcd member	428
6.2.4.1. Replacing an unhealthy etcd member whose machine is not running or whose node is not ready	428
6.2.4.2. Replacing an unhealthy etcd member whose etcd pod is crashlooping	438
6.2.4.3. Replacing an unhealthy bare metal etcd member whose machine is not running or whose node is not ready	442
6.2.5. Additional resources	451
6.3. DISASTER RECOVERY	452
6.3.1. About disaster recovery	452

6.3.1.1. Testing restore procedures	452
6.3.2. Quorum restoration	454
6.3.2.1. Restoring etcd quorum for high availability clusters	454
6.3.2.2. Additional resources	457
6.3.3. Restoring to a previous cluster state	457
6.3.3.1. About restoring to a previous cluster state	458
6.3.3.2. Restoring to a previous cluster state for a single node	458
6.3.3.3. Restoring to a previous cluster state for more than one node	459
6.3.3.4. Restoring a cluster manually from an etcd backup	461
6.3.3.5. Additional resources	468
6.3.3.6. Issues and workarounds for restoring a persistent storage state	468
6.3.4. Recovering from expired control plane certificates	469
6.3.4.1. Recovering from expired control plane certificates	469

CHAPTER 1. BACKUP AND RESTORE

1.1. CONTROL PLANE BACKUP AND RESTORE OPERATIONS

As a cluster administrator, you might need to stop an OpenShift Container Platform cluster for a period and restart it later. Some reasons for restarting a cluster are that you need to perform maintenance on a cluster or want to reduce resource costs. In OpenShift Container Platform, you can perform a [graceful shutdown of a cluster](#) so that you can easily restart the cluster later.

You must [back up etcd data](#) before shutting down a cluster; etcd is the key-value store for OpenShift Container Platform, which persists the state of all resource objects. An etcd backup plays a crucial role in disaster recovery. In OpenShift Container Platform, you can also [replace an unhealthy etcd member](#).

When you want to get your cluster running again, [restart the cluster gracefully](#).



NOTE

A cluster's certificates expire one year after the installation date. You can shut down a cluster and expect it to restart gracefully while the certificates are still valid. Although the cluster automatically retrieves the expired control plane certificates, you must still [approve the certificate signing requests \(CSRs\)](#).

You might run into several situations where OpenShift Container Platform does not work as expected, such as:

- You have a cluster that is not functional after the restart because of unexpected conditions, such as node failure or network connectivity issues.
- You have deleted something critical in the cluster by mistake.
- You have lost the majority of your control plane hosts, leading to etcd quorum loss.

You can always recover from a disaster situation by [restoring your cluster to its previous state](#) using the saved etcd snapshots.

Additional resources

- [Quorum protection with machine lifecycle hooks](#)

1.2. APPLICATION BACKUP AND RESTORE OPERATIONS

As a cluster administrator, you can back up and restore applications running on OpenShift Container Platform by using the OpenShift API for Data Protection (OADP).

OADP backs up and restores Kubernetes resources and internal images, at the granularity of a namespace, by using the version of Velero that is appropriate for the version of OADP you install, according to the table in [Downloading the Velero CLI tool](#). OADP backs up and restores persistent volumes (PVs) by using snapshots or Restic. For details, see [OADP features](#).

1.2.1. OADP requirements

OADP has the following requirements:

- You must be logged in as a user with a **cluster-admin** role.

- You must have object storage for storing backups, such as one of the following storage types:
 - OpenShift Data Foundation
 - Amazon Web Services
 - Microsoft Azure
 - Google Cloud
 - S3-compatible object storage
 - IBM Cloud® Object Storage S3



NOTE

If you want to use CSI backup on OCP 4.11 and later, install OADP 1.1.x.

OADP 1.0.x does not support CSI backup on OCP 4.11 and later. OADP 1.0. x includes Velero 1.7.x and expects the API group **snapshot.storage.k8s.io/v1beta1**, which is not present on OCP 4.11 and later.

- To back up PVs with snapshots, you must have cloud storage that has a native snapshot API or supports Container Storage Interface (CSI) snapshots, such as the following providers:
 - Amazon Web Services
 - Microsoft Azure
 - Google Cloud
 - CSI snapshot-enabled cloud storage, such as Ceph RBD or Ceph FS



NOTE

If you do not want to back up PVs by using snapshots, you can use [Restic](#), which is installed by the OADP Operator by default.

1.2.2. Backing up and restoring applications

You back up applications by creating a **Backup** custom resource (CR). See [Creating a Backup CR](#). You can configure the following backup options:

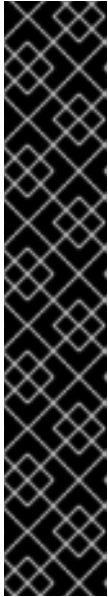
- [Creating backup hooks](#) to run commands before or after the backup operation
- [Scheduling backups](#)
- [Backing up applications with File System Backup: Kopia or Restic](#)
- You restore application backups by creating a **Restore** (CR). See [Creating a Restore CR](#).
- You can configure [restore hooks](#) to run commands in init containers or in the application container during the restore operation.

CHAPTER 2. SHUTTING DOWN THE CLUSTER GRACEFULLY

This document describes the process to gracefully shut down your cluster. You might need to temporarily shut down your cluster for maintenance reasons, or to save on resource costs.

2.1. PREREQUISITES

- Take an [etcd backup](#) prior to shutting down the cluster.



IMPORTANT

It is important to take an etcd backup before performing this procedure so that your cluster can be restored if you encounter any issues when restarting the cluster.

For example, the following conditions can cause the restarted cluster to malfunction:

- etcd data corruption during shutdown
- Node failure due to hardware
- Network connectivity issues

If your cluster fails to recover, follow the steps to [restore to a previous cluster state](#).

2.2. SHUTTING DOWN THE CLUSTER

You can shut down your cluster in a graceful manner so that it can be restarted at a later date.



NOTE

You can shut down a cluster until a year from the installation date and expect it to restart gracefully. After a year from the installation date, the cluster certificates expire. However, you might need to manually approve the pending certificate signing requests (CSRs) to recover kubelet certificates when the cluster restarts.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have taken an etcd backup.
- If you are running a single-node OpenShift cluster, you must evacuate all workload pods off of the cluster before you shut it down.

Procedure

1. If you are shutting the cluster down for an extended period, determine the date on which certificates expire and run the following command:

```
$ oc -n openshift-kube-apiserver-operator get secret kube-apiserver-to-kubelet-signer -o jsonpath='{.metadata.annotations.auth\.openshift\.io/certificate-not-after}'
```

Example output

```
2022-08-05T14:37:50Zuser@user:~ $ 1
```

- 1 To ensure that the cluster can restart gracefully, plan to restart it on or before the specified date. As the cluster restarts, the process might require you to manually approve the pending certificate signing requests (CSRs) to recover kubelet certificates.

2. Mark all the nodes in the cluster as unschedulable. You can do this from your cloud provider's web console, or by running the following loop:

```
$ for node in $(oc get nodes -o jsonpath='{.items[*].metadata.name}'); do echo ${node} ; oc adm cordon ${node} ; done
```

Example output

```
ci-ln-mgdnf4b-72292-n547t-master-0
node/ci-ln-mgdnf4b-72292-n547t-master-0 cordoned
ci-ln-mgdnf4b-72292-n547t-master-1
node/ci-ln-mgdnf4b-72292-n547t-master-1 cordoned
ci-ln-mgdnf4b-72292-n547t-master-2
node/ci-ln-mgdnf4b-72292-n547t-master-2 cordoned
ci-ln-mgdnf4b-72292-n547t-worker-a-s7ntl
node/ci-ln-mgdnf4b-72292-n547t-worker-a-s7ntl cordoned
ci-ln-mgdnf4b-72292-n547t-worker-b-cmc9k
node/ci-ln-mgdnf4b-72292-n547t-worker-b-cmc9k cordoned
ci-ln-mgdnf4b-72292-n547t-worker-c-vcmtn
node/ci-ln-mgdnf4b-72292-n547t-worker-c-vcmtn cordoned
```

3. Evacuate the pods using the following method:

```
$ for node in $(oc get nodes -l node-role.kubernetes.io/worker -o jsonpath='{.items[*].metadata.name}'); do echo ${node} ; oc adm drain ${node} --delete-emptydir-data --ignore-daemonsets=true --timeout=15s --force ; done
```

4. Shut down all of the nodes in the cluster. You can do this from the web console for your cloud provider web console, or by running the following loop. Shutting down the nodes by using one of these methods allows pods to terminate gracefully, which reduces the chance for data corruption.



NOTE

Ensure that the control plane node with the API VIP assigned is the last node processed in the loop. Otherwise, the shutdown command fails.

```
$ for node in $(oc get nodes -o jsonpath='{.items[*].metadata.name}'); do oc debug node/${node} -- chroot /host shutdown -h 1 ; done 1
```

- 1 **-h 1** indicates how long, in minutes, this process lasts before the control plane nodes are shut down. For large-scale clusters with 10 nodes or more, set to **-h 10** or longer to make sure all the compute nodes have time to shut down first.

Example output

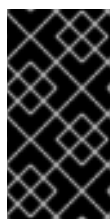
```
Starting pod/ip-10-0-130-169us-east-2computeinternal-debug ...
To use host binaries, run `chroot /host`
Shutdown scheduled for Mon 2021-09-13 09:36:17 UTC, use 'shutdown -c' to cancel.
Removing debug pod ...
Starting pod/ip-10-0-150-116us-east-2computeinternal-debug ...
To use host binaries, run `chroot /host`
Shutdown scheduled for Mon 2021-09-13 09:36:29 UTC, use 'shutdown -c' to cancel.
```



NOTE

It is not necessary to drain control plane nodes of the standard pods that ship with OpenShift Container Platform prior to shutdown. Cluster administrators are responsible for ensuring a clean restart of their own workloads after the cluster is restarted. If you drained control plane nodes prior to shutdown because of custom workloads, you must mark the control plane nodes as schedulable before the cluster will be functional again after restart.

5. Shut off any cluster dependencies that are no longer needed, such as external storage or an LDAP server. Be sure to consult your vendor's documentation before doing so.



IMPORTANT

If you deployed your cluster on a cloud-provider platform, do not shut down, suspend, or delete the associated cloud resources. If you delete the cloud resources of a suspended virtual machine, OpenShift Container Platform might not restore successfully.

2.3. ADDITIONAL RESOURCES

- [Restarting the cluster gracefully](#)

CHAPTER 3. RESTARTING THE CLUSTER GRACEFULLY

This document describes the process to restart your cluster after a graceful shutdown.

Even though the cluster is expected to be functional after the restart, the cluster might not recover due to unexpected conditions, for example:

- etcd data corruption during shutdown
- Node failure due to hardware
- Network connectivity issues

If your cluster fails to recover, follow the steps to [restore to a previous cluster state](#).

3.1. PREREQUISITES

- You have [gracefully shut down your cluster](#).

3.2. RESTARTING THE CLUSTER

You can restart your cluster after it has been shut down gracefully.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- This procedure assumes that you gracefully shut down the cluster.

Procedure

1. Turn on the control plane nodes.
 - If you are using the **admin.kubeconfig** from the cluster installation and the API virtual IP address (VIP) is up, complete the following steps:
 - a. Set the **KUBECONFIG** environment variable to the **admin.kubeconfig** path.
 - b. For each control plane node in the cluster, run the following command:

```
$ oc adm uncordon <node>
```

- If you do not have access to your **admin.kubeconfig** credentials, complete the following steps:
 - a. Use SSH to connect to a control plane node.
 - b. Copy the **localhost-recovery.kubeconfig** file to the **/root** directory.
 - c. Use that file to run the following command for each control plane node in the cluster:

```
$ oc adm uncordon <node>
```

2. Power on any cluster dependencies, such as external storage or an LDAP server.

3. Start all cluster machines.

Use the appropriate method for your cloud environment to start the machines, for example, from your cloud provider's web console.

Wait approximately 10 minutes before continuing to check the status of control plane nodes.

4. Verify that all control plane nodes are ready.

```
$ oc get nodes -l node-role.kubernetes.io/master
```

The control plane nodes are ready if the status is **Ready**, as shown in the following output:

NAME	STATUS	ROLES	AGE	VERSION
ip-10-0-168-251.ec2.internal	Ready	control-plane,master	75m	v1.33.4
ip-10-0-170-223.ec2.internal	Ready	control-plane,master	75m	v1.33.4
ip-10-0-211-16.ec2.internal	Ready	control-plane,master	75m	v1.33.4

5. If the control plane nodes are *not* ready, then check whether there are any pending certificate signing requests (CSRs) that must be approved.

- a. Get the list of current CSRs:

```
$ oc get csr
```

- b. Review the details of a CSR to verify that it is valid:

```
$ oc describe csr <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- c. Approve each valid CSR:

```
$ oc adm certificate approve <csr_name>
```

6. After the control plane nodes are ready, verify that all worker nodes are ready.

```
$ oc get nodes -l node-role.kubernetes.io/worker
```

The worker nodes are ready if the status is **Ready**, as shown in the following output:

NAME	STATUS	ROLES	AGE	VERSION
ip-10-0-179-95.ec2.internal	Ready	worker	64m	v1.33.4
ip-10-0-182-134.ec2.internal	Ready	worker	64m	v1.33.4
ip-10-0-250-100.ec2.internal	Ready	worker	64m	v1.33.4

7. If the worker nodes are *not* ready, then check whether there are any pending certificate signing requests (CSRs) that must be approved.

- a. Get the list of current CSRs:

```
$ oc get csr
```

- b. Review the details of a CSR to verify that it is valid:

```
$ oc describe csr <csr_name> 1
```

1 **<csr_name>** is the name of a CSR from the list of current CSRs.

- c. Approve each valid CSR:

```
$ oc adm certificate approve <csr_name>
```

8. After the control plane and compute nodes are ready, mark all the nodes in the cluster as schedulable by running the following command:

```
$ for node in $(oc get nodes -o jsonpath='{.items[*].metadata.name}'); do echo ${node} ; oc adm uncordon ${node} ; done
```

9. Verify that the cluster started properly.

- a. Check that there are no degraded cluster Operators.

```
$ oc get clusteroperators
```

Check that there are no cluster Operators with the **DEGRADED** condition set to **True**.

NAME SINCE	VERSION	AVAILABLE	PROGRESSING	DEGRADED
authentication	4.20.0	True	False	False 59m
cloud-credential	4.20.0	True	False	False 85m
cluster-autoscaler	4.20.0	True	False	False 73m
config-operator	4.20.0	True	False	False 73m
console	4.20.0	True	False	False 62m
csi-snapshot-controller	4.20.0	True	False	False 66m
dns	4.20.0	True	False	False 76m
etcd	4.20.0	True	False	False 76m
...				

- b. Check that all nodes are in the **Ready** state:

```
$ oc get nodes
```

Check that the status for all nodes is **Ready**.

NAME	STATUS	ROLES	AGE	VERSION
ip-10-0-168-251.ec2.internal	Ready	control-plane,master	82m	v1.33.4
ip-10-0-170-223.ec2.internal	Ready	control-plane,master	82m	v1.33.4
ip-10-0-179-95.ec2.internal	Ready	worker	70m	v1.33.4
ip-10-0-182-134.ec2.internal	Ready	worker	70m	v1.33.4
ip-10-0-211-16.ec2.internal	Ready	control-plane,master	82m	v1.33.4
ip-10-0-250-100.ec2.internal	Ready	worker	69m	v1.33.4

If the cluster did not start properly, you might need to restore your cluster using an etcd backup. For more information, see "Restoring to a previous cluster state".

Additional resources

- See [Restoring to a previous cluster state](#) for how to use an etcd backup to restore if your cluster failed to recover after restarting.

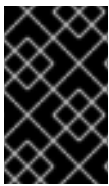
CHAPTER 4. HIBERNATING AN OPENSIFT CONTAINER PLATFORM CLUSTER

You can hibernate your OpenShift Container Platform cluster for up to 90 days.

4.1. ABOUT CLUSTER HIBERNATION

OpenShift Container Platform clusters can be hibernated in order to save money on cloud hosting costs. You can hibernate your OpenShift Container Platform cluster for up to 90 days and expect it to resume successfully.

You must wait at least 24 hours after cluster installation before hibernating your cluster to allow for the first certification rotation.



IMPORTANT

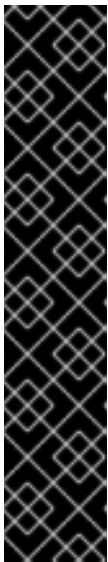
If you must hibernate your cluster before the 24 hour certificate rotation, use the following procedure instead: [Enabling OpenShift 4 Clusters to Stop and Resume Cluster VMs](#).

When hibernating a cluster, you must hibernate all cluster nodes. It is not supported to suspend only certain nodes.

After resuming, it can take up to 45 minutes for the cluster to become ready.

4.2. PREREQUISITES

- Take an [etcd backup](#) prior to hibernating the cluster.



IMPORTANT

It is important to take an etcd backup before hibernating so that your cluster can be restored if you encounter any issues when resuming the cluster.

For example, the following conditions can cause the resumed cluster to malfunction:

- etcd data corruption during hibernation
- Node failure due to hardware
- Network connectivity issues

If your cluster fails to recover, follow the steps to [restore to a previous cluster state](#).

4.3. HIBERNATING A CLUSTER

You can hibernate a cluster for up to 90 days. The cluster can recover if certificates expire while the cluster was in hibernation.

Prerequisites

- The cluster has been running for at least 24 hours to allow the first certificate rotation to complete.



IMPORTANT

If you must hibernate your cluster before the 24 hour certificate rotation, use the following procedure instead: [Enabling OpenShift 4 Clusters to Stop and Resume Cluster VMs](#).

- You have taken an etcd backup.
- You have access to the cluster as a user with the **cluster-admin** role.

Procedure

1. Confirm that your cluster has been installed for at least 24 hours.
2. Ensure that all nodes are in a good state by running the following command:

```
$ oc get nodes
```

Example output

NAME	STATUS	ROLES	AGE	VERSION
ci-ln-812tb4k-72292-8bcj7-master-0	Ready	control-plane,master	32m	v1.33.4
ci-ln-812tb4k-72292-8bcj7-master-1	Ready	control-plane,master	32m	v1.33.4
ci-ln-812tb4k-72292-8bcj7-master-2	Ready	control-plane,master	32m	v1.33.4
Ci-ln-812tb4k-72292-8bcj7-worker-a-zhdvk	Ready	worker	19m	v1.33.4
ci-ln-812tb4k-72292-8bcj7-worker-b-9hrmv	Ready	worker	19m	v1.33.4
ci-ln-812tb4k-72292-8bcj7-worker-c-q8mw2	Ready	worker	19m	v1.33.4

All nodes should show **Ready** in the **STATUS** column.

3. Ensure that all cluster Operators are in a good state by running the following command:

```
$ oc get clusteroperators
```

Example output

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.20.0-0	True	False	False	51m
baremetal	4.20.0-0	True	False	False	72m
cloud-controller-manager	4.20.0-0	True	False	False	75m
cloud-credential	4.20.0-0	True	False	False	77m
cluster-api	4.20.0-0	True	False	False	42m
cluster-autoscaler	4.20.0-0	True	False	False	72m
config-operator	4.20.0-0	True	False	False	72m
console	4.20.0-0	True	False	False	55m
...					

All cluster Operators should show **AVAILABLE=True**, **PROGRESSING=False**, and **DEGRADED=False**.

4. Ensure that all machine config pools are in a good state by running the following command:

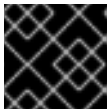
```
$ oc get mcp
```

Example output

```
NAME CONFIG UPDATED UPDATING DEGRADED
MACHINECOUNT READYMACHINECOUNT UPDATEDMACHINECOUNT
DEGRADEDMACHINECOUNT AGE
master rendered-master-87871f187930e67233c837e1d07f49c7 True False False 3
3 3 0 96m
worker rendered-worker-3c4c459dc5d90017983d7e72928b8aed True False False 3
3 3 0 96m
```

All machine config pools should show **UPDATING=False** and **DEGRADED=False**.

5. Stop the cluster virtual machines:
Use the tools native to your cluster's cloud environment to shut down the cluster's virtual machines.



IMPORTANT

If you use a bastion virtual machine, do not shut down this virtual machine.

Additional resources

- [Backing up etcd](#)

4.4. RESUMING A HIBERNATED CLUSTER

When you resume a hibernated cluster within 90 days, you might have to approve certificate signing requests (CSRs) for the nodes to become ready.

It can take around 45 minutes for the cluster to resume, depending on the size of your cluster.

Prerequisites

- You hibernated your cluster less than 90 days ago.
- You have access to the cluster as a user with the **cluster-admin** role.

Procedure

1. Within 90 days of cluster hibernation, resume the cluster virtual machines:
Use the tools native to your cluster's cloud environment to resume the cluster's virtual machines.
2. Wait about 5 minutes, depending on the number of nodes in your cluster.
3. Approve CSRs for the nodes:
 - a. Check that there is a CSR for each node in the **NotReady** state:

```
$ oc get csr
```

Example output

NAME	AGE	SIGNERNAME	REQUESTOR
csr-4dwsd	37m	kubernetes.io/kube-apiserver-client	system:node:ci-ln-812tb4k-72292-8bcj7-worker-c-q8mw2
	24h		Pending
csr-4vrbr	49m	kubernetes.io/kube-apiserver-client	system:node:ci-ln-812tb4k-72292-8bcj7-master-1
	24h		Pending
csr-4wk5x	51m	kubernetes.io/kubelet-serving	system:node:ci-ln-812tb4k-72292-8bcj7-master-1
	<none>		Pending
csr-84vb6	51m	kubernetes.io/kube-apiserver-client-kubelet	system:serviceaccount:openshift-machine-config-operator:node-bootstrapper
	<none>		Pending

- b. Approve each valid CSR by running the following command:

```
$ oc adm certificate approve <csr_name>
```

- c. Verify that all necessary CSRs were approved by running the following command:

```
$ oc get csr
```

Example output

NAME	AGE	SIGNERNAME	REQUESTOR
csr-4dwsd	37m	kubernetes.io/kube-apiserver-client	system:node:ci-ln-812tb4k-72292-8bcj7-worker-c-q8mw2
	24h		Approved,Issued
csr-4vrbr	49m	kubernetes.io/kube-apiserver-client	system:node:ci-ln-812tb4k-72292-8bcj7-master-1
	24h		Approved,Issued
csr-4wk5x	51m	kubernetes.io/kubelet-serving	system:node:ci-ln-812tb4k-72292-8bcj7-master-1
	<none>		Approved,Issued
csr-84vb6	51m	kubernetes.io/kube-apiserver-client-kubelet	system:serviceaccount:openshift-machine-config-operator:node-bootstrapper
	<none>		Approved,Issued

CSRs should show **Approved,Issued** in the **CONDITION** column.

4. Verify that all nodes now show as ready by running the following command:

```
$ oc get nodes
```

Example output

NAME	STATUS	ROLES	AGE	VERSION
ci-ln-812tb4k-72292-8bcj7-master-0	Ready	control-plane,master	32m	v1.33.4
ci-ln-812tb4k-72292-8bcj7-master-1	Ready	control-plane,master	32m	v1.33.4
ci-ln-812tb4k-72292-8bcj7-master-2	Ready	control-plane,master	32m	v1.33.4
ci-ln-812tb4k-72292-8bcj7-worker-a-zhdvk	Ready	worker	19m	v1.33.4
ci-ln-812tb4k-72292-8bcj7-worker-b-9hrmv	Ready	worker	19m	v1.33.4
ci-ln-812tb4k-72292-8bcj7-worker-c-q8mw2	Ready	worker	19m	v1.33.4

All nodes should show **Ready** in the **STATUS** column. It might take a few minutes for all nodes to become ready after approving the CSRs.

5. Wait for cluster Operators to restart to load the new certificates.
This might take 5 or 10 minutes.
6. Verify that all cluster Operators are in a good state by running the following command:

```
$ oc get clusteroperators
```

Example output

```
NAME                VERSION AVAILABLE PROGRESSING DEGRADED SINCE
MESSAGE
authentication      4.20.0-0 True      False      False      51m
baremetal            4.20.0-0 True      False      False      72m
cloud-controller-manager 4.20.0-0 True      False      False      75m
cloud-credential     4.20.0-0 True      False      False      77m
cluster-api          4.20.0-0 True      False      False      42m
cluster-autoscaler   4.20.0-0 True      False      False      72m
config-operator      4.20.0-0 True      False      False      72m
console              4.20.0-0 True      False      False      55m
...
```

All cluster Operators should show **AVAILABLE=True**, **PROGRESSING=False**, and **DEGRADED=False**.

CHAPTER 5. OADP APPLICATION BACKUP AND RESTORE

5.1. INTRODUCTION TO OPENSIFT API FOR DATA PROTECTION

The OpenShift API for Data Protection (OADP) product safeguards customer applications on OpenShift Container Platform. It offers comprehensive disaster recovery protection, covering OpenShift Container Platform applications, application-related cluster resources, persistent volumes, and internal images. OADP is also capable of backing up both containerized applications and virtual machines (VMs).

However, OADP does not serve as a disaster recovery solution for [etcd](#) or OpenShift Operators.



IMPORTANT

OADP support is applicable to customer workload namespaces and cluster scope resources.

Full cluster [backup](#) and [restore](#) are not supported.

5.1.1. OpenShift API for Data Protection APIs

OADP provides APIs that enable multiple approaches to customizing backups and preventing the inclusion of unnecessary or inappropriate resources.

OADP provides the following APIs:

- [Backup](#)
- [Restore](#)
- [Schedule](#)
- [BackupStorageLocation](#)
- [VolumeSnapshotLocation](#)

5.1.1.1. Support for OpenShift API for Data Protection

Table 5.1. Supported versions of OADP

Version	OCP version	General availability	Full support ends	Maintenance ends	Extended Update Support (EUS)	Extended Update Support Term 2 (EUS Term 2)
1.5	4.19	17 June 2025	Release of 1.6	Release of 1.7	EUS must be on OCP 4.20	EUS Term 2 must be on OCP 4.20

1.4	4 . 1 4 4 . 1 5 4 . 1 6 4 . 1 7 4 . 1 8	10 Jul 2024	Release of 1.5	Release of 1.6	27 Jun 2026 EUS must be on OCP 4.16	27 Jun 2027 EUS Term 2 must be on OCP 4.16
1.3	4 . 1 2 4 . 1 3 4 . 1 4 4 . 1 5	29 Nov 2023	10 Jul 2024	Release of 1.5	31 Oct 2025 EUS must be on OCP 4.14	31 Oct 2026 EUS Term 2 must be on OCP 4.14

5.1.1.1.1. Unsupported versions of the OADP Operator

Table 5.2. Previous versions of the OADP Operator which are no longer supported

Version	General availability	Full support ended	Maintenance ended
1.2	14 Jun 2023	29 Nov 2023	10 Jul 2024
1.1	01 Sep 2022	14 Jun 2023	29 Nov 2023

1.0	09 Feb 2022	01 Sep 2022	14 Jun 2023

For more details about EUS, see [Extended Update Support](#).

For more details about EUS Term 2, see [Extended Update Support Term 2](#).

Additional resources

- [Backing up etcd](#)

5.2. OADP RELEASE NOTES

5.2.1. OADP 1.5 release notes

The release notes for OpenShift API for Data Protection (OADP) describe new features and enhancements, deprecated features, product recommendations, known issues, and resolved issues.



NOTE

For additional information about OADP, see [OpenShift API for Data Protection \(OADP\) FAQs](#)

5.2.1.1. OADP 1.5.3 release notes

OpenShift API for Data Protection (OADP) 1.5.3 is a Container Grade Only (CGO) release, which is released to refresh the health grades of the containers. No code was changed in the product itself compared to that of OADP 1.5.2.

5.2.1.2. OADP 1.5.2 release notes

The OpenShift API for Data Protection (OADP) 1.5.2 release notes lists resolved issues.

5.2.1.2.1. Resolved issues

Self-signed certificate for internal image backup should not break other BSLs

Before this update, OADP would only process the first custom CA certificate found among all backup storage locations (BSLs) and apply it globally. This behavior prevented multiple BSLs with different CA certificates from working correctly. Additionally, system-trusted certificates were not included, causing failures when connecting to standard services. With this update, OADP now:

- Concatenates all unique CA certificates from AWS BSLs into a single bundle.
- Includes system-trusted certificates automatically.
- Enables multiple BSLs with different custom CA certificates to operate simultaneously.
- Only processes CA certificates when image backup is enabled (default behavior).

This enhancement improves compatibility for environments using multiple storage providers with different certificate requirements, particularly when backing up internal images to AWS S3-compatible storage with self-signed certificates.

[OADP-6765](#)

5.2.1.3. OADP 1.5.1 release notes

The OpenShift API for Data Protection (OADP) 1.5.1 release notes lists new features, resolved issues, known issues, and deprecated features.

5.2.1.3.1. New features

CloudStorage API is fully supported

The **CloudStorage** API feature, available as a Technology Preview before this update, is fully supported from OADP 1.5.1. The **CloudStorage** API automates the creation of a bucket for object storage.

[OADP-3307](#)

New DataProtectionTest custom resource is available

The **DataProtectionTest** (DPT) is a custom resource (CR) that provides a framework to validate your OADP configuration. The DPT CR checks and reports information for the following parameters:

- The upload performance of the backups to the object storage.
- The Container Storage Interface (CSI) snapshot readiness for persistent volume claims.
- The storage bucket configuration, such as encryption and versioning.

Using this information in the DPT CR, you can ensure that your data protection environment is properly configured and performing according to the set configuration.

Note that you must configure **STORAGE_ACCOUNT_ID** when using DPT with OADP on Azure.

[OADP-6300](#)

New node agent load affinity configurations are available

- **Node agent load affinity:** You can schedule the node agent pods on specific nodes by using the **spec.podConfig.nodeSelector** object of the **DataProtectionApplication** (DPA) custom resource (CR). You can add more restrictions on the node agent pods scheduling by using the **nodeagent.loadAffinity** object in the DPA spec.
- **Repository maintenance job affinity configurations:** You can use the repository maintenance job affinity configurations in the **DataProtectionApplication** (DPA) custom resource (CR) only if you use Kopia as the backup repository.
You have the option to configure the load affinity at the global level affecting all repositories, or for each repository. You can also use a combination of global and per-repository configuration.
- **Velero load affinity:** You can use the **podConfig.nodeSelector** object to assign the Velero pod to specific nodes. You can also configure the **velero.loadAffinity** object for pod-level affinity and anti-affinity.

[OADP-5832](#)

Node agent load concurrency is available

With this update, users can control the maximum number of node agent operations that can run simultaneously on each node within their cluster. It also enables better resource management, optimizing backup and restore workflows for improved performance and a more streamlined experience.

5.2.1.3.2. Resolved issues

DataProtectionApplicationSpec overflowed annotation limit, causing potential misconfiguration in deployments

Before this update, the **DataProtectionApplicationSpec** used deprecated **PodAnnotations**, which led to an annotation limit overflow. This caused potential misconfigurations in deployments. In this release, we have added **PodConfig** for annotations in pods deployed by the Operator, ensuring consistent annotations and improved manageability for end users. As a result, deployments should now be more reliable and easier to manage.

[OADP-6454](#)

Root file system for OADP controller manager is now read-only

Before this update, the **manager** container of the **openshift-adp-controller-manager-*** pod was configured to run with a writable root file system. As a consequence, this could allow for tampering with the container's file system or the writing of foreign executables. With this release, the container's security context has been updated to set the root file system to read-only while ensuring necessary functions that require write access, such as the Kopia cache, continue to operate correctly. As a result, the container is hardened against potential threats.

nonAdmin.enable: false in multiple DPAs no longer causes reconcile issues

Before this update, when a user attempted to create a second non-admin **DataProtectionApplication** (DPA) on a cluster where one already existed, the new DPA failed to reconcile. With this release, the restriction on Non-Admin Controller installation to one per cluster has been removed. As a result, users can install multiple Non-Admin Controllers across the cluster without encountering errors.

[OADP-6500](#)

OADP supports self-signed certificates

Before this update, using a self-signed certificate for backup images with a storage provider such as Minio resulted in an **x509: certificate signed by unknown authority** error during the backup process. With this release, certificate validation has been updated to support self-signed certificates in OADP, ensuring successful backups.

[OADP-641](#)

velero describe includes **defaultVolumesToFsBackup**

Before this update, the **velero describe** output command omitted the **defaultVolumesToFsBackup** flag. This affected the visibility of backup configuration details for users. With this release, the **velero describe** output includes the **defaultVolumesToFsBackup** flag information, improving the visibility of backup settings.

[OADP-5762](#)

DPT CR no longer fail when **s3Url** is secured

Before this update, **DataProtectionTest** (DPT) failed to run when **s3Url** was secured due to an

unverified certificate because the DPT CR lacked the ability to skip or add the **caCert** in the **spec** field. As a consequence, data upload failure occurred due to an unverified certificate. With this release, DPT CR has been updated to accept and skip CA cert in **spec** field, resolving SSL verification errors. As a result, DPT no longer fails when using secured **s3Url**.

OADP-6235

Adding a **backupLocation** to DPA with an existing **backupLocation** name is not rejected

Before this update, adding a second **backupLocation** with the same name in **DataProtectionApplication** (DPA) caused OADP to enter an invalid state, leading to Backup and Restore failures due to Velero's inability to read Secret credentials. As a consequence, Backup and Restore operations failed. With this release, the duplicate **backupLocation** names in DPA are no longer allowed, preventing Backup and Restore failures. As a result, duplicate **backupLocation** names are rejected, ensuring seamless data protection.

OADP-6459

5.2.1.3.3. Known issues

The restore fails for backups created on OpenStack using the Cinder CSI driver

When you start a restore operation for a backup that was created on an OpenStack platform using the Cinder Container Storage Interface (CSI) driver, the initial backup only succeeds after the source application is manually scaled down. The restore job fails, preventing you from successfully recovering your application's data and state from the backup. No known workaround exists.

OADP-5552

Datamover pods scheduled on unexpected nodes during backup if the **nodeAgent.loadAffinity** parameter has many elements

Due to an issue in Velero 1.14 and later, the OADP node-agent only processes the first **nodeSelector** element within the **loadAffinity** array. As a consequence, if you define multiple **nodeSelector** objects, all objects except the first are ignored, potentially causing datamover pods to be scheduled on unexpected nodes during a backup.

To work around this problem, consolidate all required **matchExpressions** from multiple **nodeSelector** objects into the first **nodeSelector** object. As a result, all node affinity rules are correctly applied, ensuring datamover pods are scheduled to the appropriate nodes.

OADP-6469

OADP Backup fails when using CA certificates with aliased command

The CA certificate is not stored as a file on the running Velero container. As a consequence, the user experience degraded due to missing **caCert** in Velero container, requiring manual setup and downloads. To work around this problem, manually add cert to the Velero deployment. For instructions, see [Using cacert with velero command aliased via velero deployment](#).

OADP-4668

The **nodeSelector** spec is not supported for the Data Mover restore action

When a Data Protection Application (DPA) is created with the **nodeSelector** field set in the **nodeAgent** parameter, Data Mover restore partially fails instead of completing the restore operation. No known workaround exists.

[OADP-4743](#)

Image streams backups are partially failing when the DPA is configured with **caCert**

An unverified certificate in the S3 connection during backups with **caCert** in **DataProtectionApplication** (DPA) causes the **ocp-django** application's backup to partially fail and result in data loss. No known workaround exists.

[OADP-4817](#)

Kopia does not delete cache on worker node

When the **ephemeral-storage** parameter is configured and running file system restore, the cache is not automatically deleted from the worker node. As a consequence, the **/var** partition overflows during backup restore, causing increased storage usage and potential resource exhaustion. To work around this problem, restart the node agent pod, which clears the cache. As a result, cache is deleted.

[OADP-4855](#)

Google Cloud VSL backups fail with Workload Identity because of invalid project configuration

When performing a **volumeSnapshotLocation** (VSL) backup on Google Cloud Workload Identity, the Velero Google Cloud plugin creates an invalid API request if the Google Cloud project is also specified in the **snapshotLocations** configuration of **DataProtectionApplication** (DPA). As a consequence, the Google Cloud API returns a **RESOURCE_PROJECT_INVALID** error, and the backup job finishes with a **PartiallyFailed** status. No known workaround exists.

[OADP-6697](#)

VSL backups fail for **CloudStorage** API on AWS with STS

The **volumeSnapshotLocation** (VSL) backup fails because of missing the **AZURE_RESOURCE_GROUP** parameter in the credentials file, even if **AZURE_RESOURCE_GROUP** is already mentioned in the **DataProtectionApplication** (DPA) config for VSL. No known workaround exists.

[OADP-6676](#)

Backups of applications with **ImageStreams** fail on Azure with STS

When backing up applications that include image stream resources on an Azure cluster using STS, the OADP plugin incorrectly attempts to locate a secret-based credential for the container registry. As a consequence, the required secret is not found in the STS environment, causing the **ImageStream** custom backup action to fail. This results in the overall backup status marked as **PartiallyFailed**. No known workaround exists.

[OADP-6675](#)

DPA reconciliation fails for **CloudStorageRef** configuration

When a user creates a bucket and uses the **backupLocations.bucket.cloudStorageRef** configuration, bucket credentials are not present in the **DataProtectionApplication** (DPA) custom resource (CR). As

a result, the DPA reconciliation fails even if bucket credentials are present in the **CloudStorage** CR. To work around this problem, add the same credentials to the **backupLocations** section of the DPA CR.

[OADP-6669](#)

5.2.1.3.4. Deprecated features

The `configuration.restic` specification field has been deprecated

With OADP 1.5.0, the **`configuration.restic`** specification field has been deprecated. Use the **`nodeAgent`** section with the **`uploaderType`** field for selecting **`kopia`** or **`restic`** as a **`uploaderType`**. Note that Restic is deprecated in OADP 1.5.0.

[OADP-5158](#)

5.2.1.4. OADP 1.5.0 release notes

The OpenShift API for Data Protection (OADP) 1.5.0 release notes lists resolved issues and known issues.

5.2.1.4.1. New features

OADP 1.5.0 introduces a new Self-Service feature

OADP 1.5.0 introduces a new feature named OADP Self-Service, enabling namespace admin users to back up and restore applications on the OpenShift Container Platform. In the earlier versions of OADP, you needed the cluster-admin role to perform OADP operations such as backing up and restoring an application, creating a backup storage location, and so on.

From OADP 1.5.0 onward, you do not need the cluster-admin role to perform the backup and restore operations. You can use OADP with the namespace admin role. The namespace admin role has administrator access only to the namespace the user is assigned to. You can use the Self-Service feature only after the cluster administrator installs the OADP Operator and provides the necessary permissions.

[OADP-4001](#)

Collecting logs with the `must-gather` tool has been improved with a Markdown summary

You can collect logs, and information about OpenShift API for Data Protection (OADP) custom resources by using the **`must-gather`** tool. The **`must-gather`** data must be attached to all customer cases. This tool generates a Markdown output file with the collected information, which is located in the **`must-gather`** logs clusters directory.

[OADP-5384](#)

`dataMoverPrepareTimeout` and `resourceTimeout` parameters are now added to `nodeAgent` within the DPA

The **`nodeAgent`** field in Data Protection Application (DPA) now includes the following parameters:

- **`dataMoverPrepareTimeout`**: Defines the duration the **`DataUpload`** or **`DataDownload`** process will wait. The default value is 30 minutes.
- **`resourceTimeout`**: Sets the timeout for resource processes not addressed by other specific timeout parameters. The default value is 10 minutes.

[OADP-3736](#)**Use the `spec.configuration.nodeAgent` parameter in DPA for configuring `nodeAgent` daemon set**

Velero no longer uses the **`node-agent-config`** config map for configuring the **`nodeAgent`** daemon set. With this update, you must use the new **`spec.configuration.nodeAgent`** parameter in a Data Protection Application (DPA) for configuring the **`nodeAgent`** daemon set.

[OADP-5042](#)**Configuring DPA with the backup repository configuration config map is now possible**

With Velero 1.15 and later, you can now configure the total size of a cache per repository. This prevents pods from being removed due to running out of ephemeral storage. See the following new parameters added to the **`NodeAgentConfig`** field in DPA:

- **`cacheLimitMB`**: Sets the local data cache size limit in megabytes.
- **`fullMaintenanceInterval`**: The default value is 24 hours. Controls the removal rate of deleted Velero backups from the Kopia repository using the following override options:
 - **`normalGC: 24 hours`**
 - **`fastGC: 12 hours`**
 - **`eagerGC: 6 hours`**

[OADP-5900](#)**Enhancing the node-agent security**

With this update, the following changes are added:

- A new **`configuration`** option is now added to the **`velero`** field in DPA.
- The default value for the **`disableFsBackup`** parameter is **`false`** or **`non-existing`**. With this update, the following options are added to the **`SecurityContext`** field:
 - **`Privileged: true`**
 - **`AllowPrivilegeEscalation: true`**
- If you set the **`disableFsBackup`** parameter to **`true`**, it removes the following mounts from the node-agent:
 - **`host-pods`**
 - **`host-plugins`**
- Modifies that the node-agent is always run as a non-root user.
- Changes the root file system to read only.
- Updates the following mount points with the write access:
 - **`/home/velero`**

- **tmp/credentials**

- Uses the **SeccompProfileTypeRuntimeDefault** option for the **SeccompProfile** parameter.

[OADP-5031](#)**Adds DPA support for parallel item backup**

By default, only one thread processes an item block. Velero 1.16 supports a parallel item backup, where multiple items within a backup can be processed in parallel.

You can use the optional Velero server parameter **--item-block-worker-count** to run additional worker threads to process items in parallel. To enable this in OADP, set the **dpa.Spec.Configuration.Velero.ItemBlockWorkerCount** parameter to an integer value greater than zero.

**NOTE**

Running multiple full backups in parallel is not yet supported.

[OADP-5635](#)**OADP logs are now available in the JSON format**

With the of release OADP 1.5.0, the logs are now available in the JSON format. It helps to have pre-parsed data in their Elastic logs management system.

[OADP-3391](#)**The `oc get dpa` command now displays **RECONCILED** status**

With this release, the **oc get dpa** command now displays **RECONCILED** status instead of displaying only **NAME** and **AGE** to improve user experience. For example:

```
$ oc get dpa -n openshift-adp
NAME          RECONCILED  AGE
velero-sample True         2m51s
```

[OADP-1338](#)**5.2.1.4.2. Resolved issues****Containers now use **FallbackToLogsOnError** for **terminationMessagePolicy****

With this release, the **terminationMessagePolicy** field can now set the **FallbackToLogsOnError** value for the OpenShift API for Data Protection (OADP) Operator containers such as **operator-manager**, **velero**, **node-agent**, and **non-admin-controller**.

This change ensures that if a container exits with an error and the termination message file is empty, OpenShift uses the last portion of the container logs output as the termination message.

[OADP-5183](#)**Namespace admin can now access the application after restore**

Previously, the namespace admin could not execute an application after the restore operation with the following errors:

- **exec operation is not allowed because the pod's security context exceeds your permissions**
- **unable to validate against any security context constraint**
- **not usable by user or serviceaccount, provider restricted-v2**

With this update, this issue is now resolved and the namespace admin can access the application successfully after the restore.

[OADP-5611](#)

Specifying status restoration at the individual resource instance level using the annotation is now possible

Previously, status restoration was only configured at the resource type using the **restoreStatus** field in the **Restore** custom resource (CR).

With this release, you can now specify the status restoration at the individual resource instance level using the following annotation:

```
metadata:
  annotations:
    velero.io/restore-status: "true"
```

[OADP-5968](#)

Restore is now successful with **excludedClusterScopedResources**

Previously, on performing the backup of an application with the **excludedClusterScopedResources** field set to **storageclasses, Namespace** parameter, the backup was successful but the restore partially failed. With this update, the restore is successful.

[OADP-5239](#)

Backup is completed even if it gets restarted during the **waitingForPluginOperations** phase

Previously, a backup was marked as failed with the following error message:

```
failureReason: found a backup with status "InProgress" during the server starting,
mark it as "Failed"
```

With this update, the backup is completed if it gets restarted during the **waitingForPluginOperations** phase.

[OADP-2941](#)

Error messages are now more informative when the `disableFsbackup`` parameter is set to **true** in DPA

Previously, when the **spec.configuration.velero.disableFsBackup** field from a Data Protection Application (DPA) was set to **true**, the backup partially failed with an error, which was not informative.

This update makes error messages more useful for troubleshooting issues. For example, error messages indicating that **disableFsBackup: true** is the issue in a DPA or not having access to a DPA if it is for non-administrator users.

OADP-5952

Handles AWS STS credentials in the `parseAWSSecret`

Previously, AWS credentials using STS authentication were not properly validated.

With this update, the **`parseAWSSecret`** function detects STS-specific fields, and updates the **`ensureSecretDataExists`** function to handle STS profiles correctly.

OADP-6105

The **`repositoryMaintenance`** job affinity config is available to configure

Previously, the new configurations for repository maintenance job pod affinity was missing from a DPA specification.

With this update, the **`repositoryMaintenance`** job affinity config is now available to map a **`BackupRepository`** identifier to its configuration.

OADP-6134

The **`ValidationErrors`** field fades away once the CR specification is correct

Previously, when a schedule CR was created with a wrong **`spec.schedule`** value and the same was later patched with a correct value, the **`ValidationErrors`** field still existed. Consequently, the **`ValidationErrors`** field was displaying incorrect information even though the spec was correct.

With this update, the **`ValidationErrors`** field fades away once the CR specification is correct.

OADP-5419

The **`volumeSnapshotContents`** custom resources are restored when the **`includedNamespaces`** field is used in **`restoreSpec`**

Previously, when a restore operation was triggered with the **`includedNamespace`** field in a restore specification, restore operation was completed successfully but no **`volumeSnapshotContents`** custom resources (CR) were created and the PVCs were in a **`Pending`** status.

With this update, **`volumeSnapshotContents`** CR are restored even when the **`includedNamespaces`** field is used in **`restoreSpec`**. As a result, an application pod is in a **`Running`** state after restore.

OADP-5939

OADP operator successfully creates bucket on top of AWS

Previously, the container was configured with the **`readOnlyRootFilesystem: true`** setting for security, but the code attempted to create temporary files in the **`/tmp`** directory using the **`os.CreateTemp()`** function. Consequently, while using the AWS STS authentication with the Cloud Credential Operator (CCO) flow, OADP failed to create temporary files that were required for AWS credential handling with the following error:

```
ERROR unable to determine if bucket exists. {"error": "open /tmp/aws-shared-credentials1211864681: read-only file system"}
```

With this update, the following changes are added to address this issue:

- A new **emptyDir** volume named **tmp-dir** to the controller pod specification.
- A volume mount to the container, which mounts this volume to the **/tmp** directory.
- For security best practices, the **readOnlyRootFilesystem: true** is maintained.
- Replaced the deprecated **ioutil.TempFile()** function with the recommended **os.CreateTemp()** function.
- Removed the unnecessary **io/ioutil** import, which is no longer needed.

[OADP-6019](#)

For a complete list of all issues resolved in this release, see the list of [OADP 1.5.0 resolved issues](#) in Jira.

5.2.1.4.3. Known issues

Kopia does not delete all the artifacts after backup expiration

Even after deleting a backup, Kopia does not delete the volume artifacts from the **\${bucket_name}/kopia/\$openshift-adp** on the S3 location after the backup expired. Information related to the expired and removed data files remains in the metadata. To ensure that OpenShift API for Data Protection (OADP) functions properly, the data is not deleted, and it exists in the **/kopia/** directory, for example:

- **kopia.repository**: Main repository format information such as encryption, version, and other details.
- **kopia.blobcfg**: Configuration for how data blobs are named.
- **kopia.maintenance**: Tracks maintenance owner, schedule, and last successful build.
- **log**: Log blobs.

[OADP-5131](#)

For a complete list of all known issues in this release, see the list of [OADP 1.5.0 known issues](#) in Jira.

5.2.1.4.4. Deprecated features

The **configuration.restic** specification field has been deprecated

With OpenShift API for Data Protection (OADP) 1.5.0, the **configuration.restic** specification field has been deprecated. Use the **nodeAgent** section with the **uploaderType** field for selecting **kopia** or **restic** as a **uploaderType**. Note that, Restic is deprecated in OpenShift API for Data Protection (OADP) 1.5.0.

[OADP-5158](#)

5.2.1.4.5. Technology Preview

Support for HyperShift hosted OpenShift clusters is available as a Technology Preview

OADP can support and facilitate application migrations within HyperShift hosted OpenShift clusters as a Technology Preview. It ensures a seamless backup and restore operation for applications in hosted clusters.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).

[OADP-3930](#)

5.2.1.5. Upgrading OADP 1.4 to 1.5



NOTE

Always upgrade to the next minor version. Do not skip versions. To update to a later version, upgrade only one channel at a time. For example, to upgrade from OADP 1.1 to 1.3, upgrade first to 1.2, and then to 1.3.

5.2.1.5.1. Changes from OADP 1.4 to 1.5

The Velero server has been updated from version 1.14 to 1.16.

This changes the following:

Version Support changes

OpenShift API for Data Protection implements a streamlined version support policy. Red Hat supports only one version of OpenShift API for Data Protection (OADP) on one OpenShift version to ensure better stability and maintainability. OADP 1.5.0 is only supported on OpenShift 4.19 version.

OADP Self-Service

OADP 1.5.0 introduces a new feature named OADP Self-Service, enabling namespace admin users to back up and restore applications on the OpenShift Container Platform. In the earlier versions of OADP, you needed the cluster-admin role to perform OADP operations such as backing up and restoring an application, creating a backup storage location, and so on.

From OADP 1.5.0 onward, you do not need the cluster-admin role to perform the backup and restore operations. You can use OADP with the namespace admin role. The namespace admin role has administrator access only to the namespace the user is assigned to. You can use the Self-Service feature only after the cluster administrator installs the OADP Operator and provides the necessary permissions.

backupPVC and restorePVC configurations

A **backupPVC** resource is an intermediate persistent volume claim (PVC) to access data during the data movement backup operation. You create a **readonly** backup PVC by using the **nodeAgent.backupPVC** section of the **DataProtectionApplication** (DPA) custom resource.

A **restorePVC** resource is an intermediate PVC that is used to write data during the Data Mover restore operation.

You can configure **restorePVC** in the DPA by using the **ignoreDelayBinding** field.

5.2.1.5.2. Backing up the DPA configuration

You must back up your current **DataProtectionApplication** (DPA) configuration.

Procedure

- Save your current DPA configuration by running the following command:

Example command

```
$ oc get dpa -n openshift-adp -o yaml > dpa.orig.backup
```

5.2.1.5.3. Upgrading the OADP Operator

You can upgrade the OpenShift API for Data Protection (OADP) Operator using the following procedure.



NOTE

Do not install OADP 1.5.0 on a OpenShift 4.18 cluster.

Prerequisites

- You have installed the latest OADP 1.4.6.
- You have backed up your data.

Procedure

1. Upgrade OpenShift 4.18 to OpenShift 4.19.



NOTE

OpenShift API for Data Protection (OADP) 1.4 is not supported on OpenShift 4.19.

2. Change your subscription channel for the OADP Operator from **stable-1.4** to **stable**.
3. Wait for the Operator and containers to update and restart.

5.2.1.5.4. Converting DPA to the new version for OADP 1.5.0

The OpenShift API for Data Protection (OADP) 1.4 is not supported on OpenShift 4.19. You can convert Data Protection Application (DPA) to the new OADP 1.5 version by using the new **spec.configuration.nodeAgent** field and its sub-fields.

Procedure

1. To configure **nodeAgent** daemon set, use the **spec.configuration.nodeAgent** parameter in DPA. See the following example:

Example DataProtectionApplication configuration

```
...
spec:
  configuration:
    nodeAgent:
```

```
enable: true
uploaderType: kopia
...
```

2. To configure **nodeAgent** daemon set by using the **ConfigMap** resource named **node-agent-config**, see the following example configuration:

Example config map

```
...
spec:
  configuration:
    nodeAgent:
      backupPVC:
        ...
      loadConcurrency:
        ...
      podResources:
        ...
      restorePVC:
        ...
    ...
```

5.2.1.5.5. Verifying the upgrade

You can verify the OpenShift API for Data Protection (OADP) upgrade by using the following procedure.

Procedure

1. Verify that the **DataProtectionApplication** (DPA) has been reconciled successfully:

```
$ oc get dpa dpa-sample -n openshift-adp
```

Example output

NAME	RECONCILED	AGE
dpa-sample	True	2m51s



NOTE

The **RECONCILED** column must be **True**.

2. Verify that the installation finished by viewing the OADP resources by running the following command:

```
$ oc get all -n openshift-adp
```

Example output

NAME	READY	STATUS	RESTARTS	AGE
pod/node-agent-9pjz9	1/1	Running	0	3d17h

```

pod/node-agent-fmn84          1/1   Running 0    3d17h
pod/node-agent-xw2dg          1/1   Running 0    3d17h
pod/openshift-adp-controller-manager-76b8bc8d7b-kgkcw 1/1   Running 0    3d17h
pod/velero-64475b8c5b-nh2qc   1/1   Running 0    3d17h

```

```

NAME                                TYPE      CLUSTER-IP      EXTERNAL-IP
PORT(S)  AGE
service/openshift-adp-controller-manager-metrics-service ClusterIP 172.30.194.192
<none>   8443/TCP 3d17h
service/openshift-adp-velero-metrics-svc          ClusterIP 172.30.190.174 <none>
8085/TCP 3d17h

```

```

NAME                                DESIRED  CURRENT  READY  UP-TO-DATE  AVAILABLE  NODE
SELECTOR  AGE
daemonset.apps/node-agent 3      3      3      3      3      <none>      3d17h

```

```

NAME                                READY  UP-TO-DATE  AVAILABLE  AGE
deployment.apps/openshift-adp-controller-manager 1/1    1      1      3d17h
deployment.apps/velero                        1/1    1      1      3d17h

```

```

NAME                                DESIRED  CURRENT  READY  AGE
replicaset.apps/openshift-adp-controller-manager-76b8bc8d7b 1      1      1      3d17h
replicaset.apps/openshift-adp-controller-manager-85ff975b8 0      0      0      3d17h
replicaset.apps/velero-64475b8c5b                        1      1      1      3d17h
replicaset.apps/velero-8b5bc54fd                        0      0      0      3d17h
replicaset.apps/velero-f5c9ffb66                        0      0      0      3d17h

```

**NOTE**

The **node-agent** pods are created only while using **restic** or **kopia** in **DataProtectionApplication** (DPA). In OADP 1.4.0 and OADP 1.3.0 version, the **node-agent** pods are labeled as **restic**.

3. Verify the backup storage location and confirm that the **PHASE** is **Available** by running the following command:

```
$ oc get backupstoragelocations.velero.io -n openshift-adp
```

Example output

```

NAME          PHASE    LAST VALIDATED  AGE    DEFAULT
dpa-sample-1  Available 1s             3d16h true

```

5.3. OADP PERFORMANCE

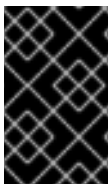
5.3.1. OADP recommended network settings

For a supported experience with OpenShift API for Data Protection (OADP), you should have a stable and resilient network across OpenShift nodes, S3 storage, and in supported cloud environments that meet OpenShift network requirement recommendations.

To ensure successful backup and restore operations for deployments with remote S3 buckets located off-cluster with suboptimal data paths, it is recommended that your network settings meet the following minimum requirements in such less optimal conditions:

- Bandwidth (network upload speed to object storage): Greater than 2 Mbps for small backups and 10-100 Mbps depending on the data volume for larger backups.
- Packet loss: 1%
- Packet corruption: 1%
- Latency: 100ms

Ensure that your OpenShift Container Platform network performs optimally and meets OpenShift Container Platform network requirements.



IMPORTANT

Although Red Hat provides supports for standard backup and restore failures, it does not provide support for failures caused by network settings that do not meet the recommended thresholds.

5.4. OADP FEATURES AND PLUGINS

OpenShift API for Data Protection (OADP) features provide options for backing up and restoring applications.

The default plugins enable Velero to integrate with certain cloud providers and to back up and restore OpenShift Container Platform resources.

5.4.1. OADP features

OpenShift API for Data Protection (OADP) supports the following features:

Backup

You can use OADP to back up all applications on the OpenShift Platform, or you can filter the resources by type, namespace, or label.

OADP backs up Kubernetes objects and internal images by saving them as an archive file on object storage. OADP backs up persistent volumes (PVs) by creating snapshots with the native cloud snapshot API or with the Container Storage Interface (CSI). For cloud providers that do not support snapshots, OADP backs up resources and PV data with Restic.



NOTE

You must exclude Operators from the backup of an application for backup and restore to succeed.

Restore

You can restore resources and PVs from a backup. You can restore all objects in a backup or filter the objects by namespace, PV, or label.

**NOTE**

You must exclude Operators from the backup of an application for backup and restore to succeed.

Schedule

You can schedule backups at specified intervals.

Hooks

You can use hooks to run commands in a container on a pod, for example, **fsfreeze** to freeze a file system. You can configure a hook to run before or after a backup or restore. Restore hooks can run in an init container or in the application container.

5.4.2. OADP plugins

The OpenShift API for Data Protection (OADP) provides default Velero plugins that are integrated with storage providers to support backup and snapshot operations. You can create [custom plugins](#) based on the Velero plugins.

OADP also provides plugins for OpenShift Container Platform resource backups, OpenShift Virtualization resource backups, and Container Storage Interface (CSI) snapshots.

Table 5.3. OADP plugins

OADP plugin	Function	Storage location
aws	Backs up and restores Kubernetes objects.	AWS S3
	Backs up and restores volumes with snapshots.	AWS EBS
azure	Backs up and restores Kubernetes objects.	Microsoft Azure Blob storage
	Backs up and restores volumes with snapshots.	Microsoft Azure Managed Disks
gcp	Backs up and restores Kubernetes objects.	Google Cloud Storage
	Backs up and restores volumes with snapshots.	Google Compute Engine Disks
openshift	Backs up and restores OpenShift Container Platform resources. ^[1]	Object store
kubevirt	Backs up and restores OpenShift Virtualization resources. ^[2]	Object store

OADP plugin	Function	Storage location
csi	Backs up and restores volumes with CSI snapshots. ^[3]	Cloud storage that supports CSI snapshots
hypershift	Backs up and restores HyperShift hosted cluster resources. ^[4]	Object store

1. Mandatory.
2. Virtual machine disks are backed up with CSI snapshots or Restic.
3. The **csi** plugin uses the Kubernetes CSI snapshot API.
 - OADP 1.1 or later uses **snapshot.storage.k8s.io/v1**
 - OADP 1.0 uses **snapshot.storage.k8s.io/v1beta1**
4. Do not add the **hypershift** plugin in the **DataProtectionApplication** custom resource if the cluster is not a HyperShift hosted cluster.

5.4.3. About OADP Velero plugins

You can configure two types of plugins when you install Velero:

- Default cloud provider plugins
- Custom plugins

Both types of plugin are optional, but most users configure at least one cloud provider plugin.

5.4.3.1. Default Velero cloud provider plugins

You can install any of the following default Velero cloud provider plugins when you configure the **oadp_v1alpha1_dpa.yaml** file during deployment:

- **aws** (Amazon Web Services)
- **gcp** (Google Cloud)
- **azure** (Microsoft Azure)
- **openshift** (OpenShift Velero plugin)
- **csi** (Container Storage Interface)
- **kubevirt** (KubeVirt)

You specify the desired default plugins in the **oadp_v1alpha1_dpa.yaml** file during deployment.

Example file

The following **.yaml** file installs the **openshift**, **aws**, **azure**, and **gcp** plugins:

–

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: dpa-sample
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - aws
        - azure
        - gcp

```

5.4.3.2. Custom Velero plugins

You can install a custom Velero plugin by specifying the plugin **image** and **name** when you configure the **oadp_v1alpha1_dpa.yaml** file during deployment.

You specify the desired custom plugins in the **oadp_v1alpha1_dpa.yaml** file during deployment.

Example file

The following **.yaml** file installs the default **openshift**, **azure**, and **gcp** plugins and a custom plugin that has the name **custom-plugin-example** and the image **quay.io/example-repo/custom-velero-plugin**:

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: dpa-sample
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - azure
        - gcp
      customPlugins:
        - name: custom-plugin-example
          image: quay.io/example-repo/custom-velero-plugin

```

5.4.4. Supported architectures for OADP

OpenShift API for Data Protection (OADP) supports the following architectures:

- AMD64
- ARM64
- PPC64le
- s390x

**NOTE**

OADP 1.2.0 and later versions support the ARM64 architecture.

5.4.5. OADP support for IBM Power and IBM Z

OpenShift API for Data Protection (OADP) is platform neutral. The information that follows relates only to IBM Power® and to IBM Z®.

- OADP 1.3.6 was tested successfully against OpenShift Container Platform 4.12, 4.13, 4.14, and 4.15 for both IBM Power® and IBM Z®. The sections that follow give testing and support information for OADP 1.3.6 in terms of backup locations for these systems.
- OADP 1.4.6 was tested successfully against OpenShift Container Platform 4.14, 4.15, 4.16, and 4.17 for both IBM Power® and IBM Z®. The sections that follow give testing and support information for OADP 1.4.6 in terms of backup locations for these systems.
- OADP 1.5.0 was tested successfully against OpenShift Container Platform 4.19 for both IBM Power® and IBM Z®. The sections that follow give testing and support information for OADP 1.5.0 in terms of backup locations for these systems.

5.4.5.1. OADP support for target backup locations using IBM Power

- IBM Power® running with OpenShift Container Platform 4.12, 4.13, 4.14, and 4.15, and OADP 1.3.6 was tested successfully against an AWS S3 backup location target. Although the test involved only an AWS S3 target, Red Hat supports running IBM Power® with OpenShift Container Platform 4.13, 4.14, and 4.15, and OADP 1.3.6 against all S3 backup location targets, which are not AWS, as well.
- IBM Power® running with OpenShift Container Platform 4.14, 4.15, 4.16, and 4.17, and OADP 1.4.6 was tested successfully against an AWS S3 backup location target. Although the test involved only an AWS S3 target, Red Hat supports running IBM Power® with OpenShift Container Platform 4.14, 4.15, 4.16, and 4.17, and OADP 1.4.6 against all S3 backup location targets, which are not AWS, as well.
- IBM Power® running with OpenShift Container Platform 4.19 and OADP 1.5.0 was tested successfully against an AWS S3 backup location target. Although the test involved only an AWS S3 target, Red Hat supports running IBM Power® with OpenShift Container Platform 4.19 and OADP 1.5.0 against all S3 backup location targets, which are not AWS, as well.

5.4.5.2. OADP testing and support for target backup locations using IBM Z

- IBM Z® running with OpenShift Container Platform 4.12, 4.13, 4.14, and 4.15, and 1.3.6 was tested successfully against an AWS S3 backup location target. Although the test involved only an AWS S3 target, Red Hat supports running IBM Z® with OpenShift Container Platform 4.13, 4.14, and 4.15, and 1.3.6 against all S3 backup location targets, which are not AWS, as well.
- IBM Z® running with OpenShift Container Platform 4.14, 4.15, 4.16, and 4.17, and 1.4.6 was tested successfully against an AWS S3 backup location target. Although the test involved only an AWS S3 target, Red Hat supports running IBM Z® with OpenShift Container Platform 4.14, 4.15, 4.16, and 4.17, and 1.4.6 against all S3 backup location targets, which are not AWS, as well.
- IBM Z® running with OpenShift Container Platform 4.19 and OADP 1.5.0 was tested successfully against an AWS S3 backup location target. Although the test involved only an AWS S3 target, Red Hat supports running IBM Z® with OpenShift Container Platform 4.19 and OADP 1.5.0 against all S3 backup location targets, which are not AWS, as well.

5.4.5.2.1. Known issue of OADP using IBM Power(R) and IBM Z(R) platforms

- Currently, there are backup method restrictions for Single-node OpenShift clusters deployed on IBM Power® and IBM Z® platforms. Only NFS storage is currently compatible with Single-node OpenShift clusters on these platforms. In addition, only the File System Backup (FSB) methods such as Kopia and Restic are supported for backup and restore operations. There is currently no workaround for this issue.

5.4.6. OADP and FIPS

Federal Information Processing Standards (FIPS) are a set of computer security standards developed by the United States federal government in line with the Federal Information Security Management Act (FISMA).

OpenShift API for Data Protection (OADP) has been tested and works on FIPS-enabled OpenShift Container Platform clusters.

5.4.7. Avoiding the Velero plugin panic error

A missing secret can cause a panic error for the Velero plugin during image stream backups.

When the backup and the Backup Storage Location (BSL) are managed outside the scope of the Data Protection Application (DPA), the OADP controller does not create the relevant **oadp-<bsl_name>-<bsl_provider>-registry-secret** parameter.

During the backup operation, the OpenShift Velero plugin panics on the **imagestream** backup, with the following panic error:

```
024-02-27T10:46:50.028951744Z time="2024-02-27T10:46:50Z" level=error msg="Error backing up item"
backup=openshift-adp/<backup name> error="error executing custom action
(groupResource=imagestreams.image.openshift.io,
namespace=<BSL Name>, name=postgres): rpc error: code = Aborted desc = plugin panicked:
runtime error: index out of range with length 1, stack trace: goroutine 94...
```

Use the following workaround to avoid the Velero plugin panic error.

Procedure

- Label the custom BSL with the relevant label by using the following command:

```
$ oc label backupstoragelocations.velero.io <bsl_name> app.kubernetes.io/component=bsl
```

- After the BSL is labeled, wait until the DPA reconciles.



NOTE

You can force the reconciliation by making any minor change to the DPA itself.

Verification

- After the DPA is reconciled, confirm that the parameter has been created and that the correct registry data has been populated into it by entering the following command:

```
$ oc -n openshift-adp get secret/oadp-<bsl_name>-<bsl_provider>-registry-secret -o json | jq
-r '.data'
```

5.4.8. Workaround for OpenShift ADP Controller segmentation fault

If you configure a Data Protection Application (DPA) with both **cloudstorage** and **restic** enabled, the **openshift-adp-controller-manager** pod crashes and restarts indefinitely until the pod fails with a crash loop segmentation fault.

Define either **velero** or **cloudstorage** when you configure a DPA. Otherwise, the **openshift-adp-controller-manager** pod fails with a crash loop segmentation fault due to the following settings:

- If you define both **velero** and **cloudstorage**, the **openshift-adp-controller-manager** fails.
- If you do not define both **velero** and **cloudstorage**, the **openshift-adp-controller-manager** fails.

For more information about this issue, see [OADP-1054](#).

5.5. OADP USE CASES

5.5.1. Backup using OpenShift API for Data Protection and Red Hat OpenShift Data Foundation (ODF)

Following is a use case for using OADP and ODF to back up an application.

5.5.1.1. Backing up an application using OADP and ODF

In this use case, you back up an application by using OADP and store the backup in an object storage provided by Red Hat OpenShift Data Foundation (ODF).

- You create an object bucket claim (OBC) to configure the backup storage location. You use ODF to configure an Amazon S3-compatible object storage bucket. ODF provides MultiCloud Object Gateway (NooBaa MCG) and Ceph Object Gateway, also known as RADOS Gateway (RGW), object storage service. In this use case, you use NooBaa MCG as the backup storage location.
- You use the NooBaa MCG service with OADP by using the **aws** provider plugin.
- You configure the Data Protection Application (DPA) with the backup storage location (BSL).
- You create a backup custom resource (CR) and specify the application namespace to back up.
- You create and verify the backup.

Prerequisites

- You installed the OADP Operator.
- You installed the ODF Operator.
- You have an application with a database running in a separate namespace.

Procedure

1. Create an OBC manifest file to request a NooBaa MCG bucket as shown in the following example:

```
apiVersion: objectbucket.io/v1alpha1
kind: ObjectBucketClaim
metadata:
  name: test-obc
  namespace: openshift-adp
spec:
  storageClassName: openshift-storage.noobaa.io
  generateBucketName: test-backup-bucket
```

where:

test-obc

Specifies the name of the object bucket claim.

test-backup-bucket

Specifies the name of the bucket.

2. Create the OBC by running the following command:

```
$ oc create -f <obc_file_name>
```

where:

<obc_file_name>

Specifies the file name of the object bucket claim manifest.

3. When you create an OBC, ODF creates a **secret** and a **config map** with the same name as the object bucket claim. The **secret** has the bucket credentials, and the **config map** has information to access the bucket. To get the bucket name and bucket host from the generated config map, run the following command:

```
$ oc extract --to=- cm/test-obc
```

test-obc is the name of the OBC.

Example output

```
# BUCKET_NAME
backup-c20...41fd
# BUCKET_PORT
443
# BUCKET_REGION

# BUCKET_SUBREGION

# BUCKET_HOST
s3.openshift-storage.svc
```

4. To get the bucket credentials from the generated **secret**, run the following command:

```
$ oc extract --to=- secret/test-obc
```

Example output

```
# AWS_ACCESS_KEY_ID
ebYR....xLNMc
# AWS_SECRET_ACCESS_KEY
YXf...+NaCkdyC3QPym
```

5. Get the public URL for the S3 endpoint from the s3 route in the **openshift-storage** namespace by running the following command:

```
$ oc get route s3 -n openshift-storage
```

6. Create a **cloud-credentials** file with the object bucket credentials as shown in the following command:

```
[default]
aws_access_key_id=<AWS_ACCESS_KEY_ID>
aws_secret_access_key=<AWS_SECRET_ACCESS_KEY>
```

7. Create the **cloud-credentials** secret with the **cloud-credentials** file content as shown in the following command:

```
$ oc create secret generic \
  cloud-credentials \
  -n openshift-adp \
  --from-file cloud=cloud-credentials
```

8. Configure the Data Protection Application (DPA) as shown in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: oadp-backup
  namespace: openshift-adp
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
    velero:
      defaultPlugins:
        - aws
        - openshift
        - csi
      defaultSnapshotMoveData: true
  backupLocations:
    - velero:
        config:
          profile: "default"
          region: noobaa
          s3Url: https://s3.openshift-storage.svc
```

```
s3ForcePathStyle: "true"
insecureSkipTLSVerify: "true"
provider: aws
default: true
credential:
  key: cloud
  name: cloud-credentials
objectStorage:
  bucket: <bucket_name>
  prefix: oadp
```

where:

defaultSnapshotMoveData

Set to **true** to use the OADP Data Mover to enable movement of Container Storage Interface (CSI) snapshots to a remote object storage.

s3Url

Specifies the S3 URL of ODF storage.

<bucket_name>

Specifies the bucket name.

9. Create the DPA by running the following command:

```
$ oc apply -f <dpa_filename>
```

10. Verify that the DPA is created successfully by running the following command. In the example output, you can see the **status** object has **type** field set to **Reconciled**. This means, the DPA is successfully created.

```
$ oc get dpa -o yaml
```

Example output

```
apiVersion: v1
items:
- apiVersion: oadp.openshift.io/v1alpha1
  kind: DataProtectionApplication
  metadata:
    namespace: openshift-adp
    #...#
  spec:
    backupLocations:
    - velero:
        config:
          #...#
  status:
    conditions:
    - lastTransitionTime: "20....9:54:02Z"
      message: Reconcile complete
      reason: Complete
      status: "True"
      type: Reconciled
```



```
kind: List
metadata:
  resourceVersion: ""
```

11. Verify that the backup storage location (BSL) is available by running the following command:

```
$ oc get backupstoragelocations.velero.io -n openshift-adp
```

Example output

```
NAME          PHASE    LAST VALIDATED  AGE  DEFAULT
dpa-sample-1  Available  3s             15s  true
```

12. Configure a backup CR as shown in the following example:

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: test-backup
  namespace: openshift-adp
spec:
  includedNamespaces:
    - <application_namespace>
```

where:

<application_namespace>

Specifies the namespace for the application to back up.

13. Create the backup CR by running the following command:

```
$ oc apply -f <backup_cr_filename>
```

Verification

- Verify that the backup object is in the **Completed** phase by running the following command. For more details, see the example output.

```
$ oc describe backup test-backup -n openshift-adp
```

Example output

```
Name:      test-backup
Namespace: openshift-adp
# ....#
Status:
  Backup Item Operations Attempted: 1
  Backup Item Operations Completed: 1
  Completion Timestamp:      2024-09-25T10:17:01Z
  Expiration:                2024-10-25T10:16:31Z
  Format Version:            1.1.0
  Hook Status:
  Phase: Completed
```

```

Progress:
  Items Backed Up: 34
  Total Items:    34
  Start Timestamp: 2024-09-25T10:16:31Z
  Version:        1
  Events:         <none>

```

5.5.2. OpenShift API for Data Protection (OADP) restore use case

Following is a use case for using OADP to restore a backup to a different namespace.

5.5.2.1. Restoring an application to a different namespace using OADP

Restore a backup of an application by using OADP to a new target namespace, **test-restore-application**. To restore a backup, you create a restore custom resource (CR) as shown in the following example. In the restore CR, the source namespace refers to the application namespace that you included in the backup. You then verify the restore by changing your project to the new restored namespace and verifying the resources.

Prerequisites

- You installed the OADP Operator.
- You have the backup of an application to be restored.

Procedure

1. Create a restore CR as shown in the following example:

```

apiVersion: velero.io/v1
kind: Restore
metadata:
  name: test-restore
  namespace: openshift-adp
spec:
  backupName: <backup_name>
  restorePVs: true
  namespaceMapping:
    <application_namespace>: test-restore-application

```

where:

test-restore

Specifies the name of the restore CR.

<backup_name>

Specifies the name of the backup.

<application_namespace>

Specifies the target namespace to restore to. **namespaceMapping** maps the source application namespace to the target application namespace. **test-restore-application** is the name of target namespace where you want to restore the backup.

2. Apply the restore CR by running the following command:

■

```
$ oc apply -f <restore_cr_filename>
```

Verification

1. Verify that the restore is in the **Completed** phase by running the following command:

```
$ oc describe restores.velero.io <restore_name> -n openshift-adp
```

2. Change to the restored namespace **test-restore-application** by running the following command:

```
$ oc project test-restore-application
```

3. Verify the restored resources such as persistent volume claim (pvc), service (svc), deployment, secret, and config map by running the following command:

```
$ oc get pvc,svc,deployment,secret,configmap
```

Example output

```
NAME                                STATUS  VOLUME
persistentvolumeclaim/mysql  Bound  pvc-9b3583db-...-14b86

NAME      TYPE      CLUSTER-IP  EXTERNAL-IP  PORT(S)  AGE
service/mysql  ClusterIP  172....157  <none>       3306/TCP  2m56s
service/todolist  ClusterIP  172.....15  <none>       8000/TCP  2m56s

NAME      READY  UP-TO-DATE  AVAILABLE  AGE
deployment.apps/mysql  0/1    1           0          2m55s

NAME                                TYPE      DATA  AGE
secret/builder-dockercfg-6bfmd      kubernet... 1      2m57s
secret/default-dockercfg-hz9kz      kubernet... 1      2m57s
secret/deployer-dockercfg-86cvd      kubernet... 1      2m57s
secret/mysql-persistent-sa-dockercfg-rgp9b  kubernet... 1      2m57s

NAME      DATA  AGE
configmap/kube-root-ca.crt  1      2m57s
configmap/openshift-service-ca.crt  1      2m57s
```

5.5.3. Including a self-signed CA certificate during backup

You can include a self-signed Certificate Authority (CA) certificate in the Data Protection Application (DPA) and then back up an application. You store the backup in a NooBaa bucket provided by Red Hat OpenShift Data Foundation (ODF).

5.5.3.1. Backing up an application and its self-signed CA certificate

The **s3.openshift-storage.svc** service, provided by ODF, uses a Transport Layer Security protocol (TLS) certificate that is signed with the self-signed service CA.

To prevent a **certificate signed by unknown authority** error, you must include a self-signed CA certificate in the backup storage location (BSL) section of **DataProtectionApplication** custom resource (CR). For this situation, you must complete the following tasks:

- Request a NooBaa bucket by creating an object bucket claim (OBC).
- Extract the bucket details.
- Include a self-signed CA certificate in the **DataProtectionApplication** CR.
- Back up an application.

Prerequisites

- You installed the OADP Operator.
- You installed the ODF Operator.
- You have an application with a database running in a separate namespace.

Procedure

1. Create an OBC manifest to request a NooBaa bucket as shown in the following example:

```
apiVersion: objectbucket.io/v1alpha1
kind: ObjectBucketClaim
metadata:
  name: test-obc
  namespace: openshift-adp
spec:
  storageClassName: openshift-storage.noobaa.io
  generateBucketName: test-backup-bucket
```

where:

test-obc

Specifies the name of the object bucket claim.

test-backup-bucket

Specifies the name of the bucket.

2. Create the OBC by running the following command:

```
$ oc create -f <obc_file_name>
```

3. When you create an OBC, ODF creates a **secret** and a **ConfigMap** with the same name as the object bucket claim. The **secret** object contains the bucket credentials, and the **ConfigMap** object contains information to access the bucket. To get the bucket name and bucket host from the generated config map, run the following command:

```
$ oc extract --to=- cm/test-obc
```

test-obc is the name of the OBC.

Example output

```
# BUCKET_NAME
backup-c20...41fd
# BUCKET_PORT
443
# BUCKET_REGION

# BUCKET_SUBREGION

# BUCKET_HOST
s3.openshift-storage.svc
```

4. To get the bucket credentials from the **secret** object, run the following command:

```
$ oc extract --to=- secret/test-obc
```

Example output

```
# AWS_ACCESS_KEY_ID
ebYR....xLNMc
# AWS_SECRET_ACCESS_KEY
YXf...+NaCkdyC3QPym
```

5. Create a **cloud-credentials** file with the object bucket credentials by using the following example configuration:

```
[default]
aws_access_key_id=<AWS_ACCESS_KEY_ID>
aws_secret_access_key=<AWS_SECRET_ACCESS_KEY>
```

6. Create the **cloud-credentials** secret with the **cloud-credentials** file content by running the following command:

```
$ oc create secret generic \
  cloud-credentials \
  -n openshift-adp \
  --from-file cloud=cloud-credentials
```

7. Extract the service CA certificate from the **openshift-service-ca.crt** config map by running the following command. Ensure that you encode the certificate in **Base64** format and note the value to use in the next step.

```
$ oc get cm/openshift-service-ca.crt \
  -o jsonpath='{.data.service-ca\.crt}' | base64 -w0; echo
```

Example output

```
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0...
....gpwOHMwaG9CRmk5a3....FLS0tLS0K
```

8. Configure the **DataProtectionApplication** CR manifest file with the bucket name and CA certificate as shown in the following example:

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: oadp-backup
  namespace: openshift-adp
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
    velero:
      defaultPlugins:
        - aws
        - openshift
        - csi
      defaultSnapshotMoveData: true
  backupLocations:
    - velero:
        config:
          profile: "default"
          region: noobaa
          s3Url: https://s3.openshift-storage.svc
          s3ForcePathStyle: "true"
          insecureSkipTLSVerify: "false"
        provider: aws
        default: true
        credential:
          key: cloud
          name: cloud-credentials
        objectStorage:
          bucket: <bucket_name>
          prefix: oadp
          caCert: <ca_cert>

```

where:

insecureSkipTLSVerify

Specifies whether SSL/TLS security is enabled. If set to **true**, SSL/TLS security is disabled. If set to **false**, SSL/TLS security is enabled.

<bucket_name>

Specifies the name of the bucket extracted in an earlier step.

<ca_cert>

Specifies the **Base64** encoded certificate from the previous step.

9. Create the **DataProtectionApplication** CR by running the following command:

```
$ oc apply -f <dpa_filename>
```

10. Verify that the **DataProtectionApplication** CR is created successfully by running the following command:

```
$ oc get dpa -o yaml
```

Example output

```

apiVersion: v1
items:
- apiVersion: oadp.openshift.io/v1alpha1
  kind: DataProtectionApplication
  metadata:
    namespace: openshift-adp
    #...#
  spec:
    backupLocations:
    - velero:
        config:
          #...#
  status:
    conditions:
    - lastTransitionTime: "20....9:54:02Z"
      message: Reconcile complete
      reason: Complete
      status: "True"
      type: Reconciled
  kind: List
  metadata:
    resourceVersion: ""

```

11. Verify that the backup storage location (BSL) is available by running the following command:

```
$ oc get backupstoragelocations.velero.io -n openshift-adp
```

Example output

NAME	PHASE	LAST VALIDATED	AGE	DEFAULT
dpa-sample-1	Available	3s	15s	true

12. Configure the **Backup** CR by using the following example:

```

apiVersion: velero.io/v1
kind: Backup
metadata:
  name: test-backup
  namespace: openshift-adp
spec:
  includedNamespaces:
  - <application_namespace>

```

where:

<application_namespace>

Specifies the namespace for the application to back up.

13. Create the **Backup** CR by running the following command:

```
$ oc apply -f <backup_cr_filename>
```

Verification

- Verify that the **Backup** object is in the **Completed** phase by running the following command:

```
$ oc describe backup test-backup -n openshift-adp
```

Example output

```
Name:      test-backup
Namespace: openshift-adp
# ....#
Status:
  Backup Item Operations Attempted: 1
  Backup Item Operations Completed: 1
  Completion Timestamp:      2024-09-25T10:17:01Z
  Expiration:      2024-10-25T10:16:31Z
  Format Version:      1.1.0
  Hook Status:
  Phase: Completed
  Progress:
    Items Backed Up: 34
    Total Items: 34
  Start Timestamp: 2024-09-25T10:16:31Z
  Version: 1
  Events:      <none>
```

5.5.4. Using the legacy-aws Velero plugin

If you are using an AWS S3-compatible backup storage location, you might get a **SignatureDoesNotMatch** error while backing up your application. This error occurs because some backup storage locations still use the older versions of the S3 APIs, which are incompatible with the newer AWS SDK for Go V2. To resolve this issue, you can use the **legacy-aws** Velero plugin in the **DataProtectionApplication** custom resource (CR). The **legacy-aws** Velero plugin uses the older AWS SDK for Go V1, which is compatible with the legacy S3 APIs, ensuring successful backups.

5.5.4.1. Using the legacy-aws Velero plugin in the DataProtectionApplication CR

In the following use case, you configure the **DataProtectionApplication** CR with the **legacy-aws** Velero plugin and then back up an application.



NOTE

Depending on the backup storage location you choose, you can use either the **legacy-aws** or the **aws** plugin in your **DataProtectionApplication** CR. If you use both of the plugins in the **DataProtectionApplication** CR, the following error occurs: **aws and legacy-aws can not be both specified in DPA spec.configuration.velero.defaultPlugins.**

Prerequisites

- You have installed the OADP Operator.
- You have configured an AWS S3-compatible object storage as a backup location.

- You have an application with a database running in a separate namespace.

Procedure

1. Configure the **DataProtectionApplication** CR to use the **legacy-aws** Velero plugin as shown in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: oadp-backup
  namespace: openshift-adp
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
  velero:
    defaultPlugins:
      - legacy-aws
      - openshift
      - csi
    defaultSnapshotMoveData: true
  backupLocations:
    - velero:
        config:
          profile: "default"
          region: noobaa
          s3Url: https://s3.openshift-storage.svc
          s3ForcePathStyle: "true"
          insecureSkipTLSVerify: "true"
        provider: aws
        default: true
        credential:
          key: cloud
          name: cloud-credentials
        objectStorage:
          bucket: <bucket_name>
          prefix: oadp
```

where:

legacy-aws

Specifies to use the **legacy-aws** plugin.

<bucket_name>

Specifies the bucket name.

2. Create the **DataProtectionApplication** CR by running the following command:

```
$ oc apply -f <dpa_filename>
```

3. Verify that the **DataProtectionApplication** CR is created successfully by running the following command. In the example output, you can see the **status** object has the **type** field set to **Reconciled** and the **status** field set to **"True"**. That status indicates that the

DataProtectionApplication CR is successfully created.

```
$ oc get dpa -o yaml
```

Example output

```
apiVersion: v1
items:
- apiVersion: oadp.openshift.io/v1alpha1
  kind: DataProtectionApplication
  metadata:
    namespace: openshift-adp
    #...#
  spec:
    backupLocations:
    - velero:
        config:
          #...#
  status:
    conditions:
    - lastTransitionTime: "20....9:54:02Z"
      message: Reconcile complete
      reason: Complete
      status: "True"
      type: Reconciled
  kind: List
  metadata:
    resourceVersion: ""
```

- Verify that the backup storage location (BSL) is available by running the following command:

```
$ oc get backupstoragelocations.velero.io -n openshift-adp
```

You should see an output similar to the following example:

NAME	PHASE	LAST VALIDATED	AGE	DEFAULT
dpa-sample-1	Available	3s	15s	true

- Configure a **Backup** CR as shown in the following example:

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: test-backup
  namespace: openshift-adp
spec:
  includedNamespaces:
  - <application_namespace>
```

where:

<application_namespace>

Specifies the namespace for the application to back up.

6. Create the **Backup** CR by running the following command:

```
$ oc apply -f <backup_cr_filename>
```

Verification

- Verify that the backup object is in the **Completed** phase by running the following command. For more details, see the example output.

```
$ oc describe backups.velero.io test-backup -n openshift-adp
```

Example output

```
Name:      test-backup
Namespace: openshift-adp
# ...#
Status:
  Backup Item Operations Attempted: 1
  Backup Item Operations Completed: 1
  Completion Timestamp:      2024-09-25T10:17:01Z
  Expiration:                2024-10-25T10:16:31Z
  Format Version:            1.1.0
  Hook Status:
  Phase: Completed
  Progress:
    Items Backed Up: 34
    Total Items:    34
  Start Timestamp: 2024-09-25T10:16:31Z
  Version:        1
  Events:         <none>
```

5.5.5. Backing up workloads on OADP with OpenShift Container Platform

To back up and restore workloads on ROSA, you can use OADP. You can create a backup of a workload, restore it from the backup, and verify the restoration. You can also clean up the OADP Operator, backup storage, and AWS resources when they are no longer needed.

5.5.5.1. Performing a backup with OADP and OpenShift Container Platform

The following example **hello-world** application has no persistent volumes (PVs) attached. Perform a backup by using OpenShift API for Data Protection (OADP) with OpenShift Container Platform.

Either Data Protection Application (DPA) configuration will work.

Procedure

1. Create a workload to back up by running the following commands:

```
$ oc create namespace hello-world
```

```
$ oc new-app -n hello-world --image=docker.io/openshift/hello-openshift
```

2. Expose the route by running the following command:

```
$ oc expose service/hello-openshift -n hello-world
```

3. Check that the application is working by running the following command:

```
$ curl `oc get route/hello-openshift -n hello-world -o jsonpath='{.spec.host}'`
```

You should see an output similar to the following example:

```
Hello OpenShift!
```

4. Back up the workload by running the following command:

```
$ cat << EOF | oc create -f -
  apiVersion: velero.io/v1
  kind: Backup
  metadata:
    name: hello-world
    namespace: openshift-adp
  spec:
    includedNamespaces:
      - hello-world
    storageLocation: ${CLUSTER_NAME}-dpa-1
    ttl: 720h0m0s
EOF
```

5. Wait until the backup is complete, and then run the following command:

```
$ watch "oc -n openshift-adp get backup hello-world -o json | jq .status"
```

You should see an output similar to the following example:

```
{
  "completionTimestamp": "2022-09-07T22:20:44Z",
  "expiration": "2022-10-07T22:20:22Z",
  "formatVersion": "1.1.0",
  "phase": "Completed",
  "progress": {
    "itemsBackedUp": 58,
    "totalItems": 58
  },
  "startTimestamp": "2022-09-07T22:20:22Z",
  "version": 1
}
```

6. Delete the demo workload by running the following command:

```
$ oc delete ns hello-world
```

7. Restore the workload from the backup by running the following command:

```
$ cat << EOF | oc create -f -
```

```
apiVersion: velero.io/v1
kind: Restore
metadata:
  name: hello-world
  namespace: openshift-adp
spec:
  backupName: hello-world
EOF
```

8. Wait for the Restore to finish by running the following command:

```
$ watch "oc -n openshift-adp get restore hello-world -o json | jq .status"
```

You should see an output similar to the following example:

```
{
  "completionTimestamp": "2022-09-07T22:25:47Z",
  "phase": "Completed",
  "progress": {
    "itemsRestored": 38,
    "totalItems": 38
  },
  "startTimestamp": "2022-09-07T22:25:28Z",
  "warnings": 9
}
```

9. Check that the workload is restored by running the following command:

```
$ oc -n hello-world get pods
```

You should see an output similar to the following example:

```
NAME                                READY STATUS RESTARTS AGE
hello-openshift-9f885f7c6-kdjpi 1/1   Running 0      90s
```

10. Check the JSONPath by running the following command:

```
$ curl `oc get route/hello-openshift -n hello-world -o jsonpath='{.spec.host}'`
```

You should see an output similar to the following example:

```
Hello OpenShift!
```



NOTE

For troubleshooting tips, see the [troubleshooting documentation](#).

5.5.5.2. Cleaning up a cluster after a backup with OADP and ROSA STS

If you need to uninstall the OpenShift API for Data Protection (OADP) Operator together with the backups and the S3 bucket from this example, follow these instructions.

Procedure

1. Delete the workload by running the following command:

```
$ oc delete ns hello-world
```

2. Delete the Data Protection Application (DPA) by running the following command:

```
$ oc -n openshift-adp delete dpa ${CLUSTER_NAME}-dpa
```

3. Delete the cloud storage by running the following command:

```
$ oc -n openshift-adp delete cloudstorage ${CLUSTER_NAME}-oadp
```



WARNING

If this command hangs, you might need to delete the finalizer by running the following command:

```
$ oc -n openshift-adp patch cloudstorage ${CLUSTER_NAME}-oadp -p '{"metadata":{"finalizers":null}}' --type=merge
```

4. If the Operator is no longer required, remove it by running the following command:

```
$ oc -n openshift-adp delete subscription oadp-operator
```

5. Remove the namespace from the Operator:

```
$ oc delete ns openshift-adp
```

6. If the backup and restore resources are no longer required, remove them from the cluster by running the following command:

```
$ oc delete backups.velero.io hello-world
```

7. To delete backup, restore and remote objects in AWS S3 run the following command:

```
$ velero backup delete hello-world
```

8. If you no longer need the Custom Resource Definitions (CRD), remove them from the cluster by running the following command:

```
$ for CRD in `oc get crds | grep velero | awk '{print $1}'`; do oc delete crd $CRD; done
```

9. Delete the AWS S3 bucket by running the following commands:

```
$ aws s3 rm s3://${CLUSTER_NAME}-oadp --recursive
```

```
$ aws s3api delete-bucket --bucket ${CLUSTER_NAME}-oadp
```

10. Detach the policy from the role by running the following command:

```
$ aws iam detach-role-policy --role-name "${ROLE_NAME}" --policy-arn "${POLICY_ARN}"
```

11. Delete the role by running the following command:

```
$ aws iam delete-role --role-name "${ROLE_NAME}"
```

5.6. INSTALLING OADP

5.6.1. About installing OADP

As a cluster administrator, you install the OpenShift API for Data Protection (OADP) by installing the OADP Operator. The OADP Operator installs [Velero 1.16](#).



NOTE

Starting from OADP 1.0.4, all OADP 1.0.z versions can only be used as a dependency of the Migration Toolkit for Containers Operator and are not available as a standalone Operator.

To back up Kubernetes resources and internal images, you must have object storage as a backup location, such as one of the following storage types:

- [Amazon Web Services](#)
- [Microsoft Azure](#)
- [Google Cloud](#)
- [Multicloud Object Gateway](#)
- IBM Cloud® Object Storage S3
- AWS S3 compatible object storage, such as Multicloud Object Gateway or MinIO

You can configure multiple backup storage locations within the same namespace for each individual OADP deployment.



NOTE

Unless specified otherwise, "NooBaa" refers to the open source project that provides lightweight object storage, while "Multicloud Object Gateway (MCG)" refers to the Red Hat distribution of NooBaa.

For more information on the MCG, see [Accessing the Multicloud Object Gateway with your applications](#).

You can back up persistent volumes (PVs) by using snapshots or a File System Backup (FSB).

To back up PVs with snapshots, you must have a cloud provider that supports either a native snapshot API or Container Storage Interface (CSI) snapshots, such as one of the following cloud providers:

- [Amazon Web Services](#)
- [Microsoft Azure](#)
- [Google Cloud](#)
- CSI snapshot-enabled cloud provider, such as [OpenShift Data Foundation](#)



NOTE

If you want to use CSI backup on OCP 4.11 and later, install OADP 1.1.x.

OADP 1.0.x does not support CSI backup on OCP 4.11 and later. OADP 1.0. x includes Velero 1.7.x and expects the API group **snapshot.storage.k8s.io/v1beta1**, which is not present on OCP 4.11 and later.

If your cloud provider does not support snapshots or if your storage is NFS, you can back up applications with [Backing up applications with File System Backup: Kopia or Restic](#) on object storage.

You create a default **Secret** and then you install the Data Protection Application.

5.6.1.1. AWS S3 compatible backup storage providers

OADP works with many S3-compatible object storage providers. Several object storage providers are certified and tested with every release of OADP. Various S3 providers are known to work with OADP but are not specifically tested and certified. These providers will be supported on a best-effort basis. Additionally, there are a few S3 object storage providers with known issues and limitations that are listed in this documentation.



NOTE

Red Hat will provide support for OADP on any S3-compatible storage, but support will stop if the S3 endpoint is determined to be the root cause of an issue.

5.6.1.1.1. Certified backup storage providers

The following AWS S3 compatible object storage providers are fully supported by OADP through the AWS plugin for use as backup storage locations:

- MinIO
- Multicloud Object Gateway (MCG)
- Amazon Web Services (AWS) S3
- IBM Cloud® Object Storage S3
- Ceph RADOS Gateway (Ceph Object Gateway)
- Red Hat Container Storage
- Red Hat OpenShift Data Foundation

- NetApp ONTAP S3 Object Storage
- [Scality ARTESCA S3 object storage](#)



NOTE

The following compatible object storage providers are supported and have their own Velero object store plugins:

- Google Cloud
- Microsoft Azure

5.6.1.1.2. Unsupported backup storage providers

The following AWS S3 compatible object storage providers, are known to work with Velero through the AWS plugin, for use as backup storage locations, however, they are unsupported and have not been tested by Red Hat:

- Oracle Cloud
- DigitalOcean
- NooBaa, unless installed using Multicloud Object Gateway (MCG)
- Tencent Cloud
- Ceph RADOS v12.2.7
- Quobyte
- Cloudian HyperStore



NOTE

Unless specified otherwise, "NooBaa" refers to the open source project that provides lightweight object storage, while "Multicloud Object Gateway (MCG)" refers to the Red Hat distribution of NooBaa.

For more information on the MCG, see [Accessing the Multicloud Object Gateway with your applications](#).

5.6.1.1.3. Backup storage providers with known limitations

The following AWS S3 compatible object storage providers are known to work with Velero through the AWS plugin with a limited feature set:

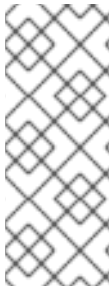
- Swift - It works for use as a backup storage location for backup storage, but is not compatible with Restic for filesystem-based volume backup and restore.

5.6.1.2. Configuring Multicloud Object Gateway (MCG) for disaster recovery on OpenShift Data Foundation

If you use cluster storage for your MCG bucket **backupStorageLocation** on OpenShift Data Foundation, configure MCG as an external object store.

**WARNING**

Failure to configure MCG as an external object store might lead to backups not being available.

**NOTE**

Unless specified otherwise, "NooBaa" refers to the open source project that provides lightweight object storage, while "Multicloud Object Gateway (MCG)" refers to the Red Hat distribution of NooBaa.

For more information on the MCG, see [Accessing the Multicloud Object Gateway with your applications](#).

Procedure

- Configure MCG as an external object store as described in [Adding storage resources for hybrid or Multicloud](#).

Additional resources

- [Overview of backup and snapshot locations in the Velero documentation](#)

5.6.1.3. About OADP update channels

When you install an OADP Operator, you choose an *update channel*. This channel determines which upgrades to the OADP Operator and to Velero you receive. You can switch channels at any time.

The following update channels are available:

- The **stable** channel is now deprecated. The **stable** channel contains the patches (z-stream updates) of OADP **ClusterServiceVersion** for **OADP.v1.1.z** and older versions from **OADP.v1.0.z**.
- The **stable-1.0** channel is deprecated and is not supported.
- The **stable-1.1** channel is deprecated and is not supported.
- The **stable-1.2** channel is deprecated and is not supported.
- The **stable-1.3** channel contains **OADP.v1.3.z**, the most recent OADP 1.3 **ClusterServiceVersion**.
- The **stable-1.4** channel contains **OADP.v1.4.z**, the most recent OADP 1.4 **ClusterServiceVersion**.

For more information, see [OpenShift Operator Life Cycles](#).

Which update channel is right for you?

- The **stable** channel is now deprecated. If you are already using the stable channel, you will continue to get updates from **OADP.v1.1.z**.
- Choose the **stable-1.y** update channel to install OADP 1.y and to continue receiving patches for it. If you choose this channel, you will receive all z-stream patches for version 1.y.z.

When must you switch update channels?

- If you have OADP 1.y installed, and you want to receive patches only for that y-stream, you must switch from the **stable** update channel to the **stable-1.y** update channel. You will then receive all z-stream patches for version 1.y.z.
- If you have OADP 1.0 installed, want to upgrade to OADP 1.1, and then receive patches only for OADP 1.1, you must switch from the **stable-1.0** update channel to the **stable-1.1** update channel. You will then receive all z-stream patches for version 1.1.z.
- If you have OADP 1.y installed, with y greater than 0, and want to switch to OADP 1.0, you must uninstall your OADP Operator and then reinstall it using the **stable-1.0** update channel. You will then receive all z-stream patches for version 1.0.z.



NOTE

You cannot switch from OADP 1.y to OADP 1.0 by switching update channels. You must uninstall the Operator and then reinstall it.

5.6.1.4. Installation of OADP on multiple namespaces

You can install OpenShift API for Data Protection into multiple namespaces on the same cluster so that multiple project owners can manage their own OADP instance. This use case has been validated with File System Backup (FSB) and Container Storage Interface (CSI).

You install each instance of OADP as specified by the per-platform procedures contained in this document with the following additional requirements:

- All deployments of OADP on the same cluster must be the same version, for example, 1.4.0. Installing different versions of OADP on the same cluster is **not** supported.
- Each individual deployment of OADP must have a unique set of credentials and at least one **BackupStorageLocation** configuration. You can also use multiple **BackupStorageLocation** configurations within the same namespace.
- By default, each OADP deployment has cluster-level access across namespaces. OpenShift Container Platform administrators need to carefully review potential impacts, such as not backing up and restoring to and from the same namespace concurrently.

5.6.1.5. OADP support for backup data immutability

Starting with OADP 1.4, you can store OADP backups in an AWS S3 bucket with enabled versioning. The versioning support is only for AWS S3 buckets and not for S3-compatible buckets.

See the following list for specific cloud provider limitations:

- AWS S3 service supports backups because an S3 object lock applies only to versioned buckets. You can still update the object data for the new version. However, when backups are deleted, old versions of the objects are not deleted.

- OADP backups are not supported and might not work as expected when you enable immutability on Azure Storage Blob.
- Google Cloud storage policy only supports bucket-level immutability. Therefore, it is not feasible to implement it in the Google Cloud environment.

Depending on your storage provider, the immutability options are called differently:

- S3 object lock
- Object retention
- Bucket versioning
- Write Once Read Many (WORM) buckets

The primary reason for the absence of support for other S3-compatible object storage is that OADP initially saves the state of a backup as *finalizing* and then verifies whether any asynchronous operations are in progress.

Additional resources

- [Cluster service version](#)

5.6.1.6. Velero CPU and memory requirements based on collected data

The following recommendations are based on observations of performance made in the scale and performance lab. The backup and restore resources can be impacted by the type of plugin, the amount of resources required by that backup or restore, and the respective data contained in the persistent volumes (PVs) related to those resources.

5.6.1.6.1. CPU and memory requirement for configurations

Configuration types	[1] Average usage	[2] Large usage	resourceTimeouts
CSI	Velero: CPU- Request 200m, Limits 1000m Memory - Request 256Mi, Limits 1024Mi	Velero: CPU- Request 200m, Limits 2000m Memory- Request 256Mi, Limits 2048Mi	N/A
Restic	[3] Restic: CPU- Request 1000m, Limits 2000m Memory - Request 16Gi, Limits 32Gi	[4] Restic: CPU - Request 2000m, Limits 8000m Memory - Request 16Gi, Limits 40Gi	900m
[5] Data Mover	N/A	N/A	10m - average usage 60m - large usage

1. Average usage - use these settings for most usage situations.
2. Large usage - use these settings for large usage situations, such as a large PV (500GB Usage), multiple namespaces (100+), or many pods within a single namespace (2000 pods+), and for optimal performance for backup and restore involving large datasets.
3. Restic resource usage corresponds to the amount of data, and type of data. For example, many small files or large amounts of data can cause Restic to use large amounts of resources. The [Velero](#) documentation references 500m as a supplied default, for most of our testing we found a 200m request suitable with 1000m limit. As cited in the Velero documentation, exact CPU and memory usage is dependent on the scale of files and directories, in addition to environmental limitations.
4. Increasing the CPU has a significant impact on improving backup and restore times.
5. Data Mover - Data Mover default resourceTimeout is 10m. Our tests show that for restoring a large PV (500GB usage), it is required to increase the resourceTimeout to 60m.



NOTE

The resource requirements listed throughout the guide are for average usage only. For large usage, adjust the settings as described in the table above.

5.6.1.6.2. NodeAgent CPU for large usage

Testing shows that increasing **NodeAgent** CPU can significantly improve backup and restore times when using OpenShift API for Data Protection (OADP).



IMPORTANT

You can tune your OpenShift Container Platform environment based on your performance analysis and preference. Use CPU limits in the workloads when you use Kopia for file system backups.

If you do not use CPU limits on the pods, the pods can use excess CPU when it is available. If you specify CPU limits, the pods might be throttled if they exceed their limits. Therefore, the use of CPU limits on the pods is considered an anti-pattern.

Ensure that you are accurately specifying CPU requests so that pods can take advantage of excess CPU. Resource allocation is guaranteed based on CPU requests rather than CPU limits.

Testing showed that running Kopia with 20 cores and 32 Gi memory supported backup and restore operations of over 100 GB of data, multiple namespaces, or over 2000 pods in a single namespace. Testing detected no CPU limiting or memory saturation with these resource specifications.

In some environments, you might need to adjust Ceph MDS pod resources to avoid pod restarts, which occur when default settings cause resource saturation.

For more information about how to set the pod resources limit in Ceph MDS pods, see [Changing the CPU and memory resources on the rook-ceph pods](#).

5.6.2. Installing the OADP Operator

You can install the OpenShift API for Data Protection (OADP) Operator on OpenShift Container Platform 4.20 by using Operator Lifecycle Manager (OLM).

The OADP Operator installs [Velero 1.16](#).

5.6.2.1. Installing the OADP Operator

Use the following procedure to install the OADP Operator.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.

Procedure

- In the OpenShift Container Platform web console, click **Ecosystem → Software Catalog**.
- Use the **Filter by keyword** field to find the **OADP Operator**.
- Select the **OADP Operator** and click **Install**.
- Click **Install** to install the Operator in the **openshift-adp** project.
- Click **Ecosystem → Installed Operators** to verify the installation.

5.6.2.2. OADP-Velero-OpenShift Container Platform version relationship

OADP version	Velero version	OpenShift Container Platform version
1.3.0	{velero-1.12}	4.12-4.15
1.3.1	{velero-1.12}	4.12-4.15
1.3.2	{velero-1.12}	4.12-4.15
1.3.3	{velero-1.12}	4.12-4.15
1.3.4	{velero-1.12}	4.12-4.15
1.3.5	{velero-1.12}	4.12-4.15
1.4.0	{velero-1.14}	4.14-4.18
1.4.1	{velero-1.14}	4.14-4.18
1.4.2	{velero-1.14}	4.14-4.18
1.4.3	{velero-1.14}	4.14-4.18

OADP version	Velero version	OpenShift Container Platform version
1.5.0	{velero-1.16}	4.19

5.7. CONFIGURING OADP WITH AWS S3 COMPATIBLE STORAGE

5.7.1. Configuring the OpenShift API for Data Protection with AWS S3 compatible storage

You install the OpenShift API for Data Protection (OADP) with Amazon Web Services (AWS) S3 compatible storage by installing the OADP Operator. The Operator installs [Velero 1.16](#).



NOTE

Starting from OADP 1.0.4, all OADP 1.0.z versions can only be used as a dependency of the Migration Toolkit for Containers Operator and are not available as a standalone Operator.

You configure AWS for Velero, create a default **Secret**, and then install the Data Protection Application. For more details, see [Installing the OADP Operator](#).

To install the OADP Operator in a restricted network environment, you must first disable the default software catalog sources and mirror the Operator catalog. See [Using Operator Lifecycle Manager in disconnected environments](#) for details.

5.7.1.1. About Amazon Simple Storage Service, Identity and Access Management, and GovCloud

Review Amazon Simple Storage Service (S3), Identity and Access Management (IAM), and AWS GovCloud requirements to configure backup storage with appropriate security controls. This helps you meet federal data security requirements and use correct endpoints.

AWS S3 is a storage solution of Amazon for the internet. As an authorized user, you can use this service to store and retrieve any amount of data whenever you want, from anywhere on the web.

You securely control access to Amazon S3 and other Amazon services by using the AWS Identity and Access Management (IAM) web service.

You can use IAM to manage permissions that control which AWS resources users can access. You use IAM to both authenticate, or verify that a user is who they claim to be, and to authorize, or grant permissions to use resources.

AWS GovCloud (US) is an Amazon storage solution developed to meet the stringent and specific data security requirements of the United States Federal Government. AWS GovCloud (US) works the same as Amazon S3 except for the following:

- You cannot copy the contents of an Amazon S3 bucket in the AWS GovCloud (US) regions directly to or from another AWS region.

- If you use Amazon S3 policies, use the AWS GovCloud (US) Amazon Resource Name (ARN) identifier to unambiguously specify a resource across all of AWS, such as in IAM policies, Amazon S3 bucket names, and API calls.
 - In AWS GovCloud (US) regions, ARNs have an identifier that is different from the one in other standard AWS regions, **arn:aws-us-gov**. If you need to specify the US-West or US-East region, use one of the following ARNs:
 - For US-West, use **us-gov-west-1**.
 - For US-East, use **us-gov-east-1**.
 - For all other standard regions, ARNs begin with: **arn:aws**.
- In AWS GovCloud (US) regions, use the endpoints listed in the **AWS GovCloud (US-East)** and **AWS GovCloud (US-West)** rows of the "Amazon S3 endpoints" table on [Amazon Simple Storage Service endpoints and quotas](#). If you are processing export-controlled data, use one of the SSL/TLS endpoints. If you have FIPS requirements, use a FIPS 140-2 endpoint such as <https://s3-fips.us-gov-west-1.amazonaws.com> or <https://s3-fips.us-gov-east-1.amazonaws.com>.
- To find the other AWS-imposed restrictions, see [How Amazon Simple Storage Service Differs for AWS GovCloud \(US\)](#).

5.7.1.2. Configuring Amazon Web Services

Configure Amazon Web Services (AWS) S3 storage and Identity and Access Management (IAM) credentials for backup storage with OADP. This provides the necessary permissions and storage infrastructure for data protection operations.

Prerequisites

- You must have the [AWS CLI](#) installed.

Procedure

1. Set the **BUCKET** variable:

```
$ BUCKET=<your_bucket>
```

2. Set the **REGION** variable:

```
$ REGION=<your_region>
```

3. Create an AWS S3 bucket:

```
$ aws s3api create-bucket \
  --bucket $BUCKET \
  --region $REGION \
  --create-bucket-configuration LocationConstraint=$REGION
```

where:

LocationConstraint

Specifies the bucket configuration location constraint. **us-east-1** does not support **LocationConstraint**. If your region is **us-east-1**, omit **--create-bucket-configuration LocationConstraint=\$REGION**.

4. Create an IAM user:

```
$ aws iam create-user --user-name velero
```

where:

velero

Specifies the user name. If you want to use Velero to back up multiple clusters with multiple S3 buckets, create a unique user name for each cluster.

5. Create a **velero-policy.json** file:

```
$ cat > velero-policy.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeVolumes",
        "ec2:DescribeSnapshots",
        "ec2:CreateTags",
        "ec2:CreateVolume",
        "ec2:CreateSnapshot",
        "ec2>DeleteSnapshot"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3>DeleteObject",
        "s3:PutObject",
        "s3:AbortMultipartUpload",
        "s3:ListMultipartUploadParts"
      ],
      "Resource": [
        "arn:aws:s3:::${BUCKET}/*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket",
        "s3:GetBucketLocation",
        "s3:ListBucketMultipartUploads"
      ],
      "Resource": [
        "arn:aws:s3:::${BUCKET}"
      ]
    }
  ]
}
```

```
}
]
}
EOF
```

6. Attach the policies to give the **velero** user the minimum necessary permissions:

```
$ aws iam put-user-policy \
  --user-name velero \
  --policy-name velero \
  --policy-document file:///velero-policy.json
```

7. Create an access key for the **velero** user:

```
$ aws iam create-access-key --user-name velero

{
  "AccessKey": {
    "UserName": "velero",
    "Status": "Active",
    "CreateDate": "2017-07-31T22:24:41.576Z",
    "SecretAccessKey": <AWS_SECRET_ACCESS_KEY>,
    "AccessKeyId": <AWS_ACCESS_KEY_ID>
  }
}
```

8. Create a **credentials-velero** file:

```
$ cat << EOF > ./credentials-velero
[default]
aws_access_key_id=<AWS_ACCESS_KEY_ID>
aws_secret_access_key=<AWS_SECRET_ACCESS_KEY>
EOF
```

You use the **credentials-velero** file to create a **Secret** object for AWS before you install the Data Protection Application.

5.7.1.3. About backup and snapshot locations and their secrets

Review backup location, snapshot location, and secret configuration requirements for the **DataProtectionApplication** custom resource (CR). This helps you understand storage options and credential management for data protection operations.

5.7.1.3.1. Backup locations

You can specify one of the following AWS S3-compatible object storage solutions as a backup location:

- Multicloud Object Gateway (MCG)
- Red Hat Container Storage
- Ceph RADOS Gateway; also known as Ceph Object Gateway
- Red Hat OpenShift Data Foundation

- MinIO

Velero backs up OpenShift Container Platform resources, Kubernetes objects, and internal images as an archive file on object storage.

5.7.1.3.2. Snapshot locations

If you use your cloud provider's native snapshot API to back up persistent volumes, you must specify the cloud provider as the snapshot location.

If you use Container Storage Interface (CSI) snapshots, you do not need to specify a snapshot location because you will create a **VolumeSnapshotClass** CR to register the CSI driver.

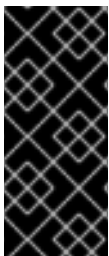
If you use File System Backup (FSB), you do not need to specify a snapshot location because FSB backs up the file system on object storage.

5.7.1.3.3. Secrets

If the backup and snapshot locations use the same credentials or if you do not require a snapshot location, you create a default **Secret**.

If the backup and snapshot locations use different credentials, you create two secret objects:

- Custom **Secret** for the backup location, which you specify in the **DataProtectionApplication** CR.
- Default **Secret** for the snapshot location, which is not referenced in the **DataProtectionApplication** CR.



IMPORTANT

The Data Protection Application requires a default **Secret**. Otherwise, the installation will fail.

If you do not want to specify backup or snapshot locations during the installation, you can create a default **Secret** with an empty **credentials-velero** file.

5.7.1.3.4. Creating a default Secret

You create a default **Secret** if your backup and snapshot locations use the same credentials or if you do not require a snapshot location.

The default name of the **Secret** is **cloud-credentials**.



NOTE

The **DataProtectionApplication** custom resource (CR) requires a default **Secret**. Otherwise, the installation will fail. If the name of the backup location **Secret** is not specified, the default name is used.

If you do not want to use the backup location credentials during the installation, you can create a **Secret** with the default name by using an empty **credentials-velero** file.

Prerequisites

- Your object storage and cloud storage, if any, must use the same credentials.
- You must configure object storage for Velero.

Procedure

1. Create a **credentials-velero** file for the backup storage location in the appropriate format for your cloud provider.
See the following example:

```
[default]
aws_access_key_id=<AWS_ACCESS_KEY_ID>
aws_secret_access_key=<AWS_SECRET_ACCESS_KEY>
```

2. Create a **Secret** custom resource (CR) with the default name:

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file cloud=credentials-velero
```

The **Secret** is referenced in the **spec.backupLocations.credential** block of the **DataProtectionApplication** CR when you install the Data Protection Application.

5.7.1.3.5. Creating profiles for different credentials

If your backup and snapshot locations use different credentials, you create separate profiles in the **credentials-velero** file.

Then, you create a **Secret** object and specify the profiles in the **DataProtectionApplication** custom resource (CR).

Procedure

1. Create a **credentials-velero** file with separate profiles for the backup and snapshot locations, as in the following example:

```
[backupStorage]
aws_access_key_id=<AWS_ACCESS_KEY_ID>
aws_secret_access_key=<AWS_SECRET_ACCESS_KEY>

[volumeSnapshot]
aws_access_key_id=<AWS_ACCESS_KEY_ID>
aws_secret_access_key=<AWS_SECRET_ACCESS_KEY>
```

2. Create a **Secret** object with the **credentials-velero** file:

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file cloud=credentials-velero 1
```

3. Add the profiles to the **DataProtectionApplication** CR, as in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
```

```

name: <dpa_sample>
namespace: openshift-adp
spec:
...
backupLocations:
- name: default
  velero:
    provider: aws
    default: true
    objectStorage:
      bucket: <bucket_name>
      prefix: <prefix>
    config:
      region: us-east-1
      profile: "backupStorage"
    credential:
      key: cloud
      name: cloud-credentials
snapshotLocations:
- velero:
    provider: aws
    config:
      region: us-west-2
      profile: "volumeSnapshot"

```

5.7.1.3.6. Creating an OADP SSE-C encryption key for additional data security

Configure server-side encryption with customer-provided keys (SSE-C) to add an additional layer of encryption for backup data stored in Amazon Web Services (AWS) S3. This protects backup data if AWS credentials become exposed.

Amazon Web Services (AWS) S3 applies server-side encryption with AWS S3 managed keys (SSE-S3) as the base level of encryption for every bucket in Amazon S3.

OpenShift API for Data Protection (OADP) encrypts data by using SSL/TLS, HTTPS, and the **velero-repo-credentials** secret when transferring the data from a cluster to storage. To protect backup data in case of lost or stolen AWS credentials, apply an additional layer of encryption.

The velero-plugin-for-aws plugin provides several additional encryption methods. You should review its configuration options and consider implementing additional encryption.

You can store your own encryption keys by using server-side encryption with customer-provided keys (SSE-C). This feature provides additional security if your AWS credentials become exposed.



WARNING

Be sure to store cryptographic keys in a secure and safe manner. Encrypted data and backups cannot be recovered if you do not have the encryption key.

Prerequisites

- To make OADP mount a secret that contains your SSE-C key to the Velero pod at **/credentials**, use the following default secret name for AWS: **cloud-credentials**, and leave at least one of the following labels empty:
 - **dpa.spec.backupLocations[].velero.credential**
 - **dpa.spec.snapshotLocations[].velero.credential**
 This is a workaround for a known issue: <https://issues.redhat.com/browse/OADP-3971>.



NOTE

The following procedure contains an example of a **spec:backupLocations** block that does not specify credentials. This example would trigger an OADP secret mounting.

- If you need the backup location to have credentials with a different name than **cloud-credentials**, you must add a snapshot location, such as the one in the following example, that does not contain a credential name. Because the following example does not contain a credential name, the snapshot location will use **cloud-credentials** as its secret for taking snapshots.

```
snapshotLocations:
- velero:
  config:
    profile: default
    region: <region>
    provider: aws
# ...
```

Procedure

1. Create an SSE-C encryption key:
 - a. Generate a random number and save it as a file named **sse.key** by running the following command:

```
$ dd if=/dev/urandom bs=1 count=32 > sse.key
```

2. Create an OpenShift Container Platform secret:

- If you are initially installing and configuring OADP, create the AWS credential and encryption key secret at the same time by running the following command:

```
$ oc create secret generic cloud-credentials --namespace openshift-adp --from-file
cloud=<path>/openshift_aws_credentials,customer-key=<path>/sse.key
```

- If you are updating an existing installation, edit the values of the **cloud-credential secret** block of the **DataProtectionApplication** CR manifest, as in the following example:

```
apiVersion: v1
data:
  cloud:
    W2Rfa2V5X2IkPSJBS0IBVkJRWUlyRkQ0TIFHRFFPQiIKYXdzX3NIY3JldF9hY2Nlc3Nfa2V
    5P<snip>rUE1mNWVSbTN5K2FpeWhUTUQyQk1WZHBOlgo=
    customer-key: v+<snip>TFliq6aaXPbj8dhos=
```

```
kind: Secret
# ...
```

3. Edit the value of the **customerKeyEncryptionFile** attribute in the **backupLocations** block of the **DataProtectionApplication** CR manifest, as in the following example:

```
spec:
  backupLocations:
    - velero:
      config:
        customerKeyEncryptionFile: /credentials/customer-key
        profile: default
# ...
```



WARNING

You must restart the Velero pod to remount the secret credentials properly on an existing installation.

The installation is complete, and you can back up and restore OpenShift Container Platform resources. The data saved in AWS S3 storage is encrypted with the new key, and you cannot download it from the AWS S3 console or API without the additional encryption key.

Verification

To verify that you cannot download the encrypted files without the inclusion of an additional key, create a test file, upload it, and then try to download it.

1. Create a test file by running the following command:

```
$ echo "encrypt me please" > test.txt
```

2. Upload the test file by running the following command:

```
$ aws s3api put-object \
  --bucket <bucket> \
  --key test.txt \
  --body test.txt \
  --sse-customer-key file:///sse.key \
  --sse-customer-algorithm AES256
```

3. Try to download the file. In either the Amazon web console or the terminal, run the following command:

```
$ s3cmd get s3://<bucket>/test.txt test.txt
```

The download fails because the file is encrypted with an additional key.

4. Download the file with the additional encryption key by running the following command:

—

```
$ aws s3api get-object \
  --bucket <bucket> \
  --key test.txt \
  --sse-customer-key fileb://sse.key \
  --sse-customer-algorithm AES256 \
  downloaded.txt
```

5. Read the file contents by running the following command:

```
$ cat downloaded.txt
```

```
encrypt me please
```

5.7.1.3.6.1. Downloading a file with an SSE-C encryption key for files backed up by Velero

When you are verifying an SSE-C encryption key, you can also download the file with the additional encryption key for files that were backed up with Velero.

Procedure

- Download the file with the additional encryption key for files backed up by Velero by running the following command:

```
$ aws s3api get-object \
  --bucket <bucket> \
  --key velero/backups/mysql-persistent-customerkeyencryptionfile4/mysql-persistent-
customerkeyencryptionfile4.tar.gz \
  --sse-customer-key fileb://sse.key \
  --sse-customer-algorithm AES256 \
  --debug \
  velero_download.tar.gz
```

5.7.1.4. Installing the Data Protection Application

You install the Data Protection Application (DPA) by creating an instance of the **DataProtectionApplication** API.

Prerequisites

- You must install the OADP Operator.
- You must configure object storage as a backup location.
- If you use snapshots to back up PVs, your cloud provider must support either a native snapshot API or Container Storage Interface (CSI) snapshots.
- If the backup and snapshot locations use the same credentials, you must create a **Secret** with the default name, **cloud-credentials**.
- If the backup and snapshot locations use different credentials, you must create a **Secret** with the default name, **cloud-credentials**, which contains separate profiles for the backup and snapshot location credentials.



NOTE

If you do not want to specify backup or snapshot locations during the installation, you can create a default **Secret** with an empty **credentials-velero** file. If there is no default **Secret**, the installation will fail.

Procedure

1. Click **Ecosystem** → **Installed Operators** and select the OADP Operator.
2. Under **Provided APIs**, click **Create instance** in the **DataProtectionApplication** box.
3. Click **YAML View** and update the parameters of the **DataProtectionApplication** manifest:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - aws
      resourceTimeout: 10m
  nodeAgent:
    enable: true
    uploaderType: kopia
    podConfig:
      nodeSelector: <node_selector>
  backupLocations:
    - name: default
      velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket_name>
          prefix: <prefix>
        config:
          region: <region>
          profile: "default"
          s3ForcePathStyle: "true"
          s3Url: <s3_url>
        credential:
          key: cloud
          name: cloud-credentials
  snapshotLocations:
    - name: default
      velero:
        provider: aws
        config:
          region: <region>
          profile: "default"
```

```
credential:  
  key: cloud  
  name: cloud-credentials
```

where:

namespace

Specifies the default namespace for OADP which is **openshift-adp**. The namespace is a variable and is configurable.

openshift

Specifies that the **openshift** plugin is mandatory.

resourceTimeout

Specifies how many minutes to wait for several Velero resources such as Velero CRD availability, volumeSnapshot deletion, and backup repository availability, before timeout occurs. The default is 10m.

nodeAgent

Specifies the administrative agent that routes the administrative requests to servers.

enable

Set this value to **true** if you want to enable **nodeAgent** and perform File System Backup.

uploaderType

Specifies the uploader type. Enter **kopia** or **restic** as your uploader. You cannot change the selection after the installation. For the Built-in DataMover you must use Kopia. The **nodeAgent** deploys a daemon set, which means that the **nodeAgent** pods run on each working node. You can configure File System Backup by adding **spec.defaultVolumesToFsBackup: true** to the **Backup** CR.

nodeSelector

Specifies the nodes on which Kopia or Restic are available. By default, Kopia or Restic run on all nodes.

bucket

Specifies a bucket as the backup storage location. If the bucket is not a dedicated bucket for Velero backups, you must specify a prefix.

prefix

Specifies a prefix for Velero backups, for example, **velero**, if the bucket is used for multiple purposes.

s3ForcePathStyle

Specifies whether to force path style URLs for S3 objects (Boolean). Not Required for AWS S3. Required only for S3 compatible storage.

s3Url

Specifies the URL of the object store that you are using to store backups. Not required for AWS S3. Required only for S3 compatible storage.

name

Specifies the name of the **Secret** object that you created. If you do not specify this value, the default name, **cloud-credentials**, is used. If you specify a custom name, the custom name is used for the backup location.

snapshotLocations

Specifies a snapshot location, unless you use CSI snapshots or a File System Backup (FSB) to back up PVs.

region

Specifies that the snapshot location must be in the same region as the PVs.

name

Specifies the name of the **Secret** object that you created. If you do not specify this value, the default name, **cloud-credentials**, is used. If you specify a custom name, the custom name is used for the snapshot location. If your backup and snapshot locations use different credentials, create separate profiles in the **credentials-velero** file.

4. Click **Create**.

Verification

1. Verify the installation by viewing the OpenShift API for Data Protection (OADP) resources by running the following command:

```
$ oc get all -n openshift-adp
```

```
NAME                                READY STATUS  RESTARTS  AGE
pod/oadp-operator-controller-manager-67d9494d47-6l8z8  2/2   Running  0         2m8s
pod/node-agent-9cq4q                                1/1   Running  0         94s
pod/node-agent-m4lts                                1/1   Running  0         94s
pod/node-agent-pv4kr                                1/1   Running  0         95s
pod/velero-588db7f655-n842v                        1/1   Running  0         95s
```

```
NAME                                TYPE          CLUSTER-IP    EXTERNAL-IP
PORT(S)  AGE
service/oadp-operator-controller-manager-metrics-service  ClusterIP    172.30.70.140
<none>      8443/TCP  2m8s
service/openshift-adp-velero-metrics-svc                  ClusterIP    172.30.10.0   <none>
8085/TCP  8h
```

```
NAME                                DESIRED  CURRENT  READY  UP-TO-DATE  AVAILABLE  NODE
SELECTOR  AGE
daemonset.apps/node-agent  3        3        3      3           3          <none>    96s
```

```
NAME                                READY  UP-TO-DATE  AVAILABLE  AGE
deployment.apps/oadp-operator-controller-manager  1/1    1           1          2m9s
deployment.apps/velero                            1/1    1           1          96s
```

```
NAME                                DESIRED  CURRENT  READY  AGE
replicaset.apps/oadp-operator-controller-manager-67d9494d47  1        1        1      2m9s
replicaset.apps/velero-588db7f655                        1        1        1      96s
```

2. Verify that the **DataProtectionApplication** (DPA) is reconciled by running the following command:

```
$ oc get dpa dpa-sample -n openshift-adp -o jsonpath='{.status}'
```

```
{"conditions":[{"lastTransitionTime":"2023-10-27T01:23:57Z","message":"Reconcile complete","reason":"Complete","status":"True","type":"Reconciled"]}]}
```

3. Verify the **type** is set to **Reconciled**.
4. Verify the backup storage location and confirm that the **PHASE** is **Available** by running the following command:

```
$ oc get backupstoragelocations.velero.io -n openshift-adp
```

```
NAME          PHASE    LAST VALIDATED  AGE    DEFAULT
dpa-sample-1  Available 1s             3d16h true
```

5.7.1.4.1. Setting Velero CPU and memory resource allocations

You set the CPU and memory resource allocations for the **Velero** pod by editing the **DataProtectionApplication** custom resource (CR) manifest.

Prerequisites

- You must have the OpenShift API for Data Protection (OADP) Operator installed.

Procedure

- Edit the values in the **spec.configuration.velero.podConfig.ResourceAllocations** block of the **DataProtectionApplication** CR manifest, as in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  # ...
  configuration:
    velero:
      podConfig:
        nodeSelector: <node_selector>
        resourceAllocations:
          limits:
            cpu: "1"
            memory: 1024Mi
          requests:
            cpu: 200m
            memory: 256Mi
```

where:

nodeSelector

Specifies the node selector to be supplied to Velero podSpec.

resourceAllocations

Specifies the resource allocations listed for average usage.



NOTE

Kopia is an option in OADP 1.3 and later releases. You can use Kopia for file system backups, and Kopia is your only option for Data Mover cases with the built-in Data Mover.

Kopia is more resource intensive than Restic, and you might need to adjust the CPU and memory requirements accordingly.

Use the **nodeSelector** field to select which nodes can run the node agent. The **nodeSelector** field is the simplest recommended form of node selection constraint. Any label specified must match the labels on each node.

5.7.1.4.2. Enabling self-signed CA certificates

You must enable a self-signed CA certificate for object storage by editing the **DataProtectionApplication** custom resource (CR) manifest to prevent a **certificate signed by unknown authority** error.

Prerequisites

- You must have the OpenShift API for Data Protection (OADP) Operator installed.

Procedure

- Edit the **spec.backupLocations.velero.objectStorage.caCert** parameter and **spec.backupLocations.velero.config** parameters of the **DataProtectionApplication** CR manifest:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  # ...
  backupLocations:
    - name: default
      velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket>
          prefix: <prefix>
          caCert: <base64_encoded_cert_string>
        config:
          insecureSkipTLSVerify: "false"
  # ...
```

where:

caCert

Specifies the Base64-encoded CA certificate string.

insecureSkipTLSVerify

Specifies the **insecureSkipTLSVerify** configuration. The configuration can be set to either **"true"** or **"false"**. If set to **"true"**, SSL/TLS security is disabled. If set to **"false"**, SSL/TLS security is enabled.

5.7.1.4.3. Using CA certificates with the velero command aliased for Velero deployment

You might want to use the Velero CLI without installing it locally on your system by creating an alias for it.

Prerequisites

- You must be logged in to the OpenShift Container Platform cluster as a user with the **cluster-admin** role.
- You must have the OpenShift CLI (**oc**) installed.

Procedure

1. To use an aliased Velero command, run the following command:

```
$ alias velero='oc -n openshift-adp exec deployment/velero -c velero -it -- ./velero'
```

2. Check that the alias is working by running the following command:

```
$ velero version
```

```
Client:
Version: v1.12.1-OADP
Git commit: -
Server:
Version: v1.12.1-OADP
```

3. To use a CA certificate with this command, you can add a certificate to the Velero deployment by running the following commands:

```
$ CA_CERT=$(oc -n openshift-adp get dataprotectionapplications.oadp.openshift.io <dpa-name> -o jsonpath='{.spec.backupLocations[0].velero.objectStorage.caCert}')
```

```
$ [[ -n $CA_CERT ]] && echo "$CA_CERT" | base64 -d | oc exec -n openshift-adp -i
deploy/velero -c velero -- bash -c "cat > /tmp/your-cacert.txt" || echo "DPA BSL has no
caCert"
```

```
$ velero describe backup <backup_name> --details --cacert /tmp/<your_cacert>.txt
```

4. To fetch the backup logs, run the following command:

```
$ velero backup logs <backup_name> --cacert /tmp/<your_cacert.txt>
```

You can use these logs to view failures and warnings for the resources that you cannot back up.

5. If the Velero pod restarts, the **/tmp/your-cacert.txt** file disappears, and you must re-create the **/tmp/your-cacert.txt** file by re-running the commands from the previous step.

6. You can check if the **/tmp/your-cacert.txt** file still exists, in the file location where you stored it, by running the following command:

```
$ oc exec -n openshift-adp -i deploy/velero -c velero -- bash -c "ls /tmp/your-cacert.txt"
/tmp/your-cacert.txt
```

In a future release of OpenShift API for Data Protection (OADP), we plan to mount the certificate to the Velero pod so that this step is not required.

5.7.1.4.4. Configuring node agents and node labels

The Data Protection Application (DPA) uses the **nodeSelector** field to select which nodes can run the node agent. The **nodeSelector** field is the recommended form of node selection constraint.

Procedure

1. Run the node agent on any node that you choose by adding a custom label:

```
$ oc label node/<node_name> node-role.kubernetes.io/nodeAgent=""
```



NOTE

Any label specified must match the labels on each node.

2. Use the same custom label in the **DPA.spec.configuration.nodeAgent.podConfig.nodeSelector** field, which you used for labeling nodes:

```
configuration:
  nodeAgent:
    enable: true
  podConfig:
    nodeSelector:
      node-role.kubernetes.io/nodeAgent: ""
```

The following example is an anti-pattern of **nodeSelector** and does not work unless both labels, **node-role.kubernetes.io/infra: ""** and **node-role.kubernetes.io/worker: ""**, are on the node:

```
configuration:
  nodeAgent:
    enable: true
  podConfig:
    nodeSelector:
      node-role.kubernetes.io/infra: ""
      node-role.kubernetes.io/worker: ""
```

5.7.1.5. Configuring the backup storage location with a MD5 checksum algorithm

You can configure the Backup Storage Location (BSL) in the Data Protection Application (DPA) to use a MD5 checksum algorithm for both Amazon Simple Storage Service (Amazon S3) and S3-compatible storage providers. The checksum algorithm calculates the checksum for uploading and downloading

objects to Amazon S3. You can use one of the following options to set the **checksumAlgorithm** field in the **spec.backupLocations.velero.config.checksumAlgorithm** section of the DPA.

- **CRC32**
- **CRC32C**
- **SHA1**
- **SHA256**

You can also set the **checksumAlgorithm** field to an empty value to skip the MD5 checksum check. If you do not set a value for the **checksumAlgorithm** field, then the default value is set to **CRC32**.

Prerequisites

- You have installed the OADP Operator.
- You have configured Amazon S3, or S3-compatible object storage as a backup location.

Procedure

- Configure the BSL in the DPA as shown in the following example:

Example Data Protection Application

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: test-dpa
  namespace: openshift-adp
spec:
  backupLocations:
  - name: default
    velero:
      config:
        checksumAlgorithm: ""
        insecureSkipTLSVerify: "true"
        profile: "default"
        region: <bucket_region>
        s3ForcePathStyle: "true"
        s3Url: <bucket_url>
      credential:
        key: cloud
        name: cloud-credentials
      default: true
      objectStorage:
        bucket: <bucket_name>
        prefix: velero
        provider: aws
      configuration:
        velero:
          defaultPlugins:
```


- openshift
- aws
- csi

where:

checksumAlgorithm

Specifies the **checksumAlgorithm**. In this example, the **checksumAlgorithm** field is set to an empty value. You can select an option from the following list: **CRC32**, **CRC32C**, **SHA1**, **SHA256**.



IMPORTANT

If you are using Noobaa as the object storage provider, and you do not set the **spec.backupLocations.velero.config.checksumAlgorithm** field in the DPA, an empty value of **checksumAlgorithm** is added to the BSL configuration.

The empty value is only added for BSLs that are created using the DPA. This value is not added if you create the BSL by using any other method.

5.7.1.6. Configuring the DPA with client burst and QPS settings

The burst setting determines how many requests can be sent to the **velero** server before the limit is applied. After the burst limit is reached, the queries per second (QPS) setting determines how many additional requests can be sent per second.

You can set the burst and QPS values of the **velero** server by configuring the Data Protection Application (DPA) with the burst and QPS values. You can use the **dpa.configuration.velero.client-burst** and **dpa.configuration.velero.client-qps** fields of the DPA to set the burst and QPS values.

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **client-burst** and the **client-qps** fields in the DPA as shown in the following example:

Example Data Protection Application

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: test-dpa
  namespace: openshift-adp
spec:
  backupLocations:
    - name: default
  velero:
    config:
      insecureSkipTLSVerify: "true"
      profile: "default"
      region: <bucket_region>
```

```

    s3ForcePathStyle: "true"
    s3Url: <bucket_url>
  credential:
    key: cloud
    name: cloud-credentials
  default: true
  objectStorage:
    bucket: <bucket_name>
    prefix: velero
    provider: aws
  configuration:
    nodeAgent:
      enable: true
      uploaderType: restic
    velero:
      client-burst: 500
      client-qps: 300
      defaultPlugins:
        - openshift
        - aws
        - kubevirt

```

where:

client-burst

Specifies the **client-burst** value. In this example, the **client-burst** field is set to 500.

client-qps

Specifies the **client-qps** value. In this example, the **client-qps** field is set to 300.

5.7.1.7. Configuring node agent load affinity

You can schedule the node agent pods on specific nodes by using the **spec.podConfig.nodeSelector** object of the **DataProtectionApplication** (DPA) custom resource (CR).

See the following example in which you can schedule the node agent pods on nodes with the label **label.io/role: cpu-1** and **other-label.io/other-role: cpu-2**.

```

...
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
    podConfig:
      nodeSelector:
        label.io/role: cpu-1
        other-label.io/other-role: cpu-2
...

```

You can add more restrictions on the node agent pods scheduling by using the **nodeagent.loadAffinity** object in the DPA spec.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the DPA spec **nodegent.loadAffinity** object as shown in the following example. In the example, you ensure that the node agent pods are scheduled only on nodes with the label **label.io/role: cpu-1** and the label **label.io/hostname** matching with either **node1** or **node2**.

```
...
spec:
  configuration:
    nodeAgent:
      enable: true
      loadAffinity:
        - nodeSelector:
            matchLabels:
              label.io/role: cpu-1
            matchExpressions:
              - key: label.io/hostname
                operator: In
                values:
                  - node1
                  - node2
        ...
```

where:

loadAffinity

Specifies the **loadAffinity** object by adding the **matchLabels** and **matchExpressions** objects.

matchExpressions

Specifies the **matchExpressions** object to add restrictions on the node agent pods scheduling.

5.7.1.8. Node agent load affinity guidelines

Use the following guidelines to configure the node agent **loadAffinity** object in the **DataProtectionApplication** (DPA) custom resource (CR).

- Use the **spec.nodeagent.podConfig.nodeSelector** object for simple node matching.
- Use the **loadAffinity.nodeSelector** object without the **podConfig.nodeSelector** object for more complex scenarios.
- You can use both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects, but the **loadAffinity** object must be equal or more restrictive as compared to the **podConfig** object. In this scenario, the **podConfig.nodeSelector** labels must be a subset of the labels used in the **loadAffinity.nodeSelector** object.

- You cannot use the **matchExpressions** and **matchLabels** fields if you have configured both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects in the DPA.
- See the following example to configure both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects in the DPA.

```
...
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
      loadAffinity:
        - nodeSelector:
            matchLabels:
              label.io/location: 'US'
              label.io/gpu: 'no'
    podConfig:
      nodeSelector:
        label.io/gpu: 'no'
```

5.7.1.9. Configuring node agent load concurrency

You can control the maximum number of node agent operations that can run simultaneously on each node within your cluster.

You can configure it using one of the following fields of the Data Protection Application (DPA):

- **globalConfig**: Defines a default concurrency limit for the node agent across all nodes.
- **perNodeConfig**: Specifies different concurrency limits for specific nodes based on **nodeSelector** labels. This provides flexibility for environments where certain nodes might have different resource capacities or roles.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.

Procedure

1. If you want to use load concurrency for specific nodes, add labels to those nodes:

```
$ oc label node/<node_name> label.io/instance-type='large'
```

2. Configure the load concurrency fields for your DPA instance:

```
configuration:
  nodeAgent:
    enable: true
    uploaderType: kopia
    loadConcurrency:
      globalConfig: 1
      perNodeConfig:
        - nodeSelector:
```

```

matchLabels:
  label.io/instance-type: large
number: 3

```

where:

globalConfig

Specifies the global concurrent number. The default value is 1, which means there is no concurrency and only one load is allowed. The **globalConfig** value does not have a limit.

label.io/instance-type

Specifies the label for per-node concurrency.

number

Specifies the per-node concurrent number. You can specify many per-node concurrent numbers, for example, based on the instance type and size. The range of per-node concurrent number is the same as the global concurrent number. If the configuration file contains a per-node concurrent number and a global concurrent number, the per-node concurrent number takes precedence.

5.7.1.10. Configuring the node agent as a non-root and non-privileged user

To enhance the node agent security, you can configure the OADP Operator node agent daemonset to run as a non-root and non-privileged user by using the **spec.configuration.velero.disableFsBackup** setting in the **DataProtectionApplication** (DPA) custom resource (CR).

By setting the **spec.configuration.velero.disableFsBackup** setting to **true**, the node agent security context sets the root file system to read-only and sets the **privileged** flag to **false**.



NOTE

Setting **spec.configuration.velero.disableFsBackup** to **true** enhances the node agent security by removing the need for privileged containers and enforcing a read-only root file system.

However, it also disables File System Backup (FSB) with Kopia. If your workloads rely on FSB for backing up volumes that do not support native snapshots, then you should evaluate whether the **disableFsBackup** configuration fits your use case.

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **disableFsBackup** field in the DPA as shown in the following example:

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: ts-dpa
  namespace: openshift-adp
spec:
  backupLocations:
    - velero:

```

```
credential:
  key: cloud
  name: cloud-credentials
default: true
objectStorage:
  bucket: <bucket_name>
  prefix: velero
provider: gcp
configuration:
  nodeAgent:
    enable: true
    uploaderType: kopia
velero:
  defaultPlugins:
    - csi
    - gcp
    - openshift
  disableFsBackup: true
```

where:

nodeAgent

Specifies to enable the node agent in the DPA.

disableFsBackup

Specifies to set the **disableFsBackup** field to **true**.

Verification

1. Verify that the node agent security context is set to run as non-root and the root file system is **readOnly** by running the following command:

```
$ oc get daemonset node-agent -o yaml
```

The example output is as following:

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  ...
  name: node-agent
  namespace: openshift-adp
  ...
spec:
  ...
  template:
    metadata:
      ...
    spec:
      containers:
        ...
        securityContext:
          allowPrivilegeEscalation: false
          capabilities:
            drop:
```

```

- ALL
privileged: false
readOnlyRootFilesystem: true
...
nodeSelector:
  kubernetes.io/os: linux
os:
  name: linux
restartPolicy: Always
schedulerName: default-scheduler
securityContext:
  runAsNonRoot: true
  seccompProfile:
    type: RuntimeDefault
serviceAccount: velero
serviceAccountName: velero
....

```

where:

allowPrivilegeEscalation

Specifies that the **allowPrivilegeEscalation** field is false.

privileged

Specifies that the **privileged** field is false.

readOnlyRootFilesystem

Specifies that the root file system is read-only.

runAsNonRoot

Specifies that the node agent is run as a non-root user.

5.7.1.11. Configuring repository maintenance

OADP repository maintenance is a background job, you can configure it independently of the node agent pods. This means that you can schedule the repository maintenance pod on a node where the node agent is or is not running.

You can use the repository maintenance job affinity configurations in the **DataProtectionApplication** (DPA) custom resource (CR) only if you use Kopia as the backup repository.

You have the option to configure the load affinity at the global level affecting all repositories. Or you can configure the load affinity for each repository. You can also use a combination of global and per-repository configuration.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the **loadAffinity** object in the DPA spec by using either one or both of the following methods:
 - Global configuration: Configure load affinity for all repositories as shown in the following example:

```
...
spec:
  configuration:
    repositoryMaintenance:
      global:
        podResources:
          cpuRequest: "100m"
          cpuLimit: "200m"
          memoryRequest: "100Mi"
          memoryLimit: "200Mi"
        loadAffinity:
          - nodeSelector:
              matchLabels:
                label.io/gpu: 'no'
            matchExpressions:
              - key: label.io/location
                operator: In
                values:
                  - US
                  - EU
```

where:

repositoryMaintenance

Specifies the **repositoryMaintenance** object as shown in the example.

global

Specifies the **global** object to configure load affinity for all repositories.

- Per-repository configuration: Configure load affinity per repository as shown in the following example:

```
...
spec:
  configuration:
    repositoryMaintenance:
      myrepositoryname:
        loadAffinity:
          - nodeSelector:
              matchLabels:
                label.io/cpu: 'yes'
```

where:

myrepositoryname

Specifies the **repositoryMaintenance** object for each repository.

5.7.1.12. Configuring Velero load affinity

With each OADP deployment, there is one Velero pod and its main purpose is to schedule Velero workloads. To schedule the Velero pod, you can use the **velero.podConfig.nodeSelector** and the **velero.loadAffinity** objects in the **DataProtectionApplication** (DPA) custom resource (CR) spec.

Use the **podConfig.nodeSelector** object to assign the Velero pod to specific nodes. You can also configure the **velero.loadAffinity** object for pod-level affinity and anti-affinity.

The OpenShift scheduler applies the rules and performs the scheduling of the Velero pod deployment.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the **velero.podConfig.nodeSelector** and the **velero.loadAffinity** objects in the DPA spec as shown in the following examples:

- **velero.podConfig.nodeSelector** object configuration:

```
...
spec:
  configuration:
    velero:
      podConfig:
        nodeSelector:
          some-label.io/custom-node-role: backup-core
```

- **velero.loadAffinity** object configuration:

```
...
spec:
  configuration:
    velero:
      loadAffinity:
        - nodeSelector:
            matchLabels:
              label.io/gpu: 'no'
            matchExpressions:
              - key: label.io/location
                operator: In
            values:
              - US
              - EU
```

5.7.1.13. Overriding the imagePullPolicy setting in the DPA

In OADP 1.4.0 or earlier, the Operator sets the **imagePullPolicy** field of the Velero and node agent pods to **Always** for all images.

In OADP 1.4.1 or later, the Operator first checks if each image has the **sha256** or **sha512** digest and sets the **imagePullPolicy** field accordingly:

- If the image has the digest, the Operator sets **imagePullPolicy** to **IfNotPresent**.
- If the image does not have the digest, the Operator sets **imagePullPolicy** to **Always**.

You can also override the **imagePullPolicy** field by using the **spec.imagePullPolicy** field in the Data Protection Application (DPA).

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **spec.imagePullPolicy** field in the DPA as shown in the following example:

Example Data Protection Application

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: test-dpa
  namespace: openshift-adp
spec:
  backupLocations:
    - name: default
      velero:
        config:
          insecureSkipTLSVerify: "true"
          profile: "default"
          region: <bucket_region>
          s3ForcePathStyle: "true"
          s3Url: <bucket_url>
        credential:
          key: cloud
          name: cloud-credentials
        default: true
        objectStorage:
          bucket: <bucket_name>
          prefix: velero
          provider: aws
      configuration:
        nodeAgent:
          enable: true
          uploaderType: kopia
        velero:
          defaultPlugins:
            - openshift
            - aws
            - kubevirt
            - csi
      imagePullPolicy: Never
```

where:

imagePullPolicy

Specifies the value for **imagePullPolicy**. In this example, the **imagePullPolicy** field is set to **Never**.

5.7.1.14. Enabling CSI in the DataProtectionApplication CR

You enable the Container Storage Interface (CSI) in the **DataProtectionApplication** custom resource (CR) in order to back up persistent volumes with CSI snapshots.

Prerequisites

- The cloud provider must support CSI snapshots.

Procedure

- Edit the **DataProtectionApplication** CR, as in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - csi
```

where:

csi

Specifies the **csi** default plugin.

5.7.1.15. Disabling the node agent in DataProtectionApplication

If you are not using **Restic**, **Kopia**, or **DataMover** for your backups, you can disable the **nodeAgent** field in the **DataProtectionApplication** custom resource (CR). Before you disable **nodeAgent**, ensure the OADP Operator is idle and not running any backups.

Procedure

1. To disable the **nodeAgent**, set the **enable** flag to **false**. See the following example:

Example DataProtectionApplication CR

```
# ...
configuration:
  nodeAgent:
    enable: false
  uploaderType: kopia
# ...
```

where:

enable

Enables the node agent.

2. To enable the **nodeAgent**, set the **enable** flag to **true**. See the following example:

Example DataProtectionApplication CR

```
# ...
configuration:
  nodeAgent:
    enable: true
    uploaderType: kopia
# ...
```

where:

enable

Enables the node agent.

You can set up a job to enable and disable the **nodeAgent** field in the **DataProtectionApplication** CR. For more information, see "Running tasks in pods using jobs".

Additional resources

- [Installing the Data Protection Application with the **kubevirt** and **openshift** plugins](#)
- [Running tasks in pods using jobs](#)

5.8. CONFIGURING OADP WITH IBM CLOUD

5.8.1. Configuring the OpenShift API for Data Protection with IBM Cloud

You install the OpenShift API for Data Protection (OADP) Operator on an IBM Cloud cluster to back up and restore applications on the cluster. You configure IBM Cloud Object Storage (COS) to store the backups.

5.8.1.1. Configuring the COS instance

You create an IBM Cloud Object Storage (COS) instance to store the OADP backup data. After you create the COS instance, configure the **HMAC** service credentials.

Prerequisites

- You have an IBM Cloud Platform account.
- You installed the [IBM Cloud CLI](#).
- You are logged in to IBM Cloud.

Procedure

1. Install the IBM Cloud Object Storage (COS) plugin by running the following command:

```
$ ibmcloud plugin install cos -f
```

2. Set a bucket name by running the following command:

```
$ BUCKET=<bucket_name>
```

3. Set a bucket region by running the following command:

```
$ REGION=<bucket_region> ❶
```

❶ ❶ Specify the bucket region, for example, **eu-gb**.

4. Create a resource group by running the following command:

```
$ ibmcloud resource group-create <resource_group_name>
```

5. Set the target resource group by running the following command:

```
$ ibmcloud target -g <resource_group_name>
```

6. Verify that the target resource group is correctly set by running the following command:

```
$ ibmcloud target
```

Example output

```
API endpoint:  https://cloud.ibm.com
Region:
User:          test-user
Account:       Test Account (fb6.....e95) <-> 2...122
Resource group: Default
```

In the example output, the resource group is set to **Default**.

7. Set a resource group name by running the following command:

```
$ RESOURCE_GROUP=<resource_group> ❶
```

❶ Specify the resource group name, for example, **"default"**.

8. Create an IBM Cloud **service-instance** resource by running the following command:

```
$ ibmcloud resource service-instance-create \
  <service_instance_name> \ ❶
  <service_name> \ ❷
```

```
<service_plan> \ 3
<region_name> 4
```

- 1** Specify a name for the **service-instance** resource.
- 2** Specify the service name. Alternatively, you can specify a service ID.
- 3** Specify the service plan for your IBM Cloud account.
- 4** Specify the region name.

Example command

```
$ ibmcloud resource service-instance-create test-service-instance cloud-object-storage \ 1
standard \
global \
-d premium-global-deployment 2
```

- 1** The service name is **cloud-object-storage**.
- 2** The **-d** flag specifies the deployment name.

9. Extract the service instance ID by running the following command:

```
$ SERVICE_INSTANCE_ID=$(ibmcloud resource service-instance test-service-instance --
output json | jq -r '[0].id')
```

10. Create a COS bucket by running the following command:

```
$ ibmcloud cos bucket-create V/
--bucket $BUCKET V/
--ibm-service-instance-id $SERVICE_INSTANCE_ID V/
--region $REGION
```

Variables such as **\$BUCKET**, **\$SERVICE_INSTANCE_ID**, and **\$REGION** are replaced by the values you set previously.

11. Create **HMAC** credentials by running the following command.

```
$ ibmcloud resource service-key-create test-key Writer --instance-name test-service-instance
--parameters {"HMAC":true}
```

12. Extract the access key ID and the secret access key from the **HMAC** credentials and save them in the **credentials-velero** file. You can use the **credentials-velero** file to create a **secret** for the backup storage location. Run the following command:

```
$ cat > credentials-velero << __EOF__
[default]
aws_access_key_id=$(ibmcloud resource service-key test-key -o json | jq -r '[
0].credentials.cos_hmac_keys.access_key_id')
```

```
aws_secret_access_key=$(ibmcloud resource service-key test-key -o json | jq -r '[0].credentials.cos_hmac_keys.secret_access_key')
__EOF__
```

5.8.1.2. Creating a default Secret

You create a default **Secret** if your backup and snapshot locations use the same credentials or if you do not require a snapshot location.



NOTE

The **DataProtectionApplication** custom resource (CR) requires a default **Secret**. Otherwise, the installation will fail. If the name of the backup location **Secret** is not specified, the default name is used.

If you do not want to use the backup location credentials during the installation, you can create a **Secret** with the default name by using an empty **credentials-velero** file.

Prerequisites

- Your object storage and cloud storage, if any, must use the same credentials.
- You must configure object storage for Velero.

Procedure

1. Create a **credentials-velero** file for the backup storage location in the appropriate format for your cloud provider.
2. Create a **Secret** custom resource (CR) with the default name:

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file cloud=credentials-velero
```

The **Secret** is referenced in the **spec.backupLocations.credential** block of the **DataProtectionApplication** CR when you install the Data Protection Application.

5.8.1.3. Creating secrets for different credentials

If your backup and snapshot locations use different credentials, you must create two **Secret** objects:

- Backup location **Secret** with a custom name. The custom name is specified in the **spec.backupLocations** block of the **DataProtectionApplication** custom resource (CR).
- Snapshot location **Secret** with the default name, **cloud-credentials**. This **Secret** is not specified in the **DataProtectionApplication** CR.

Procedure

1. Create a **credentials-velero** file for the snapshot location in the appropriate format for your cloud provider.
2. Create a **Secret** for the snapshot location with the default name:

—

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file cloud=credentials-velero
```

3. Create a **credentials-velero** file for the backup location in the appropriate format for your object storage.
4. Create a **Secret** for the backup location with a custom name:

```
$ oc create secret generic <custom_secret> -n openshift-adp --from-file cloud=credentials-velero
```

5. Add the **Secret** with the custom name to the **DataProtectionApplication** CR, as in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  ...
  backupLocations:
    - velero:
        provider: <provider>
        default: true
        credential:
          key: cloud
          name: <custom_secret> 1
        objectStorage:
          bucket: <bucket_name>
          prefix: <prefix>
```

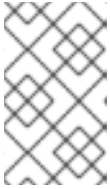
- 1 Backup location **Secret** with custom name.

5.8.1.4. Installing the Data Protection Application

You install the Data Protection Application (DPA) by creating an instance of the **DataProtectionApplication** API.

Prerequisites

- You must install the OADP Operator.
- You must configure object storage as a backup location.
- If you use snapshots to back up PVs, your cloud provider must support either a native snapshot API or Container Storage Interface (CSI) snapshots.
- If the backup and snapshot locations use the same credentials, you must create a **Secret** with the default name, **cloud-credentials**.

**NOTE**

If you do not want to specify backup or snapshot locations during the installation, you can create a default **Secret** with an empty **credentials-velero** file. If there is no default **Secret**, the installation will fail.

Procedure

1. Click **Ecosystem → Installed Operators** and select the OADP Operator.
2. Under **Provided APIs**, click **Create instance** in the **DataProtectionApplication** box.
3. Click **YAML View** and update the parameters of the **DataProtectionApplication** manifest:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  namespace: openshift-adp
  name: <dpa_name>
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - aws
        - csi
  backupLocations:
    - velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket_name>
          prefix: velero
        config:
          insecureSkipTLSVerify: 'true'
          profile: default
          region: <region_name>
          s3ForcePathStyle: 'true'
          s3Url: <s3_url>
        credential:
          key: cloud
          name: cloud-credentials
```

where:

provider

Specifies that the provider is **aws** when you use IBM Cloud as a backup storage location.

bucket

Specifies the IBM Cloud Object Storage (COS) bucket name.

region

Specifies the COS region name, for example, **eu-gb**.

s3Url

Specifies the S3 URL of the COS bucket. For example, <http://s3.eu-gb.cloud-object-storage.appdomain.cloud>. Here, **eu-gb** is the region name. Replace the region name according to your bucket region.

name

Specifies the name of the secret you created by using the access key and the secret access key from the **HMAC** credentials.

4. Click **Create**.

Verification

1. Verify the installation by viewing the OpenShift API for Data Protection (OADP) resources by running the following command:

```
$ oc get all -n openshift-adp
```

NAME	READY	STATUS	RESTARTS	AGE
pod/oadp-operator-controller-manager-67d9494d47-6l8z8	2/2	Running	0	2m8s
pod/node-agent-9cq4q	1/1	Running	0	94s
pod/node-agent-m4lts	1/1	Running	0	94s
pod/node-agent-pv4kr	1/1	Running	0	95s
pod/velero-588db7f655-n842v	1/1	Running	0	95s

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP
service/oadp-operator-controller-manager-metrics-service	ClusterIP	172.30.70.140	
<none>	8443/TCP	2m8s	
service/openshift-adp-velero-metrics-svc	ClusterIP	172.30.10.0	<none>
8085/TCP	8h		

NAME	DESIRED	CURRENT	READY	UP-TO-DATE	AVAILABLE	AGE
daemonset.apps/node-agent	3	3	3	3	<none>	96s

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
deployment.apps/oadp-operator-controller-manager	1/1	1	1	2m9s
deployment.apps/velero	1/1	1	1	96s

NAME	DESIRED	CURRENT	READY	AGE
replicaset.apps/oadp-operator-controller-manager-67d9494d47	1	1	1	2m9s
replicaset.apps/velero-588db7f655	1	1	1	96s

2. Verify that the **DataProtectionApplication** (DPA) is reconciled by running the following command:

```
$ oc get dpa dpa-sample -n openshift-adp -o jsonpath='{.status}'
```

```
{
  "conditions": [
    {
      "lastTransitionTime": "2023-10-27T01:23:57Z",
      "message": "Reconcile complete",
      "reason": "Complete",
      "status": "True",
      "type": "Reconciled"
    }
  ]
}
```

3. Verify the **type** is set to **Reconciled**.

4. Verify the backup storage location and confirm that the **PHASE** is **Available** by running the following command:

```
$ oc get backupstoragelocations.velero.io -n openshift-adp
```

NAME	PHASE	LAST VALIDATED	AGE	DEFAULT
dpa-sample-1	Available	1s	3d16h	true

5.8.1.5. Setting Velero CPU and memory resource allocations

You set the CPU and memory resource allocations for the **Velero** pod by editing the **DataProtectionApplication** custom resource (CR) manifest.

Prerequisites

- You must have the OpenShift API for Data Protection (OADP) Operator installed.

Procedure

- Edit the values in the **spec.configuration.velero.podConfig.ResourceAllocations** block of the **DataProtectionApplication** CR manifest, as in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  # ...
  configuration:
    velero:
      podConfig:
        nodeSelector: <node_selector>
        resourceAllocations:
          limits:
            cpu: "1"
            memory: 1024Mi
          requests:
            cpu: 200m
            memory: 256Mi
```

where:

nodeSelector

Specifies the node selector to be supplied to Velero podSpec.

resourceAllocations

Specifies the resource allocations listed for average usage.

**NOTE**

Kopia is an option in OADP 1.3 and later releases. You can use Kopia for file system backups, and Kopia is your only option for Data Mover cases with the built-in Data Mover.

Kopia is more resource intensive than Restic, and you might need to adjust the CPU and memory requirements accordingly.

5.8.1.6. Configuring node agents and node labels

The Data Protection Application (DPA) uses the **nodeSelector** field to select which nodes can run the node agent. The **nodeSelector** field is the recommended form of node selection constraint.

Procedure

1. Run the node agent on any node that you choose by adding a custom label:

```
$ oc label node/<node_name> node-role.kubernetes.io/nodeAgent=""
```

**NOTE**

Any label specified must match the labels on each node.

2. Use the same custom label in the **DPA.spec.configuration.nodeAgent.podConfig.nodeSelector** field, which you used for labeling nodes:

```
configuration:
  nodeAgent:
    enable: true
  podConfig:
    nodeSelector:
      node-role.kubernetes.io/nodeAgent: ""
```

The following example is an anti-pattern of **nodeSelector** and does not work unless both labels, **node-role.kubernetes.io/infra: ""** and **node-role.kubernetes.io/worker: ""**, are on the node:

```
configuration:
  nodeAgent:
    enable: true
  podConfig:
    nodeSelector:
      node-role.kubernetes.io/infra: ""
      node-role.kubernetes.io/worker: ""
```

5.8.1.7. Configuring the DPA with client burst and QPS settings

The burst setting determines how many requests can be sent to the **velero** server before the limit is applied. After the burst limit is reached, the queries per second (QPS) setting determines how many additional requests can be sent per second.

You can set the burst and QPS values of the **velero** server by configuring the Data Protection Application (DPA) with the burst and QPS values. You can use the **dpa.configuration.velero.client-burst** and **dpa.configuration.velero.client-qps** fields of the DPA to set the burst and QPS values.

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **client-burst** and the **client-qps** fields in the DPA as shown in the following example:

Example Data Protection Application

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: test-dpa
  namespace: openshift-adp
spec:
  backupLocations:
  - name: default
    velero:
      config:
        insecureSkipTLSVerify: "true"
        profile: "default"
        region: <bucket_region>
        s3ForcePathStyle: "true"
        s3Url: <bucket_url>
      credential:
        key: cloud
        name: cloud-credentials
      default: true
      objectStorage:
        bucket: <bucket_name>
        prefix: velero
      provider: aws
    configuration:
      nodeAgent:
        enable: true
        uploaderType: restic
      velero:
        client-burst: 500
        client-qps: 300
      defaultPlugins:
        - openshift
        - aws
        - kubevirt
```

where:

client-burst

Specifies the **client-burst** value. In this example, the **client-burst** field is set to 500.

client-qps

Specifies the **client-qps** value. In this example, the **client-qps** field is set to 300.

5.8.1.8. Configuring node agent load affinity

You can schedule the node agent pods on specific nodes by using the **spec.podConfig.nodeSelector** object of the **DataProtectionApplication** (DPA) custom resource (CR).

See the following example in which you can schedule the node agent pods on nodes with the label **label.io/role: cpu-1** and **other-label.io/other-role: cpu-2**.

```
...
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
      podConfig:
        nodeSelector:
          label.io/role: cpu-1
          other-label.io/other-role: cpu-2
    ...
```

You can add more restrictions on the node agent pods scheduling by using the **nodeagent.loadAffinity** object in the DPA spec.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the DPA spec **nodeagent.loadAffinity** object as shown in the following example. In the example, you ensure that the node agent pods are scheduled only on nodes with the label **label.io/role: cpu-1** and the label **label.io/hostname** matching with either **node1** or **node2**.

```
...
spec:
  configuration:
    nodeAgent:
      enable: true
      loadAffinity:
        - nodeSelector:
            matchLabels:
              label.io/role: cpu-1
            matchExpressions:
              - key: label.io/hostname
                operator: In
                values:
```

```

- node1
- node2
...

```

where:

loadAffinity

Specifies the **loadAffinity** object by adding the **matchLabels** and **matchExpressions** objects.

matchExpressions

Specifies the **matchExpressions** object to add restrictions on the node agent pods scheduling.

5.8.1.9. Node agent load affinity guidelines

Use the following guidelines to configure the node agent **loadAffinity** object in the **DataProtectionApplication** (DPA) custom resource (CR).

- Use the **spec.nodeagent.podConfig.nodeSelector** object for simple node matching.
- Use the **loadAffinity.nodeSelector** object without the **podConfig.nodeSelector** object for more complex scenarios.
- You can use both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects, but the **loadAffinity** object must be equal or more restrictive as compared to the **podConfig** object. In this scenario, the **podConfig.nodeSelector** labels must be a subset of the labels used in the **loadAffinity.nodeSelector** object.
- You cannot use the **matchExpressions** and **matchLabels** fields if you have configured both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects in the DPA.
- See the following example to configure both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects in the DPA.

```

...
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
      loadAffinity:
        - nodeSelector:
            matchLabels:
              label.io/location: 'US'
              label.io/gpu: 'no'
      podConfig:
        nodeSelector:
          label.io/gpu: 'no'

```

5.8.1.10. Configuring node agent load concurrency

You can control the maximum number of node agent operations that can run simultaneously on each node within your cluster.

You can configure it using one of the following fields of the Data Protection Application (DPA):

- **globalConfig**: Defines a default concurrency limit for the node agent across all nodes.
- **perNodeConfig**: Specifies different concurrency limits for specific nodes based on **nodeSelector** labels. This provides flexibility for environments where certain nodes might have different resource capacities or roles.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.

Procedure

1. If you want to use load concurrency for specific nodes, add labels to those nodes:

```
$ oc label node/<node_name> label.io/instance-type='large'
```

2. Configure the load concurrency fields for your DPA instance:

```
configuration:
  nodeAgent:
    enable: true
    uploaderType: kopia
    loadConcurrency:
      globalConfig: 1
      perNodeConfig:
        - nodeSelector:
            matchLabels:
              label.io/instance-type: large
          number: 3
```

where:

globalConfig

Specifies the global concurrent number. The default value is 1, which means there is no concurrency and only one load is allowed. The **globalConfig** value does not have a limit.

label.io/instance-type

Specifies the label for per-node concurrency.

number

Specifies the per-node concurrent number. You can specify many per-node concurrent numbers, for example, based on the instance type and size. The range of per-node concurrent number is the same as the global concurrent number. If the configuration file contains a per-node concurrent number and a global concurrent number, the per-node concurrent number takes precedence.

5.8.1.11. Configuring repository maintenance

OADP repository maintenance is a background job, you can configure it independently of the node agent pods. This means that you can schedule the repository maintenance pod on a node where the node agent is or is not running.

You can use the repository maintenance job affinity configurations in the **DataProtectionApplication** (DPA) custom resource (CR) only if you use Kopia as the backup repository.

You have the option to configure the load affinity at the global level affecting all repositories. Or you can configure the load affinity for each repository. You can also use a combination of global and per-repository configuration.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the **loadAffinity** object in the DPA spec by using either one or both of the following methods:
 - Global configuration: Configure load affinity for all repositories as shown in the following example:

```
...
spec:
  configuration:
    repositoryMaintenance:
      global:
        podResources:
          cpuRequest: "100m"
          cpuLimit: "200m"
          memoryRequest: "100Mi"
          memoryLimit: "200Mi"
        loadAffinity:
          - nodeSelector:
              matchLabels:
                label.io/gpu: 'no'
              matchExpressions:
                - key: label.io/location
                  operator: In
                  values:
                    - US
                    - EU
```

where:

repositoryMaintenance

Specifies the **repositoryMaintenance** object as shown in the example.

global

Specifies the **global** object to configure load affinity for all repositories.

- Per-repository configuration: Configure load affinity per repository as shown in the following example:

```
...
```

```
spec:
  configuration:
    repositoryMaintenance:
      myrepositoryname:
        loadAffinity:
          - nodeSelector:
              matchLabels:
                label.io/cpu: 'yes'
```

where:

myrepositoryname

Specifies the **repositoryMaintenance** object for each repository.

5.8.1.12. Configuring Velero load affinity

With each OADP deployment, there is one Velero pod and its main purpose is to schedule Velero workloads. To schedule the Velero pod, you can use the **velero.podConfig.nodeSelector** and the **velero.loadAffinity** objects in the **DataProtectionApplication** (DPA) custom resource (CR) spec.

Use the **podConfig.nodeSelector** object to assign the Velero pod to specific nodes. You can also configure the **velero.loadAffinity** object for pod-level affinity and anti-affinity.

The OpenShift scheduler applies the rules and performs the scheduling of the Velero pod deployment.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the **velero.podConfig.nodeSelector** and the **velero.loadAffinity** objects in the DPA spec as shown in the following examples:

- **velero.podConfig.nodeSelector** object configuration:

```
...
spec:
  configuration:
    velero:
      podConfig:
        nodeSelector:
          some-label.io/custom-node-role: backup-core
```

- **velero.loadAffinity** object configuration:

```
...
spec:
  configuration:
    velero:
```

```

loadAffinity:
  - nodeSelector:
      matchLabels:
        label.io/gpu: 'no'
      matchExpressions:
        - key: label.io/location
          operator: In
          values:
            - US
            - EU

```

5.8.1.13. Overriding the imagePullPolicy setting in the DPA

In OADP 1.4.0 or earlier, the Operator sets the **imagePullPolicy** field of the Velero and node agent pods to **Always** for all images.

In OADP 1.4.1 or later, the Operator first checks if each image has the **sha256** or **sha512** digest and sets the **imagePullPolicy** field accordingly:

- If the image has the digest, the Operator sets **imagePullPolicy** to **IfNotPresent**.
- If the image does not have the digest, the Operator sets **imagePullPolicy** to **Always**.

You can also override the **imagePullPolicy** field by using the **spec.imagePullPolicy** field in the Data Protection Application (DPA).

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **spec.imagePullPolicy** field in the DPA as shown in the following example:

Example Data Protection Application

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: test-dpa
  namespace: openshift-adp
spec:
  backupLocations:
    - name: default
  velero:
    config:
      insecureSkipTLSVerify: "true"
      profile: "default"
      region: <bucket_region>
      s3ForcePathStyle: "true"
      s3Url: <bucket_url>
    credential:
      key: cloud
      name: cloud-credentials
    default: true

```

```

    objectStorage:
      bucket: <bucket_name>
      prefix: velero
      provider: aws
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
    velero:
      defaultPlugins:
        - openshift
        - aws
        - kubevirt
        - csi
  imagePullPolicy: Never

```

where:

imagePullPolicy

Specifies the value for **imagePullPolicy**. In this example, the **imagePullPolicy** field is set to **Never**.

5.8.14. Configuring the DPA with more than one BSL

Configure the **DataProtectionApplication** (DPA) custom resource (CR) with multiple **BackupStorageLocation** (BSL) resources to store backups across different locations using provider-specific credentials. This provides backup distribution and location-specific restore capabilities.

For example, you have configured the following two BSLs:

- Configured one BSL in the DPA and set it as the default BSL.
- Created another BSL independently by using the **BackupStorageLocation** CR.

As you have already set the BSL created through the DPA as the default, you cannot set the independently created BSL again as the default. This means, at any given time, you can set only one BSL as the default BSL.

Prerequisites

- You must install the OADP Operator.
- You must create the secrets by using the credentials provided by the cloud provider.

Procedure

1. Configure the **DataProtectionApplication** CR with more than one **BackupStorageLocation** CR. See the following example:

Example DPA

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
#...
backupLocations:

```

```

- name: aws
  velero:
    provider: aws
    default: true
    objectStorage:
      bucket: <bucket_name>
      prefix: <prefix>
    config:
      region: <region_name>
      profile: "default"
    credential:
      key: cloud
      name: cloud-credentials
- name: odf
  velero:
    provider: aws
    default: false
    objectStorage:
      bucket: <bucket_name>
      prefix: <prefix>
    config:
      profile: "default"
      region: <region_name>
      s3Url: <url>
      insecureSkipTLSVerify: "true"
      s3ForcePathStyle: "true"
    credential:
      key: cloud
      name: <custom_secret_name_odf>
#...

```

where:

name: aws

Specifies a name for the first BSL.

default: true

Indicates that this BSL is the default BSL. If a BSL is not set in the **Backup CR**, the default BSL is used. You can set only one BSL as the default.

<bucket_name>

Specifies the bucket name.

<prefix>

Specifies a prefix for Velero backups. For example, **velero**.

<region_name>

Specifies the AWS region for the bucket.

cloud-credentials

Specifies the name of the default **Secret** object that you created.

name: odf

Specifies a name for the second BSL.

<url>

Specifies the URL of the S3 endpoint.

<custom_secret_name_odf>

Specifies the correct name for the **Secret**. For example, **custom_secret_name_odf**. If you do not specify a **Secret** name, the default name is used.

2. Specify the BSL to be used in the backup CR. See the following example.

Example backup CR

```
apiVersion: velero.io/v1
kind: Backup
# ...
spec:
  includedNamespaces:
    - <namespace>
  storageLocation: <backup_storage_location>
  defaultVolumesToFsBackup: true
```

where:

<namespace>

Specifies the namespace to back up.

<backup_storage_location>

Specifies the storage location.

5.8.1.15. Disabling the node agent in DataProtectionApplication

If you are not using **Restic**, **Kopia**, or **DataMover** for your backups, you can disable the **nodeAgent** field in the **DataProtectionApplication** custom resource (CR). Before you disable **nodeAgent**, ensure the OADP Operator is idle and not running any backups.

Procedure

1. To disable the **nodeAgent**, set the **enable** flag to **false**. See the following example:

Example DataProtectionApplication CR

```
# ...
configuration:
  nodeAgent:
    enable: false
  uploaderType: kopia
# ...
```

where:

enable

Enables the node agent.

2. To enable the **nodeAgent**, set the **enable** flag to **true**. See the following example:

Example DataProtectionApplication CR

```
# ...
configuration:
  nodeAgent:
    enable: true
    uploaderType: kopio
# ...
```

where:

enable

Enables the node agent.

You can set up a job to enable and disable the **nodeAgent** field in the **DataProtectionApplication** CR. For more information, see "Running tasks in pods using jobs".

5.9. CONFIGURING OADP WITH AZURE

5.9.1. Configuring the OpenShift API for Data Protection with Microsoft Azure

You install the OpenShift API for Data Protection (OADP) with Microsoft Azure by installing the OADP Operator. The Operator installs [Velero 1.16](#).



NOTE

Starting from OADP 1.0.4, all OADP 1.0.z versions can only be used as a dependency of the Migration Toolkit for Containers Operator and are not available as a standalone Operator.

You configure Azure for Velero, create a default **Secret**, and then install the Data Protection Application. For more details, see [Installing the OADP Operator](#).

To install the OADP Operator in a restricted network environment, you must first disable the default software catalog sources and mirror the Operator catalog. See [Using Operator Lifecycle Manager in disconnected environments](#) for details.

5.9.1.1. Configuring Microsoft Azure

You configure Microsoft Azure for OpenShift API for Data Protection (OADP).

Prerequisites

- You must have the [Azure CLI](#) installed.

Tools that use Azure services should always have restricted permissions to make sure that Azure resources are safe. Therefore, instead of having applications sign in as a fully privileged user, Azure offers service principals. An Azure service principal is a name that can be used with applications, hosted services, or automated tools.

This identity is used for access to resources.

- Create a service principal

- Sign in using a service principal and password
- Sign in using a service principal and certificate
- Manage service principal roles
- Create an Azure resource using a service principal
- Reset service principal credentials

For more details, see [Create an Azure service principal with Azure CLI](#).

5.9.1.2. About backup and snapshot locations and their secrets

Review backup location, snapshot location, and secret configuration requirements for the **DataProtectionApplication** custom resource (CR). This helps you understand storage options and credential management for data protection operations.

5.9.1.2.1. Backup locations

You can specify one of the following AWS S3-compatible object storage solutions as a backup location:

- Multicloud Object Gateway (MCG)
- Red Hat Container Storage
- Ceph RADOS Gateway; also known as Ceph Object Gateway
- Red Hat OpenShift Data Foundation
- MinIO

Velero backs up OpenShift Container Platform resources, Kubernetes objects, and internal images as an archive file on object storage.

5.9.1.2.2. Snapshot locations

If you use your cloud provider's native snapshot API to back up persistent volumes, you must specify the cloud provider as the snapshot location.

If you use Container Storage Interface (CSI) snapshots, you do not need to specify a snapshot location because you will create a **VolumeSnapshotClass** CR to register the CSI driver.

If you use File System Backup (FSB), you do not need to specify a snapshot location because FSB backs up the file system on object storage.

5.9.1.2.3. Secrets

If the backup and snapshot locations use the same credentials or if you do not require a snapshot location, you create a default **Secret**.

If the backup and snapshot locations use different credentials, you create two secret objects:

- Custom **Secret** for the backup location, which you specify in the **DataProtectionApplication** CR.

- Default **Secret** for the snapshot location, which is not referenced in the **DataProtectionApplication** CR.



IMPORTANT

The Data Protection Application requires a default **Secret**. Otherwise, the installation will fail.

If you do not want to specify backup or snapshot locations during the installation, you can create a default **Secret** with an empty **credentials-velero** file.

5.9.1.3. About authenticating OADP with Azure

You can authenticate OADP with Azure by using the following methods:

- A Velero-specific service principal with secret-based authentication.
- A Velero-specific storage account access key with secret-based authentication.
- Azure Security Token Service.

5.9.1.4. Using a service principal or a storage account access key

You create a default **Secret** object and reference it in the backup storage location custom resource. The credentials file for the **Secret** object can contain information about the Azure service principal or a storage account access key.

The default name of the **Secret** is **cloud-credentials-azure**.



NOTE

The **DataProtectionApplication** custom resource (CR) requires a default **Secret**. Otherwise, the installation will fail. If the name of the backup location **Secret** is not specified, the default name is used.

If you do not want to use the backup location credentials during the installation, you can create a **Secret** with the default name by using an empty **credentials-velero** file.

Prerequisites

- You have access to the OpenShift cluster as a user with **cluster-admin** privileges.
- You have an Azure subscription with appropriate permissions.
- You have installed OADP.
- You have configured an object storage for storing the backups.

Procedure

1. Create a **credentials-velero** file for the backup storage location in the appropriate format for your cloud provider.
You can use one of the following two methods to authenticate OADP with Azure.

- Use the service principal with secret-based authentication. See the following example:

```
AZURE_SUBSCRIPTION_ID=<azure_subscription_id>
AZURE_TENANT_ID=<azure_tenant_id>
AZURE_CLIENT_ID=<azure_client_id>
AZURE_CLIENT_SECRET=<azure_client_secret>
AZURE_RESOURCE_GROUP=<azure_resource_group>
AZURE_CLOUD_NAME=<azure_cloud_name>
```

- Use a storage account access key. See the following example:

```
AZURE_STORAGE_ACCOUNT_ACCESS_KEY=
<azure_storage_account_access_key>
AZURE_SUBSCRIPTION_ID=<azure_subscription_id>
AZURE_RESOURCE_GROUP=<azure_resource_group>
AZURE_CLOUD_NAME=<azure_cloud_name>
```

2. Create a **Secret** custom resource (CR) with the default name:

```
$ oc create secret generic cloud-credentials-azure -n openshift-adp --from-file
cloud=credentials-velero
```

3. Reference the **Secret** in the **spec.backupLocations.velero.credential** block of the **DataProtectionApplication** CR when you install the Data Protection Application as shown in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  ...
  backupLocations:
    - velero:
        config:
          resourceGroup: <azure_resource_group>
          storageAccount: <azure_storage_account_id>
          subscriptionId: <azure_subscription_id>
        credential:
          key: cloud
          name: <custom_secret> 1
        provider: azure
        default: true
        objectStorage:
          bucket: <bucket_name>
          prefix: <prefix>
  snapshotLocations:
    - velero:
        config:
          resourceGroup: <azure_resource_group>
          subscriptionId: <azure_subscription_id>
          incremental: "true"
        provider: azure
```

- 1 Backup location **Secret** with custom name.

5.9.1.5. Using OADP with Azure Security Token Service authentication

You can use Microsoft Entra Workload ID to access Azure storage for OADP backup and restore operations. This approach uses the signed Kubernetes service account tokens of the OpenShift cluster. These token are automatically rotated every hour and exchanged with the Azure Active Directory (AD) access tokens, eliminating the need for long-term client secrets.

To use the Azure Security Token Service (STS) configuration, you need the **credentialsMode** field set to **Manual** during cluster installation. This approach uses the Cloud Credential Operator (**ccocli**) to set up the workload identity infrastructure, including the OpenID Connect (OIDC) provider, issuer configuration, and user-assigned managed identities.



NOTE

OADP with Azure STS configuration does not support **restic** File System Backups (FSB) and restores.

Prerequisites

- You have an OpenShift cluster installed on Microsoft Azure with Microsoft Entra Workload ID configured. For more details see, [Configuring an Azure cluster to use short-term credentials](#).
- You have the Azure CLI (**az**) installed and configured.
- You have access to the OpenShift cluster as a user with **cluster-admin** privileges.
- You have an Azure subscription with appropriate permissions.



NOTE

If your OpenShift cluster was not originally installed with Microsoft Entra Workload ID, you can enable short-term credentials after installation. This post-installation configuration is supported specifically for Azure clusters.

Procedure

1. If your cluster was installed with long-term credentials, you can switch to Microsoft Entra Workload ID authentication after installation. For more details, see [Enabling Microsoft Entra Workload ID on an existing cluster](#).



IMPORTANT

After enabling Microsoft Entra Workload ID on an existing Azure cluster, you must update all cluster components that use cloud credentials, including OADP, to use the new authentication method.

2. Set the environment variables for your Azure STS configuration as shown in the following example:

```
export API_URL=$(oc whoami --show-server) # Get cluster information
export CLUSTER_NAME=$(echo "$API_URL" | sed 's|https://api\..*||' | sed 's|\.|_|')
```

```

export CLUSTER_RESOURCE_GROUP="${CLUSTER_NAME}-rg"

# Get Azure information
export AZURE_SUBSCRIPTION_ID=$(az account show --query id -o tsv)
export AZURE_TENANT_ID=$(az account show --query tenantId -o tsv)

# Set names for resources
export IDENTITY_NAME="velero"
export APP_NAME="velero-${CLUSTER_NAME}"
export STORAGE_ACCOUNT_NAME=$(echo "velero${CLUSTER_NAME}" | tr -d '-' | tr
'[:upper:]' '[:lower:]' | cut -c1-24)
export CONTAINER_NAME="velero"

```

3. Create an Azure Managed Identity for OADP as shown in the following example:

```

az identity create \ # Create managed identity
  --subscription "$AZURE_SUBSCRIPTION_ID" \
  --resource-group "$CLUSTER_RESOURCE_GROUP" \
  --name "$IDENTITY_NAME"

# Get identity details
export IDENTITY_CLIENT_ID=$(az identity show -g "$CLUSTER_RESOURCE_GROUP" -n
"$IDENTITY_NAME" --query clientId -o tsv)
export IDENTITY_PRINCIPAL_ID=$(az identity show -g
"$CLUSTER_RESOURCE_GROUP" -n "$IDENTITY_NAME" --query principalId -o tsv)

```

4. Grant the required Azure roles to the managed identity as shown in the following example:

```

export SUBSCRIPTION_ID=$(az account show --query id -o tsv) # Get subscription ID for
role assignments

# Required roles for OADP operations
REQUIRED_ROLES=(
  "Contributor"
  "Storage Blob Data Contributor"
  "Disk Snapshot Contributor"
)

for role in "${REQUIRED_ROLES[@]"; do
  echo "Assigning role: $role"
  az role assignment create \
    --assignee "$IDENTITY_PRINCIPAL_ID" \
    --role "$role" \
    --scope "/subscriptions/$SUBSCRIPTION_ID"
done

```

5. Create an Azure storage account and a container as shown in the following example:

```

az storage account create \ # Create storage account
  --name "$STORAGE_ACCOUNT_NAME" \
  --resource-group "$CLUSTER_RESOURCE_GROUP" \
  --location "$(az group show -n $CLUSTER_RESOURCE_GROUP --query location -o tsv)" \
  --sku Standard_LRS \
  --kind StorageV2

```

6. Get the OIDC issuer URL from your OpenShift cluster as shown in the following example:

```
export SERVICE_ACCOUNT_ISSUER=$(oc get authentication.config.openshift.io cluster -o
json | jq -r .spec.serviceAccountIssuer)
echo "OIDC Issuer: $SERVICE_ACCOUNT_ISSUER"
```

7. Configure Microsoft Entra Workload ID Federation as shown in the following example:

```
az identity federated-credential create \ # Create federated identity credential for Velero
service account
  --name "velero-federated-credential" \
  --identity-name "$IDENTITY_NAME" \
  --resource-group "$CLUSTER_RESOURCE_GROUP" \
  --issuer "$SERVICE_ACCOUNT_ISSUER" \
  --subject "system:serviceaccount:openshift-adp:velero" \
  --audiences "openshift"

# Create federated identity credential for OADP controller manager
az identity federated-credential create \
  --name "oadp-controller-federated-credential" \
  --identity-name "$IDENTITY_NAME" \
  --resource-group "$CLUSTER_RESOURCE_GROUP" \
  --issuer "$SERVICE_ACCOUNT_ISSUER" \
  --subject "system:serviceaccount:openshift-adp:openshift-adp-controller-manager" \
  --audiences "openshift"
```

8. Create the OADP namespace if it does not already exist by running the following command:

```
oc create namespace openshift-adp
```

9. To use the **CloudStorage** CR to create an Azure cloud storage resource, run the following command:

```
cat <<EOF | oc apply -f -
apiVersion: oadp.openshift.io/v1alpha1
kind: CloudStorage
metadata:
  name: azure-backup-storage
  namespace: openshift-adp
spec:
  name: ${CONTAINER_NAME}
  provider: azure
  creationSecret:
    name: cloud-credentials-azure
    key: azurekey
  config:
    storageAccount: ${STORAGE_ACCOUNT_NAME}
EOF
```

10. Create the **DataProtectionApplication** (DPA) custom resource (CR) and configure the Azure STS details as shown in the following example:

```
cat <<EOF | oc apply -f -
apiVersion: oadp.openshift.io/v1alpha1
```

```
kind: DataProtectionApplication
metadata:
  name: dpa-azure-workload-id-cloudstorage
  namespace: openshift-adp
spec:
  backupLocations:
  - bucket:
      cloudStorageRef:
        name: <cloud_storage_cr>
      config:
        storageAccount: <storage_account_name>
        useAAD: "true"
      credential:
        key: azurekey
        name: cloud-credentials-azure
      default: true
      prefix: velero
      name: default
  configuration:
    velero:
      defaultPlugins:
      - azure
      - openshift
      - csi
      disableFsBackup: false
  logFormat: text
  snapshotLocations:
  - name: default
    velero:
      config:
        resourceGroup: <resource_group>
        subscriptionId: <subscription_ID>
      credential:
        key: azurekey
        name: cloud-credentials-azure
      provider: azure
EOF
```

where:

<cloud_storage_cr>

Specify the **CloudStorage** CR name.

<storage_account_name>

Specify the Azure storage account name.

<resource_group>

Specify the resource group.

<subscription_ID>

Specify the subscription ID.

Verification

1. Verify that the OADP operator pods are running:

```
$ oc get pods -n openshift-adp
```

2. Verify the Azure role assignments:

```
az role assignment list --assignee ${IDENTITY_PRINCIPAL_ID} --all --query "
[.roleDefinitionName]" -o tsv
```

3. Verify Microsoft Entra Workload ID authentication:

```
$ VELERO_POD=$(oc get pods -n openshift-adp -l app.kubernetes.io/name=velero -o
jsonpath='{.items[0].metadata.name}') # Check Velero pod environment variables

# Check AZURE_CLIENT_ID environment variable
$ oc get pod ${VELERO_POD} -n openshift-adp -o jsonpath='{.spec.containers[0].env[?
(@.name=="AZURE_CLIENT_ID")]}'
```

```
# Check AZURE_FEDERATED_TOKEN_FILE environment variable
$ oc get pod ${VELERO_POD} -n openshift-adp -o jsonpath='{.spec.containers[0].env[?
(@.name=="AZURE_FEDERATED_TOKEN_FILE")]}'
```

4. Create a backup of an application and verify the backup is stored successfully in Azure storage.

5.9.1.6. Setting Velero CPU and memory resource allocations

You set the CPU and memory resource allocations for the **Velero** pod by editing the **DataProtectionApplication** custom resource (CR) manifest.

Prerequisites

- You must have the OpenShift API for Data Protection (OADP) Operator installed.

Procedure

- Edit the values in the **spec.configuration.velero.podConfig.ResourceAllocations** block of the **DataProtectionApplication** CR manifest, as in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  # ...
  configuration:
    velero:
      podConfig:
        nodeSelector: <node_selector>
        resourceAllocations:
          limits:
            cpu: "1"
            memory: 1024Mi
          requests:
            cpu: 200m
            memory: 256Mi
```

where:

nodeSelector

Specifies the node selector to be supplied to Velero podSpec.

resourceAllocations

Specifies the resource allocations listed for average usage.



NOTE

Kopia is an option in OADP 1.3 and later releases. You can use Kopia for file system backups, and Kopia is your only option for Data Mover cases with the built-in Data Mover.

Kopia is more resource intensive than Restic, and you might need to adjust the CPU and memory requirements accordingly.

Use the **nodeSelector** field to select which nodes can run the node agent. The **nodeSelector** field is the simplest recommended form of node selection constraint. Any label specified must match the labels on each node.

For more details, see [Configuring node agents and node labels](#).

5.9.1.7. Enabling self-signed CA certificates

You must enable a self-signed CA certificate for object storage by editing the **DataProtectionApplication** custom resource (CR) manifest to prevent a **certificate signed by unknown authority** error.

Prerequisites

- You must have the OpenShift API for Data Protection (OADP) Operator installed.

Procedure

- Edit the **spec.backupLocations.velero.objectStorage.caCert** parameter and **spec.backupLocations.velero.config** parameters of the **DataProtectionApplication** CR manifest:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  # ...
  backupLocations:
    - name: default
      velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket>
          prefix: <prefix>
```



```

    caCert: <base64_encoded_cert_string>
  config:
    insecureSkipTLSVerify: "false"
# ...

```

where:

caCert

Specifies the Base64-encoded CA certificate string.

insecureSkipTLSVerify

Specifies the **insecureSkipTLSVerify** configuration. The configuration can be set to either **"true"** or **"false"**. If set to **"true"**, SSL/TLS security is disabled. If set to **"false"**, SSL/TLS security is enabled.

5.9.1.8. Using CA certificates with the velero command aliased for Velero deployment

You might want to use the Velero CLI without installing it locally on your system by creating an alias for it.

Prerequisites

- You must be logged in to the OpenShift Container Platform cluster as a user with the **cluster-admin** role.
- You must have the OpenShift CLI (**oc**) installed.

Procedure

1. To use an aliased Velero command, run the following command:

```
$ alias velero='oc -n openshift-adp exec deployment/velero -c velero -it -- ./velero'
```

2. Check that the alias is working by running the following command:

```
$ velero version
```

```

Client:
Version: v1.12.1-OADP
Git commit: -
Server:
Version: v1.12.1-OADP

```

3. To use a CA certificate with this command, you can add a certificate to the Velero deployment by running the following commands:

```
$ CA_CERT=$(oc -n openshift-adp get dataprotectionapplications.oadp.openshift.io <dpa-name> -o jsonpath='{.spec.backupLocations[0].velero.objectStorage.caCert}')
```

```
$ [[ -n $CA_CERT ]] && echo "$CA_CERT" | base64 -d | oc exec -n openshift-adp -i
deploy/velero -c velero -- bash -c "cat > /tmp/your-cacert.txt" || echo "DPA BSL has no
caCert"
```

```
$ velero describe backup <backup_name> --details --cacert /tmp/<your_cacert>.txt
```

4. To fetch the backup logs, run the following command:

```
$ velero backup logs <backup_name> --cacert /tmp/<your_cacert.txt>
```

You can use these logs to view failures and warnings for the resources that you cannot back up.

5. If the Velero pod restarts, the **/tmp/your-cacert.txt** file disappears, and you must re-create the **/tmp/your-cacert.txt** file by re-running the commands from the previous step.
6. You can check if the **/tmp/your-cacert.txt** file still exists, in the file location where you stored it, by running the following command:

```
$ oc exec -n openshift-adp -i deploy/velero -c velero -- bash -c "ls /tmp/your-cacert.txt"
/tmp/your-cacert.txt
```

In a future release of OpenShift API for Data Protection (OADP), we plan to mount the certificate to the Velero pod so that this step is not required.

5.9.1.9. Installing the Data Protection Application

You install the Data Protection Application (DPA) by creating an instance of the **DataProtectionApplication** API.

Prerequisites

- You must install the OADP Operator.
- You must configure object storage as a backup location.
- If you use snapshots to back up PVs, your cloud provider must support either a native snapshot API or Container Storage Interface (CSI) snapshots.
- If the backup and snapshot locations use the same credentials, you must create a **Secret** with the default name, **cloud-credentials-azure**.
- If the backup and snapshot locations use different credentials, you must create two **Secrets**:
 - **Secret** with a custom name for the backup location. You add this **Secret** to the **DataProtectionApplication** CR.
 - **Secret** with another custom name for the snapshot location. You add this **Secret** to the **DataProtectionApplication** CR.



NOTE

If you do not want to specify backup or snapshot locations during the installation, you can create a default **Secret** with an empty **credentials-velero** file. If there is no default **Secret**, the installation will fail.

Procedure

1. Click **Ecosystem → Installed Operators** and select the OADP Operator.

2. Under **Provided APIs**, click **Create instance** in the **DataProtectionApplication** box.
3. Click **YAML View** and update the parameters of the **DataProtectionApplication** manifest:

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  configuration:
    velero:
      defaultPlugins:
        - azure
        - openshift
      resourceTimeout: 10m
  nodeAgent:
    enable: true
    uploaderType: kopia
    podConfig:
      nodeSelector: <node_selector>
  backupLocations:
    - velero:
        config:
          resourceGroup: <azure_resource_group>
          storageAccount: <azure_storage_account_id>
          subscriptionId: <azure_subscription_id>
        credential:
          key: cloud
          name: cloud-credentials-azure
        provider: azure
        default: true
        objectStorage:
          bucket: <bucket_name>
          prefix: <prefix>
  snapshotLocations:
    - velero:
        config:
          resourceGroup: <azure_resource_group>
          subscriptionId: <azure_subscription_id>
          incremental: "true"
        name: default
        provider: azure
        credential:
          key: cloud
          name: cloud-credentials-azure

```

where:

namespace

Specifies the default namespace for OADP which is **openshift-adp**. The namespace is a variable and is configurable.

openshift

Specifies that the **openshift** plugin is mandatory.

resourceTimeout

Specifies how many minutes to wait for several Velero resources such as Velero CRD availability, volumeSnapshot deletion, and backup repository availability, before timeout occurs. The default is 10m.

nodeAgent

Specifies the administrative agent that routes the administrative requests to servers.

enable

Set this value to **true** if you want to enable **nodeAgent** and perform File System Backup.

uploaderType

Specifies the uploader type. Enter **kopia** or **restic** as your uploader. You cannot change the selection after the installation. For the Built-in DataMover you must use Kopia. The **nodeAgent** deploys a daemon set, which means that the **nodeAgent** pods run on each working node. You can configure File System Backup by adding **spec.defaultVolumesToFsBackup: true** to the **Backup** CR.

nodeSelector

Specifies the nodes on which Kopia or Restic are available. By default, Kopia or Restic run on all nodes.

resourceGroup

Specifies the Azure resource group.

storageAccount

Specifies the Azure storage account ID.

subscriptionId

Specifies the Azure subscription ID.

name

Specifies the name of the **Secret** object. If you do not specify this value, the default name, **cloud-credentials-azure**, is used. If you specify a custom name, the custom name is used for the backup location.

bucket

Specifies a bucket as the backup storage location. If the bucket is not a dedicated bucket for Velero backups, you must specify a prefix.

prefix

Specifies a prefix for Velero backups, for example, **velero**, if the bucket is used for multiple purposes.

snapshotLocations

Specifies the snapshot location. You do not need to specify a snapshot location if you use CSI snapshots or Restic to back up PVs.

name

Specifies the name of the **Secret** object that you created. If you do not specify this value, the default name, **cloud-credentials-azure**, is used. If you specify a custom name, the custom name is used for the backup location.

4. Click **Create**.

Verification

1. Verify the installation by viewing the OpenShift API for Data Protection (OADP) resources by running the following command:

```
$ oc get all -n openshift-adp
```

```
NAME                                READY STATUS  RESTARTS  AGE
pod/oadp-operator-controller-manager-67d9494d47-6l8z8  2/2   Running  0         2m8s
pod/node-agent-9cq4q                        1/1   Running  0         94s
pod/node-agent-m4lts                        1/1   Running  0         94s
pod/node-agent-pv4kr                        1/1   Running  0         95s
pod/velero-588db7f655-n842v                1/1   Running  0         95s

NAME                                TYPE          CLUSTER-IP    EXTERNAL-IP
PORT(S)  AGE
service/oadp-operator-controller-manager-metrics-service  ClusterIP    172.30.70.140
<none>      8443/TCP  2m8s
service/openshift-adp-velero-metrics-svc                  ClusterIP    172.30.10.0   <none>
8085/TCP  8h

NAME                                DESIRED  CURRENT  READY  UP-TO-DATE  AVAILABLE  NODE
SELECTOR  AGE
daemonset.apps/node-agent  3        3        3      3           3          <none>    96s

NAME                                READY  UP-TO-DATE  AVAILABLE  AGE
deployment.apps/oadp-operator-controller-manager  1/1    1           1          2m9s
deployment.apps/velero                          1/1    1           1          96s

NAME                                DESIRED  CURRENT  READY  AGE
replicaset.apps/oadp-operator-controller-manager-67d9494d47  1        1        1      2m9s
replicaset.apps/velero-588db7f655                        1        1        1      96s
```

2. Verify that the **DataProtectionApplication** (DPA) is reconciled by running the following command:

```
$ oc get dpa dpa-sample -n openshift-adp -o jsonpath='{.status}'
```

```
{"conditions":[{"lastTransitionTime":"2023-10-27T01:23:57Z","message":"Reconcile complete","reason":"Complete","status":"True","type":"Reconciled"]}]}
```

3. Verify the **type** is set to **Reconciled**.
4. Verify the backup storage location and confirm that the **PHASE** is **Available** by running the following command:

```
$ oc get backupstoragelocations.velero.io -n openshift-adp
```

```
NAME          PHASE    LAST VALIDATED  AGE    DEFAULT
dpa-sample-1  Available  1s             3d16h  true
```

5.9.1.10. Configuring the DPA with client burst and QPS settings

The burst setting determines how many requests can be sent to the **velero** server before the limit is applied. After the burst limit is reached, the queries per second (QPS) setting determines how many additional requests can be sent per second.

You can set the burst and QPS values of the **velero** server by configuring the Data Protection Application (DPA) with the burst and QPS values. You can use the **dpa.configuration.velero.client-burst** and **dpa.configuration.velero.client-qps** fields of the DPA to set the burst and QPS values.

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **client-burst** and the **client-qps** fields in the DPA as shown in the following example:

Example Data Protection Application

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: test-dpa
  namespace: openshift-adp
spec:
  backupLocations:
    - name: default
      velero:
        config:
          insecureSkipTLSVerify: "true"
          profile: "default"
          region: <bucket_region>
          s3ForcePathStyle: "true"
          s3Url: <bucket_url>
        credential:
          key: cloud
          name: cloud-credentials
        default: true
        objectStorage:
          bucket: <bucket_name>
          prefix: velero
        provider: aws
  configuration:
    nodeAgent:
      enable: true
      uploaderType: restic
    velero:
      client-burst: 500
      client-qps: 300
      defaultPlugins:
        - openshift
        - aws
        - kubevirt
```

where:

client-burst

Specifies the **client-burst** value. In this example, the **client-burst** field is set to 500.

client-qps

Specifies the **client-qps** value. In this example, the **client-qps** field is set to 300.

5.9.1.11. Configuring node agents and node labels

The Data Protection Application (DPA) uses the **nodeSelector** field to select which nodes can run the node agent. The **nodeSelector** field is the recommended form of node selection constraint.

Procedure

1. Run the node agent on any node that you choose by adding a custom label:

```
$ oc label node/<node_name> node-role.kubernetes.io/nodeAgent=""
```

**NOTE**

Any label specified must match the labels on each node.

2. Use the same custom label in the **DPA.spec.configuration.nodeAgent.podConfig.nodeSelector** field, which you used for labeling nodes:

```
configuration:
  nodeAgent:
    enable: true
  podConfig:
    nodeSelector:
      node-role.kubernetes.io/nodeAgent: ""
```

The following example is an anti-pattern of **nodeSelector** and does not work unless both labels, **node-role.kubernetes.io/infra: ""** and **node-role.kubernetes.io/worker: ""**, are on the node:

```
configuration:
  nodeAgent:
    enable: true
  podConfig:
    nodeSelector:
      node-role.kubernetes.io/infra: ""
      node-role.kubernetes.io/worker: ""
```

5.9.1.12. Configuring node agent load affinity

You can schedule the node agent pods on specific nodes by using the **spec.podConfig.nodeSelector** object of the **DataProtectionApplication** (DPA) custom resource (CR).

See the following example in which you can schedule the node agent pods on nodes with the label **label.io/role: cpu-1** and **other-label.io/other-role: cpu-2**.

...

```
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
    podConfig:
      nodeSelector:
        label.io/role: cpu-1
        other-label.io/other-role: cpu-2
    ...
```

You can add more restrictions on the node agent pods scheduling by using the **nodeagent.loadAffinity** object in the DPA spec.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the DPA spec **nodeagent.loadAffinity** object as shown in the following example. In the example, you ensure that the node agent pods are scheduled only on nodes with the label **label.io/role: cpu-1** and the label **label.io/hostname** matching with either **node1** or **node2**.

```
...
spec:
  configuration:
    nodeAgent:
      enable: true
      loadAffinity:
        - nodeSelector:
            matchLabels:
              label.io/role: cpu-1
            matchExpressions:
              - key: label.io/hostname
                operator: In
                values:
                  - node1
                  - node2
        ...
```

where:

loadAffinity

Specifies the **loadAffinity** object by adding the **matchLabels** and **matchExpressions** objects.

matchExpressions

Specifies the **matchExpressions** object to add restrictions on the node agent pods scheduling.

5.9.1.13. Node agent load affinity guidelines

Use the following guidelines to configure the node agent **loadAffinity** object in the **DataProtectionApplication** (DPA) custom resource (CR).

- Use the **spec.nodeagent.podConfig.nodeSelector** object for simple node matching.
- Use the **loadAffinity.nodeSelector** object without the **podConfig.nodeSelector** object for more complex scenarios.
- You can use both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects, but the **loadAffinity** object must be equal or more restrictive as compared to the **podConfig** object. In this scenario, the **podConfig.nodeSelector** labels must be a subset of the labels used in the **loadAffinity.nodeSelector** object.
- You cannot use the **matchExpressions** and **matchLabels** fields if you have configured both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects in the DPA.
- See the following example to configure both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects in the DPA.

```
...
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
      loadAffinity:
        - nodeSelector:
            matchLabels:
              label.io/location: 'US'
              label.io/gpu: 'no'
      podConfig:
        nodeSelector:
          label.io/gpu: 'no'
```

5.9.1.14. Configuring node agent load concurrency

You can control the maximum number of node agent operations that can run simultaneously on each node within your cluster.

You can configure it using one of the following fields of the Data Protection Application (DPA):

- **globalConfig**: Defines a default concurrency limit for the node agent across all nodes.
- **perNodeConfig**: Specifies different concurrency limits for specific nodes based on **nodeSelector** labels. This provides flexibility for environments where certain nodes might have different resource capacities or roles.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.

Procedure

1. If you want to use load concurrency for specific nodes, add labels to those nodes:

```
$ oc label node/<node_name> label.io/instance-type='large'
```

2. Configure the load concurrency fields for your DPA instance:

```
configuration:
  nodeAgent:
    enable: true
    uploaderType: kopia
    loadConcurrency:
      globalConfig: 1
      perNodeConfig:
        - nodeSelector:
            matchLabels:
              label.io/instance-type: large
          number: 3
```

where:

globalConfig

Specifies the global concurrent number. The default value is 1, which means there is no concurrency and only one load is allowed. The **globalConfig** value does not have a limit.

label.io/instance-type

Specifies the label for per-node concurrency.

number

Specifies the per-node concurrent number. You can specify many per-node concurrent numbers, for example, based on the instance type and size. The range of per-node concurrent number is the same as the global concurrent number. If the configuration file contains a per-node concurrent number and a global concurrent number, the per-node concurrent number takes precedence.

5.9.1.15. Configuring the node agent as a non-root and non-privileged user

To enhance the node agent security, you can configure the OADP Operator node agent daemonset to run as a non-root and non-privileged user by using the **spec.configuration.velero.disableFsBackup** setting in the **DataProtectionApplication** (DPA) custom resource (CR).

By setting the **spec.configuration.velero.disableFsBackup** setting to **true**, the node agent security context sets the root file system to read-only and sets the **privileged** flag to **false**.



NOTE

Setting **spec.configuration.velero.disableFsBackup** to **true** enhances the node agent security by removing the need for privileged containers and enforcing a read-only root file system.

However, it also disables File System Backup (FSB) with Kopia. If your workloads rely on FSB for backing up volumes that do not support native snapshots, then you should evaluate whether the **disableFsBackup** configuration fits your use case.

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **disableFsBackup** field in the DPA as shown in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: ts-dpa
  namespace: openshift-adp
spec:
  backupLocations:
  - velero:
      credential:
        key: cloud
        name: cloud-credentials
      default: true
      objectStorage:
        bucket: <bucket_name>
        prefix: velero
      provider: gcp
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
    velero:
      defaultPlugins:
      - csi
      - gcp
      - openshift
      disableFsBackup: true
```

where:

nodeAgent

Specifies to enable the node agent in the DPA.

disableFsBackup

Specifies to set the **disableFsBackup** field to **true**.

Verification

1. Verify that the node agent security context is set to run as non-root and the root file system is **readOnly** by running the following command:

```
$ oc get daemonset node-agent -o yaml
```

The example output is as following:

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  ...
  name: node-agent
```

```

    namespace: openshift-adp
    ...
spec:
  ...
  template:
    metadata:
      ...
    spec:
      containers:
        ...
        securityContext:
          allowPrivilegeEscalation: false
          capabilities:
            drop:
              - ALL
          privileged: false
          readOnlyRootFilesystem: true
        ...
      nodeSelector:
        kubernetes.io/os: linux
      os:
        name: linux
      restartPolicy: Always
      schedulerName: default-scheduler
      securityContext:
        runAsNonRoot: true
        seccompProfile:
          type: RuntimeDefault
      serviceAccount: velero
      serviceAccountName: velero
      ....

```

where:

allowPrivilegeEscalation

Specifies that the **allowPrivilegeEscalation** field is false.

privileged

Specifies that the **privileged** field is false.

readOnlyRootFilesystem

Specifies that the root file system is read-only.

runAsNonRoot

Specifies that the node agent is run as a non-root user.

5.9.1.16. Configuring repository maintenance

OADP repository maintenance is a background job, you can configure it independently of the node agent pods. This means that you can schedule the repository maintenance pod on a node where the node agent is or is not running.

You can use the repository maintenance job affinity configurations in the **DataProtectionApplication** (DPA) custom resource (CR) only if you use Kopia as the backup repository.

You have the option to configure the load affinity at the global level affecting all repositories. Or you can configure the load affinity for each repository. You can also use a combination of global and per-repository configuration.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the **loadAffinity** object in the DPA spec by using either one or both of the following methods:
 - Global configuration: Configure load affinity for all repositories as shown in the following example:

```
...
spec:
  configuration:
    repositoryMaintenance:
      global:
        podResources:
          cpuRequest: "100m"
          cpuLimit: "200m"
          memoryRequest: "100Mi"
          memoryLimit: "200Mi"
        loadAffinity:
          - nodeSelector:
              matchLabels:
                label.io/gpu: 'no'
            matchExpressions:
              - key: label.io/location
                operator: In
                values:
                  - US
                  - EU
```

where:

repositoryMaintenance

Specifies the **repositoryMaintenance** object as shown in the example.

global

Specifies the **global** object to configure load affinity for all repositories.

- Per-repository configuration: Configure load affinity per repository as shown in the following example:

```
...
spec:
  configuration:
    repositoryMaintenance:
```

```

myrepositoryname:
  loadAffinity:
    - nodeSelector:
        matchLabels:
            label.io/cpu: 'yes'

```

where:

myrepositoryname

Specifies the **repositoryMaintenance** object for each repository.

5.9.1.17. Configuring Velero load affinity

With each OADP deployment, there is one Velero pod and its main purpose is to schedule Velero workloads. To schedule the Velero pod, you can use the **velero.podConfig.nodeSelector** and the **velero.loadAffinity** objects in the **DataProtectionApplication** (DPA) custom resource (CR) spec.

Use the **podConfig.nodeSelector** object to assign the Velero pod to specific nodes. You can also configure the **velero.loadAffinity** object for pod-level affinity and anti-affinity.

The OpenShift scheduler applies the rules and performs the scheduling of the Velero pod deployment.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the **velero.podConfig.nodeSelector** and the **velero.loadAffinity** objects in the DPA spec as shown in the following examples:

- **velero.podConfig.nodeSelector** object configuration:

```

...
spec:
  configuration:
    velero:
      podConfig:
        nodeSelector:
            some-label.io/custom-node-role: backup-core

```

- **velero.loadAffinity** object configuration:

```

...
spec:
  configuration:
    velero:
      loadAffinity:
        - nodeSelector:
            matchLabels:

```

```

label.io/gpu: 'no'
matchExpressions:
- key: label.io/location
  operator: In
  values:
  - US
  - EU

```

5.9.1.18. Overriding the imagePullPolicy setting in the DPA

In OADP 1.4.0 or earlier, the Operator sets the **imagePullPolicy** field of the Velero and node agent pods to **Always** for all images.

In OADP 1.4.1 or later, the Operator first checks if each image has the **sha256** or **sha512** digest and sets the **imagePullPolicy** field accordingly:

- If the image has the digest, the Operator sets **imagePullPolicy** to **IfNotPresent**.
- If the image does not have the digest, the Operator sets **imagePullPolicy** to **Always**.

You can also override the **imagePullPolicy** field by using the **spec.imagePullPolicy** field in the Data Protection Application (DPA).

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **spec.imagePullPolicy** field in the DPA as shown in the following example:

Example Data Protection Application

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: test-dpa
  namespace: openshift-adp
spec:
  backupLocations:
  - name: default
    velero:
      config:
        insecureSkipTLSVerify: "true"
        profile: "default"
        region: <bucket_region>
        s3ForcePathStyle: "true"
        s3Url: <bucket_url>
      credential:
        key: cloud
        name: cloud-credentials
      default: true
    objectStorage:
      bucket: <bucket_name>
      prefix: velero

```

```

    provider: aws
  configuration:
    nodeAgent:
      enable: true
    uploaderType: kopia
  velero:
    defaultPlugins:
      - openshift
      - aws
      - kubevirt
      - csi
  imagePullPolicy: Never

```

where:

imagePullPolicy

Specifies the value for **imagePullPolicy**. In this example, the **imagePullPolicy** field is set to **Never**.

5.9.1.18.1. Enabling CSI in the DataProtectionApplication CR

You enable the Container Storage Interface (CSI) in the **DataProtectionApplication** custom resource (CR) in order to back up persistent volumes with CSI snapshots.

Prerequisites

- The cloud provider must support CSI snapshots.

Procedure

- Edit the **DataProtectionApplication** CR, as in the following example:

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - csi

```

where:

csi

Specifies the **csi** default plugin.

5.9.1.18.2. Disabling the node agent in DataProtectionApplication

If you are not using **Restic**, **Kopia**, or **DataMover** for your backups, you can disable the **nodeAgent** field in the **DataProtectionApplication** custom resource (CR). Before you disable **nodeAgent**, ensure the OADP Operator is idle and not running any backups.

Procedure

1. To disable the **nodeAgent**, set the **enable** flag to **false**. See the following example:

Example DataProtectionApplication CR

```
# ...
configuration:
  nodeAgent:
    enable: false
    uploaderType: kopia
# ...
```

where:

enable

Enables the node agent.

2. To enable the **nodeAgent**, set the **enable** flag to **true**. See the following example:

Example DataProtectionApplication CR

```
# ...
configuration:
  nodeAgent:
    enable: true
    uploaderType: kopia
# ...
```

where:

enable

Enables the node agent.

You can set up a job to enable and disable the **nodeAgent** field in the **DataProtectionApplication** CR. For more information, see "Running tasks in pods using jobs".

Additional resources

- [Installing the Data Protection Application with the **kubevirt** and **openshift** plugins](#)
- [Running tasks in pods using jobs](#).
- [Configuring the OpenShift API for Data Protection \(OADP\) with multiple backup storage locations](#)

5.10. CONFIGURING OADP WITH GOOGLE CLOUD

5.10.1. Configuring the OpenShift API for Data Protection with Google Cloud

You install the OpenShift API for Data Protection (OADP) with Google Cloud by installing the OADP Operator. The Operator installs [Velero 1.16](#).

**NOTE**

Starting from OADP 1.0.4, all OADP 1.0.z versions can only be used as a dependency of the Migration Toolkit for Containers Operator and are not available as a standalone Operator.

You configure Google Cloud for Velero, create a default **Secret**, and then install the Data Protection Application. For more details, see [Installing the OADP Operator](#).

To install the OADP Operator in a restricted network environment, you must first disable the default software catalog sources and mirror the Operator catalog. See [Using Operator Lifecycle Manager in disconnected environments](#) for details.

5.10.1.1. Configuring Google Cloud

You configure Google Cloud for the OpenShift API for Data Protection (OADP).

Prerequisites

- You must have the **gcloud** and **gsutil** CLI tools installed. See the [Google cloud documentation](#) for details.

Procedure

1. Log in to Google Cloud:

```
$ gcloud auth login
```

2. Set the **BUCKET** variable:

```
$ BUCKET=<bucket> 1
```

- 1 Specify your bucket name.

3. Create the storage bucket:

```
$ gsutil mb gs://$BUCKET/
```

4. Set the **PROJECT_ID** variable to your active project:

```
$ PROJECT_ID=$(gcloud config get-value project)
```

5. Create a service account:

```
$ gcloud iam service-accounts create velero \
  --display-name "Velero service account"
```

6. List your service accounts:

```
$ gcloud iam service-accounts list
```

7. Set the **SERVICE_ACCOUNT_EMAIL** variable to match its **email** value:

```
$ SERVICE_ACCOUNT_EMAIL=$(gcloud iam service-accounts list \
--filter="displayName:Velero service account" \
--format 'value(email)')
```

8. Attach the policies to give the **velero** user the minimum necessary permissions:

```
$ ROLE_PERMISSIONS=(
  compute.disks.get
  compute.disks.create
  compute.disks.createSnapshot
  compute.snapshots.get
  compute.snapshots.create
  compute.snapshots.useReadOnly
  compute.snapshots.delete
  compute.zones.get
  storage.objects.create
  storage.objects.delete
  storage.objects.get
  storage.objects.list
  iam.serviceAccounts.signBlob
)
```

9. Create the **velero.server** custom role:

```
$ gcloud iam roles create velero.server \
--project $PROJECT_ID \
--title "Velero Server" \
--permissions "${IFS=","; echo "${ROLE_PERMISSIONS[*]}")"
```

10. Add IAM policy binding to the project:

```
$ gcloud projects add-iam-policy-binding $PROJECT_ID \
--member serviceAccount:$SERVICE_ACCOUNT_EMAIL \
--role projects/$PROJECT_ID/roles/velero.server
```

11. Update the IAM service account:

```
$ gsutil iam ch serviceAccount:$SERVICE_ACCOUNT_EMAIL:objectAdmin gs://${BUCKET}
```

12. Save the IAM service account keys to the **credentials-velero** file in the current directory:

```
$ gcloud iam service-accounts keys create credentials-velero \
--iam-account $SERVICE_ACCOUNT_EMAIL
```

You use the **credentials-velero** file to create a **Secret** object for Google Cloud before you install the Data Protection Application.

5.10.1.2. About backup and snapshot locations and their secrets

Review backup location, snapshot location, and secret configuration requirements for the **DataProtectionApplication** custom resource (CR). This helps you understand storage options and credential management for data protection operations.

5.10.1.2.1. Backup locations

You can specify one of the following AWS S3-compatible object storage solutions as a backup location:

- Multicloud Object Gateway (MCG)
- Red Hat Container Storage
- Ceph RADOS Gateway; also known as Ceph Object Gateway
- Red Hat OpenShift Data Foundation
- MinIO

Velero backs up OpenShift Container Platform resources, Kubernetes objects, and internal images as an archive file on object storage.

5.10.1.2.2. Snapshot locations

If you use your cloud provider's native snapshot API to back up persistent volumes, you must specify the cloud provider as the snapshot location.

If you use Container Storage Interface (CSI) snapshots, you do not need to specify a snapshot location because you will create a **VolumeSnapshotClass** CR to register the CSI driver.

If you use File System Backup (FSB), you do not need to specify a snapshot location because FSB backs up the file system on object storage.

5.10.1.2.3. Secrets

If the backup and snapshot locations use the same credentials or if you do not require a snapshot location, you create a default **Secret**.

If the backup and snapshot locations use different credentials, you create two secret objects:

- Custom **Secret** for the backup location, which you specify in the **DataProtectionApplication** CR.
- Default **Secret** for the snapshot location, which is not referenced in the **DataProtectionApplication** CR.



IMPORTANT

The Data Protection Application requires a default **Secret**. Otherwise, the installation will fail.

If you do not want to specify backup or snapshot locations during the installation, you can create a default **Secret** with an empty **credentials-velero** file.

5.10.1.2.4. Creating a default Secret

You create a default **Secret** if your backup and snapshot locations use the same credentials or if you do not require a snapshot location.

The default name of the **Secret** is **cloud-credentials-gcp**.



NOTE

The **DataProtectionApplication** custom resource (CR) requires a default **Secret**. Otherwise, the installation will fail. If the name of the backup location **Secret** is not specified, the default name is used.

If you do not want to use the backup location credentials during the installation, you can create a **Secret** with the default name by using an empty **credentials-velero** file.

Prerequisites

- Your object storage and cloud storage, if any, must use the same credentials.
- You must configure object storage for Velero.

Procedure

1. Create a **credentials-velero** file for the backup storage location in the appropriate format for your cloud provider.
2. Create a **Secret** custom resource (CR) with the default name:

```
$ oc create secret generic cloud-credentials-gcp -n openshift-adp --from-file
cloud=credentials-velero
```

The **Secret** is referenced in the **spec.backupLocations.credential** block of the **DataProtectionApplication** CR when you install the Data Protection Application.

5.10.1.2.5. Creating secrets for different credentials

If your backup and snapshot locations use different credentials, you must create two **Secret** objects:

- Backup location **Secret** with a custom name. The custom name is specified in the **spec.backupLocations** block of the **DataProtectionApplication** custom resource (CR).
- Snapshot location **Secret** with the default name, **cloud-credentials-gcp**. This **Secret** is not specified in the **DataProtectionApplication** CR.

Procedure

1. Create a **credentials-velero** file for the snapshot location in the appropriate format for your cloud provider.
2. Create a **Secret** for the snapshot location with the default name:

```
$ oc create secret generic cloud-credentials-gcp -n openshift-adp --from-file
cloud=credentials-velero
```

3. Create a **credentials-velero** file for the backup location in the appropriate format for your object storage.
4. Create a **Secret** for the backup location with a custom name:

```
$ oc create secret generic <custom_secret> -n openshift-adp --from-file cloud=credentials-velero
```

5. Add the **Secret** with the custom name to the **DataProtectionApplication** CR, as in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  ...
  backupLocations:
    - velero:
        provider: gcp
        default: true
        credential:
          key: cloud
          name: <custom_secret> 1
        objectStorage:
          bucket: <bucket_name>
          prefix: <prefix>
  snapshotLocations:
    - velero:
        provider: gcp
        default: true
        config:
          project: <project>
          snapshotLocation: us-west1
```

1 Backup location **Secret** with custom name.

5.10.1.3. Configuring the Data Protection Application

You can configure the Data Protection Application by setting Velero resource allocations or enabling self-signed CA certificates.

5.10.1.3.1. Setting Velero CPU and memory resource allocations

You set the CPU and memory resource allocations for the **Velero** pod by editing the **DataProtectionApplication** custom resource (CR) manifest.

Prerequisites

- You must have the OpenShift API for Data Protection (OADP) Operator installed.

Procedure

- Edit the values in the **spec.configuration.velero.podConfig.ResourceAllocations** block of the **DataProtectionApplication** CR manifest, as in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  # ...
  configuration:
    velero:
      podConfig:
        nodeSelector: <node_selector>
        resourceAllocations:
          limits:
            cpu: "1"
            memory: 1024Mi
          requests:
            cpu: 200m
            memory: 256Mi
```

where:

nodeSelector

Specifies the node selector to be supplied to Velero podSpec.

resourceAllocations

Specifies the resource allocations listed for average usage.



NOTE

Kopia is an option in OADP 1.3 and later releases. You can use Kopia for file system backups, and Kopia is your only option for Data Mover cases with the built-in Data Mover.

Kopia is more resource intensive than Restic, and you might need to adjust the CPU and memory requirements accordingly.

Use the **nodeSelector** field to select which nodes can run the node agent. The **nodeSelector** field is the simplest recommended form of node selection constraint. Any label specified must match the labels on each node.

For more details, see [Configuring node agents and node labels](#).

5.10.1.3.2. Enabling self-signed CA certificates

You must enable a self-signed CA certificate for object storage by editing the **DataProtectionApplication** custom resource (CR) manifest to prevent a **certificate signed by unknown authority** error.

Prerequisites

- You must have the OpenShift API for Data Protection (OADP) Operator installed.

Procedure

- Edit the **spec.backupLocations.velero.objectStorage.caCert** parameter and **spec.backupLocations.velero.config** parameters of the **DataProtectionApplication** CR manifest:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  # ...
  backupLocations:
    - name: default
      velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket>
          prefix: <prefix>
          caCert: <base64_encoded_cert_string>
        config:
          insecureSkipTLSVerify: "false"
  # ...
```

where:

caCert

Specifies the Base64-encoded CA certificate string.

insecureSkipTLSVerify

Specifies the **insecureSkipTLSVerify** configuration. The configuration can be set to either **"true"** or **"false"**. If set to **"true"**, SSL/TLS security is disabled. If set to **"false"**, SSL/TLS security is enabled.

5.10.1.3.3. Using CA certificates with the velero command aliased for Velero deployment

You might want to use the Velero CLI without installing it locally on your system by creating an alias for it.

Prerequisites

- You must be logged in to the OpenShift Container Platform cluster as a user with the **cluster-admin** role.
- You must have the OpenShift CLI (**oc**) installed.

Procedure

1. To use an aliased Velero command, run the following command:

```
$ alias velero='oc -n openshift-adp exec deployment/velero -c velero -it -- ./velero'
```

2. Check that the alias is working by running the following command:

—


```
$ velero version
```

```
Client:
Version: v1.12.1-OADP
Git commit: -
Server:
Version: v1.12.1-OADP
```

3. To use a CA certificate with this command, you can add a certificate to the Velero deployment by running the following commands:

```
$ CA_CERT=$(oc -n openshift-adp get dataprotectionapplications.oadp.openshift.io <dpa-name> -o jsonpath='{.spec.backupLocations[0].velero.objectStorage.caCert}')
```

```
$ [[ -n $CA_CERT ]] && echo "$CA_CERT" | base64 -d | oc exec -n openshift-adp -i deploy/velero -c velero -- bash -c "cat > /tmp/your-cacert.txt" || echo "DPA BSL has no caCert"
```

```
$ velero describe backup <backup_name> --details --cacert /tmp/<your_cacert>.txt
```

4. To fetch the backup logs, run the following command:

```
$ velero backup logs <backup_name> --cacert /tmp/<your_cacert.txt>
```

You can use these logs to view failures and warnings for the resources that you cannot back up.

5. If the Velero pod restarts, the **/tmp/your-cacert.txt** file disappears, and you must re-create the **/tmp/your-cacert.txt** file by re-running the commands from the previous step.
6. You can check if the **/tmp/your-cacert.txt** file still exists, in the file location where you stored it, by running the following command:

```
$ oc exec -n openshift-adp -i deploy/velero -c velero -- bash -c "ls /tmp/your-cacert.txt" /tmp/your-cacert.txt
```

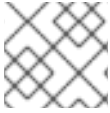
In a future release of OpenShift API for Data Protection (OADP), we plan to mount the certificate to the Velero pod so that this step is not required.

5.10.1.4. Google workload identity federation cloud authentication

Applications running outside Google Cloud use service account keys, such as usernames and passwords, to gain access to Google Cloud resources. These service account keys might become a security risk if they are not properly managed.

With Google's workload identity federation, you can use Identity and Access Management (IAM) to offer IAM roles, including the ability to impersonate service accounts, to external identities. This eliminates the maintenance and security risks associated with service account keys.

Workload identity federation handles encrypting and decrypting certificates, extracting user attributes, and validation. Identity federation externalizes authentication, passing it over to Security Token Services (STS), and reduces the demands on individual developers. Authorization and controlling access to resources remain the responsibility of the application.

**NOTE**

Google workload identity federation is available for OADP 1.3.x and later.

When backing up volumes, OADP on Google Cloud with Google workload identity federation authentication only supports CSI snapshots.

OADP on Google Cloud with Google workload identity federation authentication does not support Volume Snapshot Locations (VSL) backups. For more details, see [Google workload identity federation known issues](#).

If you do not use Google workload identity federation cloud authentication, continue to *Installing the Data Protection Application*.

Prerequisites

- You have installed a cluster in manual mode with [Google Cloud Workload Identity configured](#).
- You have access to the Cloud Credential Operator utility (**ccctl**) and to the associated workload identity pool.

Procedure

1. Create an **oadp-credrequest** directory by running the following command:

```
$ mkdir -p oadp-credrequest
```

2. Create a **CredentialsRequest.yaml** file as following:

```
echo 'apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: oadp-operator-credentials
  namespace: openshift-cloud-credential-operator
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: GCPProviderSpec
    permissions:
      - compute.disks.get
      - compute.disks.create
      - compute.disks.createSnapshot
      - compute.snapshots.get
      - compute.snapshots.create
      - compute.snapshots.useReadOnly
      - compute.snapshots.delete
      - compute.zones.get
      - storage.objects.create
      - storage.objects.delete
      - storage.objects.get
      - storage.objects.list
      - iam.serviceAccounts.signBlob
    skipServiceCheck: true
  secretRef:
    name: cloud-credentials-gcp'
```

```

    namespace: <OPERATOR_INSTALL_NS>
    serviceAccountNames:
    - velero
'> oadp-credrequest/credrequest.yaml

```

3. Use the **ccoctl** utility to process the **CredentialsRequest** objects in the **oadp-credrequest** directory by running the following command:

```

$ ccoctl gcp create-service-accounts \
  --name=<name> \
  --project=<gcp_project_id> \
  --credentials-requests-dir=oadp-credrequest \
  --workload-identity-pool=<pool_id> \
  --workload-identity-provider=<provider_id>

```

The **manifests/openshift-adp-cloud-credentials-gcp-credentials.yaml** file is now available to use in the following steps.

4. Create a namespace by running the following command:

```
$ oc create namespace <OPERATOR_INSTALL_NS>
```

5. Apply the credentials to the namespace by running the following command:

```
$ oc apply -f manifests/openshift-adp-cloud-credentials-gcp-credentials.yaml
```

5.10.1.4.1. Google workload identity federation known issues

- Volume Snapshot Location (VSL) backups finish with a **PartiallyFailed** phase when Google Cloud workload identity federation is configured. Google workload identity federation authentication does not support VSL backups.

5.10.1.5. Installing the Data Protection Application

You install the Data Protection Application (DPA) by creating an instance of the **DataProtectionApplication** API.

Prerequisites

- You must install the OADP Operator.
- You must configure object storage as a backup location.
- If you use snapshots to back up PVs, your cloud provider must support either a native snapshot API or Container Storage Interface (CSI) snapshots.
- If the backup and snapshot locations use the same credentials, you must create a **Secret** with the default name, **cloud-credentials-gcp**.
- If the backup and snapshot locations use different credentials, you must create two **Secrets**:
 - **Secret** with a custom name for the backup location. You add this **Secret** to the **DataProtectionApplication** CR.

- **Secret** with another custom name for the snapshot location. You add this **Secret** to the **DataProtectionApplication** CR.



NOTE

If you do not want to specify backup or snapshot locations during the installation, you can create a default **Secret** with an empty **credentials-velero** file. If there is no default **Secret**, the installation will fail.

Procedure

1. Click **Ecosystem** → **Installed Operators** and select the OADP Operator.
2. Under **Provided APIs**, click **Create instance** in the **DataProtectionApplication** box.
3. Click **YAML View** and update the parameters of the **DataProtectionApplication** manifest:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: <OPERATOR_INSTALL_NS>
spec:
  configuration:
    velero:
      defaultPlugins:
        - gcp
        - openshift
      resourceTimeout: 10m
  nodeAgent:
    enable: true
    uploaderType: kopia
    podConfig:
      nodeSelector: <node_selector>
  backupLocations:
    - velero:
        provider: gcp
        default: true
        credential:
          key: cloud
          name: cloud-credentials-gcp
        objectStorage:
          bucket: <bucket_name>
          prefix: <prefix>
  snapshotLocations:
    - velero:
        provider: gcp
        default: true
        config:
          project: <project>
          snapshotLocation: us-west1
        credential:
          key: cloud
          name: cloud-credentials-gcp
  backupImages: true
```

where:

namespace

Specifies the default namespace for OADP which is **openshift-adp**. The namespace is a variable and is configurable.

openshift

Specifies that the **openshift** plugin is mandatory.

resourceTimeout

Specifies how many minutes to wait for several Velero resources such as Velero CRD availability, volumeSnapshot deletion, and backup repository availability, before timeout occurs. The default is 10m.

nodeAgent

Specifies the administrative agent that routes the administrative requests to servers.

enable

Set this value to **true** if you want to enable **nodeAgent** and perform File System Backup.

uploaderType

Specifies the uploader type. Enter **kopia** or **restic** as your uploader. You cannot change the selection after the installation. For the Built-in DataMover you must use Kopia. The **nodeAgent** deploys a daemon set, which means that the **nodeAgent** pods run on each working node. You can configure File System Backup by adding **spec.defaultVolumesToFsBackup: true** to the **Backup** CR.

nodeSelector

Specifies the nodes on which Kopia or Restic are available. By default, Kopia or Restic run on all nodes.

key

Specifies the secret key that contains credentials. For Google workload identity federation cloud authentication use **service_account.json**.

name

Specifies the secret name that contains credentials. If you do not specify this value, the default name, **cloud-credentials-gcp**, is used.

bucket

Specifies a bucket as the backup storage location. If the bucket is not a dedicated bucket for Velero backups, you must specify a prefix.

prefix

Specifies a prefix for Velero backups, for example, **velero**, if the bucket is used for multiple purposes.

snapshotLocations

Specifies a snapshot location, unless you use CSI snapshots or Restic to back up PVs.

snapshotLocation

Specifies that the snapshot location must be in the same region as the PVs.

name

Specifies the name of the **Secret** object that you created. If you do not specify this value, the default name, **cloud-credentials-gcp**, is used. If you specify a custom name, the custom name is used for the backup location.

backupImages

Specifies that Google workload identity federation supports internal image backup. Set this field to **false** if you do not want to use image backup.

4. Click **Create**.

Verification

1. Verify the installation by viewing the OpenShift API for Data Protection (OADP) resources by running the following command:

```
$ oc get all -n openshift-adp
```

NAME	READY	STATUS	RESTARTS	AGE
pod/oadp-operator-controller-manager-67d9494d47-6l8z8	2/2	Running	0	2m8s
pod/node-agent-9cq4q	1/1	Running	0	94s
pod/node-agent-m4lts	1/1	Running	0	94s
pod/node-agent-pv4kr	1/1	Running	0	95s
pod/velero-588db7f655-n842v	1/1	Running	0	95s

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP
service/oadp-operator-controller-manager-metrics-service	ClusterIP	172.30.70.140	
service/openshift-adp-velero-metrics-svc	ClusterIP	172.30.10.0	<none>

NAME	DESIRED	CURRENT	READY	UP-TO-DATE	AVAILABLE	NODE
daemonset.apps/node-agent	3	3	3	3	<none>	96s

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
deployment.apps/oadp-operator-controller-manager	1/1	1	1	2m9s
deployment.apps/velero	1/1	1	1	96s

NAME	DESIRED	CURRENT	READY	AGE
replicaset.apps/oadp-operator-controller-manager-67d9494d47	1	1	1	2m9s
replicaset.apps/velero-588db7f655	1	1	1	96s

2. Verify that the **DataProtectionApplication** (DPA) is reconciled by running the following command:

```
$ oc get dpa dpa-sample -n openshift-adp -o jsonpath='{.status}'
```

```
{"conditions":[{"lastTransitionTime":"2023-10-27T01:23:57Z","message":"Reconcile complete","reason":"Complete","status":"True","type":"Reconciled"]]}
```

3. Verify the **type** is set to **Reconciled**.
4. Verify the backup storage location and confirm that the **PHASE** is **Available** by running the following command:

```
$ oc get backupstoragelocations.velero.io -n openshift-adp
```

NAME	PHASE	LAST VALIDATED	AGE	DEFAULT
dpa-sample-1	Available	1s	3d16h	true

5.10.1.6. Configuring the DPA with client burst and QPS settings

The burst setting determines how many requests can be sent to the **velero** server before the limit is applied. After the burst limit is reached, the queries per second (QPS) setting determines how many additional requests can be sent per second.

You can set the burst and QPS values of the **velero** server by configuring the Data Protection Application (DPA) with the burst and QPS values. You can use the **dpa.configuration.velero.client-burst** and **dpa.configuration.velero.client-qps** fields of the DPA to set the burst and QPS values.

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **client-burst** and the **client-qps** fields in the DPA as shown in the following example:

Example Data Protection Application

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: test-dpa
  namespace: openshift-adp
spec:
  backupLocations:
    - name: default
  velero:
    config:
      insecureSkipTLSVerify: "true"
      profile: "default"
      region: <bucket_region>
      s3ForcePathStyle: "true"
      s3Url: <bucket_url>
    credential:
      key: cloud
      name: cloud-credentials
    default: true
    objectStorage:
      bucket: <bucket_name>
      prefix: velero
    provider: aws
  configuration:
    nodeAgent:
      enable: true
      uploaderType: restic
  velero:
    client-burst: 500
    client-qps: 300
```

```
defaultPlugins:
- openshift
- aws
- kubevirt
```

where:

client-burst

Specifies the **client-burst** value. In this example, the **client-burst** field is set to 500.

client-qps

Specifies the **client-qps** value. In this example, the **client-qps** field is set to 300.

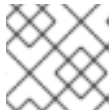
5.10.1.7. Configuring node agents and node labels

The Data Protection Application (DPA) uses the **nodeSelector** field to select which nodes can run the node agent. The **nodeSelector** field is the recommended form of node selection constraint.

Procedure

1. Run the node agent on any node that you choose by adding a custom label:

```
$ oc label node/<node_name> node-role.kubernetes.io/nodeAgent=""
```



NOTE

Any label specified must match the labels on each node.

2. Use the same custom label in the **DPA.spec.configuration.nodeAgent.podConfig.nodeSelector** field, which you used for labeling nodes:

```
configuration:
  nodeAgent:
    enable: true
  podConfig:
    nodeSelector:
      node-role.kubernetes.io/nodeAgent: ""
```

The following example is an anti-pattern of **nodeSelector** and does not work unless both labels, **node-role.kubernetes.io/infra: ""** and **node-role.kubernetes.io/worker: ""**, are on the node:

```
configuration:
  nodeAgent:
    enable: true
  podConfig:
    nodeSelector:
      node-role.kubernetes.io/infra: ""
      node-role.kubernetes.io/worker: ""
```

5.10.1.8. Configuring node agent load affinity

You can schedule the node agent pods on specific nodes by using the **spec.podConfig.nodeSelector** object of the **DataProtectionApplication** (DPA) custom resource (CR).

See the following example in which you can schedule the node agent pods on nodes with the label **label.io/role: cpu-1** and **other-label.io/other-role: cpu-2**.

```
...
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
      podConfig:
        nodeSelector:
          label.io/role: cpu-1
          other-label.io/other-role: cpu-2
    ...
```

You can add more restrictions on the node agent pods scheduling by using the **nodeagent.loadAffinity** object in the DPA spec.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the DPA spec **nodeagent.loadAffinity** object as shown in the following example. In the example, you ensure that the node agent pods are scheduled only on nodes with the label **label.io/role: cpu-1** and the label **label.io/hostname** matching with either **node1** or **node2**.

```
...
spec:
  configuration:
    nodeAgent:
      enable: true
      loadAffinity:
        - nodeSelector:
            matchLabels:
              label.io/role: cpu-1
            matchExpressions:
              - key: label.io/hostname
                operator: In
                values:
                  - node1
                  - node2
        ...
```

where:

loadAffinity

Specifies the **loadAffinity** object by adding the **matchLabels** and **matchExpressions** objects.

matchExpressions

Specifies the **matchExpressions** object to add restrictions on the node agent pods scheduling.

5.10.1.9. Node agent load affinity guidelines

Use the following guidelines to configure the node agent **loadAffinity** object in the **DataProtectionApplication** (DPA) custom resource (CR).

- Use the **spec.nodeagent.podConfig.nodeSelector** object for simple node matching.
- Use the **loadAffinity.nodeSelector** object without the **podConfig.nodeSelector** object for more complex scenarios.
- You can use both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects, but the **loadAffinity** object must be equal or more restrictive as compared to the **podConfig** object. In this scenario, the **podConfig.nodeSelector** labels must be a subset of the labels used in the **loadAffinity.nodeSelector** object.
- You cannot use the **matchExpressions** and **matchLabels** fields if you have configured both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects in the DPA.
- See the following example to configure both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects in the DPA.

```
...
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
      loadAffinity:
        - nodeSelector:
            matchLabels:
              label.io/location: 'US'
              label.io/gpu: 'no'
      podConfig:
        nodeSelector:
          label.io/gpu: 'no'
```

5.10.1.10. Configuring node agent load concurrency

You can control the maximum number of node agent operations that can run simultaneously on each node within your cluster.

You can configure it using one of the following fields of the Data Protection Application (DPA):

- **globalConfig**: Defines a default concurrency limit for the node agent across all nodes.
- **perNodeConfig**: Specifies different concurrency limits for specific nodes based on **nodeSelector** labels. This provides flexibility for environments where certain nodes might have different resource capacities or roles.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.

Procedure

1. If you want to use load concurrency for specific nodes, add labels to those nodes:

```
$ oc label node/<node_name> label.io/instance-type='large'
```

2. Configure the load concurrency fields for your DPA instance:

```
configuration:
  nodeAgent:
    enable: true
    uploaderType: kopia
    loadConcurrency:
      globalConfig: 1
      perNodeConfig:
        - nodeSelector:
            matchLabels:
              label.io/instance-type: large
          number: 3
```

where:

globalConfig

Specifies the global concurrent number. The default value is 1, which means there is no concurrency and only one load is allowed. The **globalConfig** value does not have a limit.

label.io/instance-type

Specifies the label for per-node concurrency.

number

Specifies the per-node concurrent number. You can specify many per-node concurrent numbers, for example, based on the instance type and size. The range of per-node concurrent number is the same as the global concurrent number. If the configuration file contains a per-node concurrent number and a global concurrent number, the per-node concurrent number takes precedence.

5.10.1.11. Configuring the node agent as a non-root and non-privileged user

To enhance the node agent security, you can configure the OADP Operator node agent daemonset to run as a non-root and non-privileged user by using the **spec.configuration.velero.disableFsBackup** setting in the **DataProtectionApplication** (DPA) custom resource (CR).

By setting the **spec.configuration.velero.disableFsBackup** setting to **true**, the node agent security context sets the root file system to read-only and sets the **privileged** flag to **false**.



NOTE

Setting **spec.configuration.velero.disableFsBackup** to **true** enhances the node agent security by removing the need for privileged containers and enforcing a read-only root file system.

However, it also disables File System Backup (FSB) with Kopia. If your workloads rely on FSB for backing up volumes that do not support native snapshots, then you should evaluate whether the **disableFsBackup** configuration fits your use case.

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **disableFsBackup** field in the DPA as shown in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: ts-dpa
  namespace: openshift-adp
spec:
  backupLocations:
  - velero:
      credential:
        key: cloud
        name: cloud-credentials
      default: true
      objectStorage:
        bucket: <bucket_name>
        prefix: velero
      provider: gcp
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
    velero:
      defaultPlugins:
      - csi
      - gcp
      - openshift
      disableFsBackup: true
```

where:

nodeAgent

Specifies to enable the node agent in the DPA.

disableFsBackup

Specifies to set the **disableFsBackup** field to **true**.

Verification

1. Verify that the node agent security context is set to run as non-root and the root file system is **readOnly** by running the following command:

```
$ oc get daemonset node-agent -o yaml
```

The example output is as following:

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  ...
  name: node-agent
  namespace: openshift-adp
  ...
spec:
  ...
  template:
    metadata:
      ...
    spec:
      containers:
        ...
        securityContext:
          allowPrivilegeEscalation: false
          capabilities:
            drop:
              - ALL
          privileged: false
          readOnlyRootFilesystem: true
          ...
      nodeSelector:
        kubernetes.io/os: linux
      os:
        name: linux
      restartPolicy: Always
      schedulerName: default-scheduler
      securityContext:
        runAsNonRoot: true
        seccompProfile:
          type: RuntimeDefault
      serviceAccount: velero
      serviceAccountName: velero
      ....
```

where:

allowPrivilegeEscalation

Specifies that the **allowPrivilegeEscalation** field is false.

privileged

Specifies that the **privileged** field is false.

readOnlyRootFilesystem

Specifies that the root file system is read-only.

runAsNonRoot

Specifies that the node agent is run as a non-root user.

5.10.1.12. Configuring repository maintenance

OADP repository maintenance is a background job, you can configure it independently of the node agent pods. This means that you can schedule the repository maintenance pod on a node where the node agent is or is not running.

You can use the repository maintenance job affinity configurations in the **DataProtectionApplication** (DPA) custom resource (CR) only if you use Kopia as the backup repository.

You have the option to configure the load affinity at the global level affecting all repositories. Or you can configure the load affinity for each repository. You can also use a combination of global and per-repository configuration.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the **loadAffinity** object in the DPA spec by using either one or both of the following methods:
 - Global configuration: Configure load affinity for all repositories as shown in the following example:

```
...
spec:
  configuration:
    repositoryMaintenance:
      global:
        podResources:
          cpuRequest: "100m"
          cpuLimit: "200m"
          memoryRequest: "100Mi"
          memoryLimit: "200Mi"
        loadAffinity:
          - nodeSelector:
              matchLabels:
                label.io/gpu: 'no'
            matchExpressions:
              - key: label.io/location
                operator: In
                values:
                  - US
                  - EU
```

where:

repositoryMaintenance

Specifies the **repositoryMaintenance** object as shown in the example.

global

Specifies the **global** object to configure load affinity for all repositories.

- Per-repository configuration: Configure load affinity per repository as shown in the following example:

```
...
spec:
  configuration:
    repositoryMaintenance:
      myrepositoryname:
        loadAffinity:
          - nodeSelector:
              matchLabels:
                label.io/cpu: 'yes'
```

where:

myrepositoryname

Specifies the **repositoryMaintenance** object for each repository.

5.10.13. Configuring Velero load affinity

With each OADP deployment, there is one Velero pod and its main purpose is to schedule Velero workloads. To schedule the Velero pod, you can use the **velero.podConfig.nodeSelector** and the **velero.loadAffinity** objects in the **DataProtectionApplication** (DPA) custom resource (CR) spec.

Use the **podConfig.nodeSelector** object to assign the Velero pod to specific nodes. You can also configure the **velero.loadAffinity** object for pod-level affinity and anti-affinity.

The OpenShift scheduler applies the rules and performs the scheduling of the Velero pod deployment.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the **velero.podConfig.nodeSelector** and the **velero.loadAffinity** objects in the DPA spec as shown in the following examples:
 - **velero.podConfig.nodeSelector** object configuration:

```
...
spec:
  configuration:
    velero:
```

```

    podConfig:
      nodeSelector:
        some-label.io/custom-node-role: backup-core

```

- **velero.loadAffinity** object configuration:

```

...
spec:
  configuration:
    velero:
      loadAffinity:
        - nodeSelector:
            matchLabels:
              label.io/gpu: 'no'
            matchExpressions:
              - key: label.io/location
                operator: In
            values:
              - US
              - EU

```

5.10.1.14. Overriding the imagePullPolicy setting in the DPA

In OADP 1.4.0 or earlier, the Operator sets the **imagePullPolicy** field of the Velero and node agent pods to **Always** for all images.

In OADP 1.4.1 or later, the Operator first checks if each image has the **sha256** or **sha512** digest and sets the **imagePullPolicy** field accordingly:

- If the image has the digest, the Operator sets **imagePullPolicy** to **IfNotPresent**.
- If the image does not have the digest, the Operator sets **imagePullPolicy** to **Always**.

You can also override the **imagePullPolicy** field by using the **spec.imagePullPolicy** field in the Data Protection Application (DPA).

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **spec.imagePullPolicy** field in the DPA as shown in the following example:

Example Data Protection Application

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: test-dpa
  namespace: openshift-adp
spec:
  backupLocations:
    - name: default

```



```

velero:
  config:
    insecureSkipTLSVerify: "true"
    profile: "default"
    region: <bucket_region>
    s3ForcePathStyle: "true"
    s3Url: <bucket_url>
  credential:
    key: cloud
    name: cloud-credentials
  default: true
  objectStorage:
    bucket: <bucket_name>
    prefix: velero
  provider: aws
configuration:
  nodeAgent:
    enable: true
    uploaderType: kopia
  velero:
    defaultPlugins:
      - openshift
      - aws
      - kubevirt
      - csi
imagePullPolicy: Never

```

where:

imagePullPolicy

Specifies the value for **imagePullPolicy**. In this example, the **imagePullPolicy** field is set to **Never**.

5.10.14.1. Enabling CSI in the DataProtectionApplication CR

You enable the Container Storage Interface (CSI) in the **DataProtectionApplication** custom resource (CR) in order to back up persistent volumes with CSI snapshots.

Prerequisites

- The cloud provider must support CSI snapshots.

Procedure

- Edit the **DataProtectionApplication** CR, as in the following example:

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - csi

```

-

where:

csi

Specifies the **csi** default plugin.

5.10.14.2. Disabling the node agent in DataProtectionApplication

If you are not using **Restic**, **Kopia**, or **DataMover** for your backups, you can disable the **nodeAgent** field in the **DataProtectionApplication** custom resource (CR). Before you disable **nodeAgent**, ensure the OADP Operator is idle and not running any backups.

Procedure

1. To disable the **nodeAgent**, set the **enable** flag to **false**. See the following example:

Example DataProtectionApplication CR

```
# ...
configuration:
  nodeAgent:
    enable: false
  uploaderType: kopia
# ...
```

where:

enable

Enables the node agent.

2. To enable the **nodeAgent**, set the **enable** flag to **true**. See the following example:

Example DataProtectionApplication CR

```
# ...
configuration:
  nodeAgent:
    enable: true
  uploaderType: kopia
# ...
```

where:

enable

Enables the node agent.

You can set up a job to enable and disable the **nodeAgent** field in the **DataProtectionApplication** CR. For more information, see "Running tasks in pods using jobs".

Additional resources

- [Installing the Data Protection Application with the **kubevirt** and **openshift** plugins](#)

- [Running tasks in pods using jobs](#).
- [Configuring the OpenShift API for Data Protection \(OADP\) with multiple backup storage locations](#)

5.11. CONFIGURING OADP WITH MCG

5.11.1. Configuring the OpenShift API for Data Protection with Multicloud Object Gateway

Multicloud Object Gateway (MCG) is a component of OpenShift Data Foundation, and you can configure it as a backup location in the **DataProtectionApplication** custom resource (CR). You can install the OpenShift API for Data Protection (OADP) with MCG by installing the OADP Operator. The Operator installs [Velero 1.16](#).



NOTE

Starting from OADP 1.0.4, all OADP 1.0.z versions can only be used as a dependency of the Migration Toolkit for Containers Operator and are not available as a standalone Operator.

You can create a **Secret** CR for the backup location and install the Data Protection Application. For more details, see [Installing the OADP Operator](#).

To install the OADP Operator in a restricted network environment, you must first disable the default software catalog sources and mirror the Operator catalog. For details, see [Using Operator Lifecycle Manager in disconnected environments](#).

5.11.1.1. Retrieving Multicloud Object Gateway credentials

You must retrieve the Multicloud Object Gateway (MCG) bucket credentials to create a **Secret** custom resource (CR) for OpenShift API for Data Protection (OADP).



NOTE

Although the MCG Operator is [deprecated](#), the MCG plugin is still available for OpenShift Data Foundation. To download the plugin, browse to [Download Red Hat OpenShift Data Foundation](#) and download the appropriate MCG plugin for your operating system.

Prerequisites

- You must deploy OpenShift Data Foundation by using the appropriate [Red Hat OpenShift Data Foundation deployment guide](#).

Procedure

1. Create an MCG bucket. For more information, see [Managing hybrid and multicloud resources](#).
2. Obtain the S3 endpoint, **AWS_ACCESS_KEY_ID**, **AWS_SECRET_ACCESS_KEY**, and the bucket name by running the **oc describe** command on the bucket resource.
3. Create a **credentials-velero** file:

```
$ cat << EOF > ./credentials-velero
[default]
aws_access_key_id=<AWS_ACCESS_KEY_ID>
aws_secret_access_key=<AWS_SECRET_ACCESS_KEY>
EOF
```

You can use the **credentials-velero** file to create a **Secret** object when you install the Data Protection Application.

5.11.1.2. About backup and snapshot locations and their secrets

Review backup location, snapshot location, and secret configuration requirements for the **DataProtectionApplication** custom resource (CR). This helps you understand storage options and credential management for data protection operations.

5.11.1.2.1. Backup locations

You can specify one of the following AWS S3-compatible object storage solutions as a backup location:

- Multicloud Object Gateway (MCG)
- Red Hat Container Storage
- Ceph RADOS Gateway; also known as Ceph Object Gateway
- Red Hat OpenShift Data Foundation
- MinIO

Velero backs up OpenShift Container Platform resources, Kubernetes objects, and internal images as an archive file on object storage.

5.11.1.2.2. Snapshot locations

If you use your cloud provider's native snapshot API to back up persistent volumes, you must specify the cloud provider as the snapshot location.

If you use Container Storage Interface (CSI) snapshots, you do not need to specify a snapshot location because you will create a **VolumeSnapshotClass** CR to register the CSI driver.

If you use File System Backup (FSB), you do not need to specify a snapshot location because FSB backs up the file system on object storage.

5.11.1.2.3. Secrets

If the backup and snapshot locations use the same credentials or if you do not require a snapshot location, you create a default **Secret**.

If the backup and snapshot locations use different credentials, you create two secret objects:

- Custom **Secret** for the backup location, which you specify in the **DataProtectionApplication** CR.
- Default **Secret** for the snapshot location, which is not referenced in the **DataProtectionApplication** CR.



IMPORTANT

The Data Protection Application requires a default **Secret**. Otherwise, the installation will fail.

If you do not want to specify backup or snapshot locations during the installation, you can create a default **Secret** with an empty **credentials-velero** file.

5.11.1.2.4. Creating a default Secret

You create a default **Secret** if your backup and snapshot locations use the same credentials or if you do not require a snapshot location.

The default name of the **Secret** is **cloud-credentials**.



NOTE

The **DataProtectionApplication** custom resource (CR) requires a default **Secret**. Otherwise, the installation will fail. If the name of the backup location **Secret** is not specified, the default name is used.

If you do not want to use the backup location credentials during the installation, you can create a **Secret** with the default name by using an empty **credentials-velero** file.

Prerequisites

- Your object storage and cloud storage, if any, must use the same credentials.
- You must configure object storage for Velero.

Procedure

1. Create a **credentials-velero** file for the backup storage location in the appropriate format for your cloud provider.
See the following example:

```
[default]
aws_access_key_id=<AWS_ACCESS_KEY_ID>
aws_secret_access_key=<AWS_SECRET_ACCESS_KEY>
```

2. Create a **Secret** custom resource (CR) with the default name:

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file cloud=credentials-velero
```

The **Secret** is referenced in the **spec.backupLocations.credential** block of the **DataProtectionApplication** CR when you install the Data Protection Application.

5.11.1.2.5. Creating secrets for different credentials

If your backup and snapshot locations use different credentials, you must create two **Secret** objects:

- Backup location **Secret** with a custom name. The custom name is specified in the **spec.backupLocations** block of the **DataProtectionApplication** custom resource (CR).

- Snapshot location **Secret** with the default name, **cloud-credentials**. This **Secret** is not specified in the **DataProtectionApplication** CR.

Procedure

1. Create a **credentials-velero** file for the snapshot location in the appropriate format for your cloud provider.
2. Create a **Secret** for the snapshot location with the default name:

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file cloud=credentials-velero
```

3. Create a **credentials-velero** file for the backup location in the appropriate format for your object storage.
4. Create a **Secret** for the backup location with a custom name:

```
$ oc create secret generic <custom_secret> -n openshift-adp --from-file cloud=credentials-velero
```

5. Add the **Secret** with the custom name to the **DataProtectionApplication** CR, as in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  ...
  backupLocations:
    - velero:
        config:
          profile: "default"
          region: <region_name> 1
          s3Url: <url>
          insecureSkipTLSVerify: "true"
          s3ForcePathStyle: "true"
        provider: aws
        default: true
        credential:
          key: cloud
          name: <custom_secret> 2
        objectStorage:
          bucket: <bucket_name>
          prefix: <prefix>
```

- 1 Specify the region, following the naming convention of the documentation of your object storage server.
- 2 Backup location **Secret** with custom name.

5.11.1.3. Configuring the Data Protection Application

You can configure the Data Protection Application by setting Velero resource allocations or enabling self-signed CA certificates.

5.11.1.3.1. Setting Velero CPU and memory resource allocations

You set the CPU and memory resource allocations for the **Velero** pod by editing the **DataProtectionApplication** custom resource (CR) manifest.

Prerequisites

- You must have the OpenShift API for Data Protection (OADP) Operator installed.

Procedure

- Edit the values in the **spec.configuration.velero.podConfig.ResourceAllocations** block of the **DataProtectionApplication** CR manifest, as in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  # ...
  configuration:
    velero:
      podConfig:
        nodeSelector: <node_selector>
        resourceAllocations:
          limits:
            cpu: "1"
            memory: 1024Mi
          requests:
            cpu: 200m
            memory: 256Mi
```

where:

nodeSelector

Specifies the node selector to be supplied to Velero podSpec.

resourceAllocations

Specifies the resource allocations listed for average usage.



NOTE

Kopia is an option in OADP 1.3 and later releases. You can use Kopia for file system backups, and Kopia is your only option for Data Mover cases with the built-in Data Mover.

Kopia is more resource intensive than Restic, and you might need to adjust the CPU and memory requirements accordingly.

Use the **nodeSelector** field to select which nodes can run the node agent. The **nodeSelector** field is the simplest recommended form of node selection constraint. Any label specified must match the labels on each node.

For more details, see [Configuring node agents and node labels](#).

5.11.1.3.2. Enabling self-signed CA certificates

You must enable a self-signed CA certificate for object storage by editing the **DataProtectionApplication** custom resource (CR) manifest to prevent a **certificate signed by unknown authority** error.

Prerequisites

- You must have the OpenShift API for Data Protection (OADP) Operator installed.

Procedure

- Edit the **spec.backupLocations.velero.objectStorage.caCert** parameter and **spec.backupLocations.velero.config** parameters of the **DataProtectionApplication** CR manifest:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  # ...
  backupLocations:
    - name: default
      velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket>
          prefix: <prefix>
          caCert: <base64_encoded_cert_string>
        config:
          insecureSkipTLSVerify: "false"
  # ...
```

where:

caCert

Specifies the Base64-encoded CA certificate string.

insecureSkipTLSVerify

Specifies the **insecureSkipTLSVerify** configuration. The configuration can be set to either **"true"** or **"false"**. If set to **"true"**, SSL/TLS security is disabled. If set to **"false"**, SSL/TLS security is enabled.

5.11.1.3.3. Using CA certificates with the velero command aliased for Velero deployment

You might want to use the Velero CLI without installing it locally on your system by creating an alias for it.

Prerequisites

- You must be logged in to the OpenShift Container Platform cluster as a user with the **cluster-admin** role.
- You must have the OpenShift CLI (**oc**) installed.

Procedure

1. To use an aliased Velero command, run the following command:

```
$ alias velero='oc -n openshift-adp exec deployment/velero -c velero -it -- ./velero'
```

2. Check that the alias is working by running the following command:

```
$ velero version
```

```
Client:
Version: v1.12.1-OADP
Git commit: -
Server:
Version: v1.12.1-OADP
```

3. To use a CA certificate with this command, you can add a certificate to the Velero deployment by running the following commands:

```
$ CA_CERT=$(oc -n openshift-adp get dataprotectionapplications.oadp.openshift.io <dpa-name> -o jsonpath='{.spec.backupLocations[0].velero.objectStorage.caCert}')
```

```
$ [[ -n $CA_CERT ]] && echo "$CA_CERT" | base64 -d | oc exec -n openshift-adp -i
deploy/velero -c velero -- bash -c "cat > /tmp/your-cacert.txt" || echo "DPA BSL has no
caCert"
```

```
$ velero describe backup <backup_name> --details --cacert /tmp/<your_cacert>.txt
```

4. To fetch the backup logs, run the following command:

```
$ velero backup logs <backup_name> --cacert /tmp/<your_cacert.txt>
```

You can use these logs to view failures and warnings for the resources that you cannot back up.

5. If the Velero pod restarts, the **/tmp/your-cacert.txt** file disappears, and you must re-create the **/tmp/your-cacert.txt** file by re-running the commands from the previous step.
6. You can check if the **/tmp/your-cacert.txt** file still exists, in the file location where you stored it, by running the following command:

```
$ oc exec -n openshift-adp -i deploy/velero -c velero -- bash -c "ls /tmp/your-cacert.txt"
/tmp/your-cacert.txt
```

In a future release of OpenShift API for Data Protection (OADP), we plan to mount the certificate to the Velero pod so that this step is not required.

5.11.1.4. Installing the Data Protection Application

You install the Data Protection Application (DPA) by creating an instance of the **DataProtectionApplication** API.

Prerequisites

- You must install the OADP Operator.
- You must configure object storage as a backup location.
- If you use snapshots to back up PVs, your cloud provider must support either a native snapshot API or Container Storage Interface (CSI) snapshots.
- If the backup and snapshot locations use the same credentials, you must create a **Secret** with the default name, **cloud-credentials**.
- If the backup and snapshot locations use different credentials, you must create two **Secrets**:
 - **Secret** with a custom name for the backup location. You add this **Secret** to the **DataProtectionApplication** CR.
 - **Secret** with another custom name for the snapshot location. You add this **Secret** to the **DataProtectionApplication** CR.



NOTE

If you do not want to specify backup or snapshot locations during the installation, you can create a default **Secret** with an empty **credentials-velero** file. If there is no default **Secret**, the installation will fail.

Procedure

1. Click **Ecosystem** → **Installed Operators** and select the OADP Operator.
2. Under **Provided APIs**, click **Create instance** in the **DataProtectionApplication** box.
3. Click **YAML View** and update the parameters of the **DataProtectionApplication** manifest:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  configuration:
    velero:
      defaultPlugins:
        - aws
        - openshift
      resourceTimeout: 10m
    nodeAgent:
```

```

enable: true
uploaderType: kopia
podConfig:
  nodeSelector: <node_selector>
backupLocations:
- velero:
  config:
    profile: "default"
    region: <region_name>
    s3Url: <url>
    insecureSkipTLSVerify: "true"
    s3ForcePathStyle: "true"
  provider: aws
  default: true
  credential:
    key: cloud
    name: cloud-credentials
  objectStorage:
    bucket: <bucket_name>
    prefix: <prefix>

```

where:

namespace

Specifies the default namespace for OADP which is **openshift-adp**. The namespace is a variable and is configurable.

aws

Specifies that an object store plugin corresponding to your storage locations is required. For all S3 providers, the required plugin is **aws**. For Azure and Google Cloud object stores, the **azure** or **gcp** plugin is required.

openshift

Specifies that the **openshift** plugin is mandatory.

resourceTimeout

Specifies how many minutes to wait for several Velero resources such as Velero CRD availability, volumeSnapshot deletion, and backup repository availability, before timeout occurs. The default is 10m.

nodeAgent

Specifies the administrative agent that routes the administrative requests to servers.

enable

Set this value to **true** if you want to enable **nodeAgent** and perform File System Backup.

uploaderType

Specifies the uploader type. Enter **kopia** or **restic** as your uploader. You cannot change the selection after the installation. For the Built-in DataMover you must use Kopia. The **nodeAgent** deploys a daemon set, which means that the **nodeAgent** pods run on each working node. You can configure File System Backup by adding **spec.defaultVolumesToFsBackup: true** to the **Backup** CR.

nodeSelector

Specifies the nodes on which Kopia or Restic are available. By default, Kopia or Restic run on all nodes.

region

Specifies the region, following the naming convention of the documentation of your object storage server.

s3Url

Specifies the URL of the S3 endpoint.

name

Specifies the name of the **Secret** object that you created. If you do not specify this value, the default name, **cloud-credentials**, is used. If you specify a custom name, the custom name is used for the backup location.

bucket

Specifies a bucket as the backup storage location. If the bucket is not a dedicated bucket for Velero backups, you must specify a prefix.

prefix

Specifies a prefix for Velero backups, for example, **velero**, if the bucket is used for multiple purposes.

4. Click **Create**.

Verification

1. Verify the installation by viewing the OpenShift API for Data Protection (OADP) resources by running the following command:

```
$ oc get all -n openshift-adp
```

NAME	READY	STATUS	RESTARTS	AGE
pod/oadp-operator-controller-manager-67d9494d47-6l8z8	2/2	Running	0	2m8s
pod/node-agent-9cq4q	1/1	Running	0	94s
pod/node-agent-m4lts	1/1	Running	0	94s
pod/node-agent-pv4kr	1/1	Running	0	95s
pod/velero-588db7f655-n842v	1/1	Running	0	95s

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP
service/oadp-operator-controller-manager-metrics-service	ClusterIP	172.30.70.140	
service/openshift-adp-velero-metrics-svc	ClusterIP	172.30.10.0	<none>

NAME	DESIRED	CURRENT	READY	UP-TO-DATE	AVAILABLE	AGE
daemonset.apps/node-agent	3	3	3	3	<none>	96s

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
deployment.apps/oadp-operator-controller-manager	1/1	1	1	2m9s
deployment.apps/velero	1/1	1	1	96s

NAME	DESIRED	CURRENT	READY	AGE
replicaset.apps/oadp-operator-controller-manager-67d9494d47	1	1	1	2m9s
replicaset.apps/velero-588db7f655	1	1	1	96s

2. Verify that the **DataProtectionApplication** (DPA) is reconciled by running the following command:

```
$ oc get dpa dpa-sample -n openshift-adp -o jsonpath='{.status}'
```

```
{"conditions":[{"lastTransitionTime":"2023-10-27T01:23:57Z","message":"Reconcile complete","reason":"Complete","status":"True","type":"Reconciled"}]}
```

3. Verify the **type** is set to **Reconciled**.
4. Verify the backup storage location and confirm that the **PHASE** is **Available** by running the following command:

```
$ oc get backupstoragelocations.velero.io -n openshift-adp
```

NAME	PHASE	LAST VALIDATED	AGE	DEFAULT
dpa-sample-1	Available	1s	3d16h	true

5.11.1.5. Configuring the DPA with client burst and QPS settings

The burst setting determines how many requests can be sent to the **velero** server before the limit is applied. After the burst limit is reached, the queries per second (QPS) setting determines how many additional requests can be sent per second.

You can set the burst and QPS values of the **velero** server by configuring the Data Protection Application (DPA) with the burst and QPS values. You can use the **dpa.configuration.velero.client-burst** and **dpa.configuration.velero.client-qps** fields of the DPA to set the burst and QPS values.

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **client-burst** and the **client-qps** fields in the DPA as shown in the following example:

Example Data Protection Application

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: test-dpa
  namespace: openshift-adp
spec:
  backupLocations:
    - name: default
  velero:
    config:
      insecureSkipTLSVerify: "true"
      profile: "default"
      region: <bucket_region>
      s3ForcePathStyle: "true"
```

```
s3Url: <bucket_url>
credential:
  key: cloud
  name: cloud-credentials
default: true
objectStorage:
  bucket: <bucket_name>
  prefix: velero
provider: aws
configuration:
  nodeAgent:
    enable: true
    uploaderType: restic
velero:
  client-burst: 500
  client-qps: 300
  defaultPlugins:
    - openshift
    - aws
    - kubevirt
```

where:

client-burst

Specifies the **client-burst** value. In this example, the **client-burst** field is set to 500.

client-qps

Specifies the **client-qps** value. In this example, the **client-qps** field is set to 300.

5.11.1.6. Configuring node agents and node labels

The Data Protection Application (DPA) uses the **nodeSelector** field to select which nodes can run the node agent. The **nodeSelector** field is the recommended form of node selection constraint.

Procedure

1. Run the node agent on any node that you choose by adding a custom label:

```
$ oc label node/<node_name> node-role.kubernetes.io/nodeAgent=""
```



NOTE

Any label specified must match the labels on each node.

2. Use the same custom label in the **DPA.spec.configuration.nodeAgent.podConfig.nodeSelector** field, which you used for labeling nodes:

```
configuration:
  nodeAgent:
    enable: true
  podConfig:
    nodeSelector:
      node-role.kubernetes.io/nodeAgent: ""
```

■

The following example is an anti-pattern of **nodeSelector** and does not work unless both labels, **node-role.kubernetes.io/infra: ""** and **node-role.kubernetes.io/worker: ""**, are on the node:

```
configuration:
  nodeAgent:
    enable: true
  podConfig:
    nodeSelector:
      node-role.kubernetes.io/infra: ""
      node-role.kubernetes.io/worker: ""
```

5.11.1.7. Configuring node agent load affinity

You can schedule the node agent pods on specific nodes by using the **spec.podConfig.nodeSelector** object of the **DataProtectionApplication** (DPA) custom resource (CR).

See the following example in which you can schedule the node agent pods on nodes with the label **label.io/role: cpu-1** and **other-label.io/other-role: cpu-2**.

```
...
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
    podConfig:
      nodeSelector:
        label.io/role: cpu-1
        other-label.io/other-role: cpu-2
  ...
```

You can add more restrictions on the node agent pods scheduling by using the **nodeagent.loadAffinity** object in the DPA spec.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the DPA spec **nodeagent.loadAffinity** object as shown in the following example. In the example, you ensure that the node agent pods are scheduled only on nodes with the label **label.io/role: cpu-1** and the label **label.io/hostname** matching with either **node1** or **node2**.

```
...
spec:
  configuration:
    nodeAgent:
      enable: true
```

```

loadAffinity:
  - nodeSelector:
      matchLabels:
        label.io/role: cpu-1
      matchExpressions:
        - key: label.io/hostname
          operator: In
          values:
            - node1
            - node2
            ...

```

where:

loadAffinity

Specifies the **loadAffinity** object by adding the **matchLabels** and **matchExpressions** objects.

matchExpressions

Specifies the **matchExpressions** object to add restrictions on the node agent pods scheduling.

5.11.1.8. Node agent load affinity guidelines

Use the following guidelines to configure the node agent **loadAffinity** object in the **DataProtectionApplication** (DPA) custom resource (CR).

- Use the **spec.nodeagent.podConfig.nodeSelector** object for simple node matching.
- Use the **loadAffinity.nodeSelector** object without the **podConfig.nodeSelector** object for more complex scenarios.
- You can use both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects, but the **loadAffinity** object must be equal or more restrictive as compared to the **podConfig** object. In this scenario, the **podConfig.nodeSelector** labels must be a subset of the labels used in the **loadAffinity.nodeSelector** object.
- You cannot use the **matchExpressions** and **matchLabels** fields if you have configured both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects in the DPA.
- See the following example to configure both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects in the DPA.

```

...
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
      loadAffinity:
        - nodeSelector:
            matchLabels:
              label.io/location: 'US'
              label.io/gpu: 'no'

```



```
podConfig:
  nodeSelector:
    label.io/gpu: 'no'
```

5.11.1.9. Configuring node agent load concurrency

You can control the maximum number of node agent operations that can run simultaneously on each node within your cluster.

You can configure it using one of the following fields of the Data Protection Application (DPA):

- **globalConfig**: Defines a default concurrency limit for the node agent across all nodes.
- **perNodeConfig**: Specifies different concurrency limits for specific nodes based on **nodeSelector** labels. This provides flexibility for environments where certain nodes might have different resource capacities or roles.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.

Procedure

1. If you want to use load concurrency for specific nodes, add labels to those nodes:

```
$ oc label node/<node_name> label.io/instance-type='large'
```

2. Configure the load concurrency fields for your DPA instance:

```
configuration:
  nodeAgent:
    enable: true
    uploaderType: kopia
    loadConcurrency:
      globalConfig: 1
      perNodeConfig:
        - nodeSelector:
            matchLabels:
              label.io/instance-type: large
          number: 3
```

where:

globalConfig

Specifies the global concurrent number. The default value is 1, which means there is no concurrency and only one load is allowed. The **globalConfig** value does not have a limit.

label.io/instance-type

Specifies the label for per-node concurrency.

number

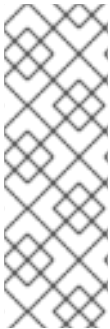
Specifies the per-node concurrent number. You can specify many per-node concurrent numbers, for example, based on the instance type and size. The range of per-node concurrent number is the same as the global concurrent number. If the configuration file

contains a per-node concurrent number and a global concurrent number, the per-node concurrent number takes precedence.

5.11.1.10. Configuring the node agent as a non-root and non-privileged user

To enhance the node agent security, you can configure the OADP Operator node agent daemonset to run as a non-root and non-privileged user by using the **spec.configuration.velero.disableFsBackup** setting in the **DataProtectionApplication** (DPA) custom resource (CR).

By setting the **spec.configuration.velero.disableFsBackup** setting to **true**, the node agent security context sets the root file system to read-only and sets the **privileged** flag to **false**.



NOTE

Setting **spec.configuration.velero.disableFsBackup** to **true** enhances the node agent security by removing the need for privileged containers and enforcing a read-only root file system.

However, it also disables File System Backup (FSB) with Kopia. If your workloads rely on FSB for backing up volumes that do not support native snapshots, then you should evaluate whether the **disableFsBackup** configuration fits your use case.

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **disableFsBackup** field in the DPA as shown in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: ts-dpa
  namespace: openshift-adp
spec:
  backupLocations:
  - velero:
      credential:
        key: cloud
        name: cloud-credentials
      default: true
      objectStorage:
        bucket: <bucket_name>
        prefix: velero
        provider: gcp
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
    velero:
      defaultPlugins:
      - csi
```

```
- gcp
- openshift
disableFsBackup: true
```

where:

nodeAgent

Specifies to enable the node agent in the DPA.

disableFsBackup

Specifies to set the **disableFsBackup** field to **true**.

Verification

1. Verify that the node agent security context is set to run as non-root and the root file system is **readOnly** by running the following command:

```
$ oc get daemonset node-agent -o yaml
```

The example output is as following:

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  ...
  name: node-agent
  namespace: openshift-adp
  ...
spec:
  ...
  template:
    metadata:
      ...
    spec:
      containers:
        ...
        securityContext:
          allowPrivilegeEscalation: false
          capabilities:
            drop:
              - ALL
          privileged: false
          readOnlyRootFilesystem: true
          ...
      nodeSelector:
        kubernetes.io/os: linux
      os:
        name: linux
      restartPolicy: Always
      schedulerName: default-scheduler
      securityContext:
        runAsNonRoot: true
      seccompProfile:
        type: RuntimeDefault
```

```

serviceAccount: velero
serviceAccountName: velero
....

```

where:

allowPrivilegeEscalation

Specifies that the **allowPrivilegeEscalation** field is false.

privileged

Specifies that the **privileged** field is false.

readOnlyRootFilesystem

Specifies that the root file system is read-only.

runAsNonRoot

Specifies that the node agent is run as a non-root user.

5.11.1.11. Configuring repository maintenance

OADP repository maintenance is a background job, you can configure it independently of the node agent pods. This means that you can schedule the repository maintenance pod on a node where the node agent is or is not running.

You can use the repository maintenance job affinity configurations in the **DataProtectionApplication** (DPA) custom resource (CR) only if you use Kopia as the backup repository.

You have the option to configure the load affinity at the global level affecting all repositories. Or you can configure the load affinity for each repository. You can also use a combination of global and per-repository configuration.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the **loadAffinity** object in the DPA spec by using either one or both of the following methods:
 - Global configuration: Configure load affinity for all repositories as shown in the following example:

```

...
spec:
  configuration:
    repositoryMaintenance:
      global:
        podResources:
          cpuRequest: "100m"
          cpuLimit: "200m"
          memoryRequest: "100Mi"

```

```

memoryLimit: "200Mi"
loadAffinity:
  - nodeSelector:
      matchLabels:
        label.io/gpu: 'no'
      matchExpressions:
        - key: label.io/location
          operator: In
          values:
            - US
            - EU

```

where:

repositoryMaintenance

Specifies the **repositoryMaintenance** object as shown in the example.

global

Specifies the **global** object to configure load affinity for all repositories.

- Per-repository configuration: Configure load affinity per repository as shown in the following example:

```

...
spec:
  configuration:
    repositoryMaintenance:
      myrepositoryname:
        loadAffinity:
          - nodeSelector:
              matchLabels:
                label.io/cpu: 'yes'

```

where:

myrepositoryname

Specifies the **repositoryMaintenance** object for each repository.

5.11.1.12. Configuring Velero load affinity

With each OADP deployment, there is one Velero pod and its main purpose is to schedule Velero workloads. To schedule the Velero pod, you can use the **velero.podConfig.nodeSelector** and the **velero.loadAffinity** objects in the **DataProtectionApplication** (DPA) custom resource (CR) spec.

Use the **podConfig.nodeSelector** object to assign the Velero pod to specific nodes. You can also configure the **velero.loadAffinity** object for pod-level affinity and anti-affinity.

The OpenShift scheduler applies the rules and performs the scheduling of the Velero pod deployment.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.

- You have configured the DPA CR.

Procedure

- Configure the **velero.podConfig.nodeSelector** and the **velero.loadAffinity** objects in the DPA spec as shown in the following examples:

- **velero.podConfig.nodeSelector** object configuration:

```
...
spec:
  configuration:
    velero:
      podConfig:
        nodeSelector:
          some-label.io/custom-node-role: backup-core
```

- **velero.loadAffinity** object configuration:

```
...
spec:
  configuration:
    velero:
      loadAffinity:
        - nodeSelector:
            matchLabels:
              label.io/gpu: 'no'
            matchExpressions:
              - key: label.io/location
                operator: In
                values:
                  - US
                  - EU
```

5.11.1.13. Overriding the imagePullPolicy setting in the DPA

In OADP 1.4.0 or earlier, the Operator sets the **imagePullPolicy** field of the Velero and node agent pods to **Always** for all images.

In OADP 1.4.1 or later, the Operator first checks if each image has the **sha256** or **sha512** digest and sets the **imagePullPolicy** field accordingly:

- If the image has the digest, the Operator sets **imagePullPolicy** to **IfNotPresent**.
- If the image does not have the digest, the Operator sets **imagePullPolicy** to **Always**.

You can also override the **imagePullPolicy** field by using the **spec.imagePullPolicy** field in the Data Protection Application (DPA).

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **spec.imagePullPolicy** field in the DPA as shown in the following example:

Example Data Protection Application

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: test-dpa
  namespace: openshift-adp
spec:
  backupLocations:
    - name: default
      velero:
        config:
          insecureSkipTLSVerify: "true"
          profile: "default"
          region: <bucket_region>
          s3ForcePathStyle: "true"
          s3Url: <bucket_url>
        credential:
          key: cloud
          name: cloud-credentials
        default: true
        objectStorage:
          bucket: <bucket_name>
          prefix: velero
        provider: aws
      configuration:
        nodeAgent:
          enable: true
          uploaderType: kopia
        velero:
          defaultPlugins:
            - openshift
            - aws
            - kubevirt
            - csi
  imagePullPolicy: Never
```

where:

imagePullPolicy

Specifies the value for **imagePullPolicy**. In this example, the **imagePullPolicy** field is set to **Never**.

5.11.1.3.1. Enabling CSI in the DataProtectionApplication CR

You enable the Container Storage Interface (CSI) in the **DataProtectionApplication** custom resource (CR) in order to back up persistent volumes with CSI snapshots.

Prerequisites

- The cloud provider must support CSI snapshots.

Procedure

- Edit the **DataProtectionApplication** CR, as in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - csi
```

where:

csi

Specifies the **csi** default plugin.

5.11.13.2. Disabling the node agent in DataProtectionApplication

If you are not using **Restic**, **Kopia**, or **DataMover** for your backups, you can disable the **nodeAgent** field in the **DataProtectionApplication** custom resource (CR). Before you disable **nodeAgent**, ensure the OADP Operator is idle and not running any backups.

Procedure

1. To disable the **nodeAgent**, set the **enable** flag to **false**. See the following example:

Example DataProtectionApplication CR

```
# ...
configuration:
  nodeAgent:
    enable: false
    uploaderType: kopia
# ...
```

where:

enable

Enables the node agent.

2. To enable the **nodeAgent**, set the **enable** flag to **true**. See the following example:

Example DataProtectionApplication CR

```
# ...
configuration:
  nodeAgent:
    enable: true
    uploaderType: kopia
# ...
```


where:

enable

Enables the node agent.

You can set up a job to enable and disable the **nodeAgent** field in the **DataProtectionApplication** CR. For more information, see "Running tasks in pods using jobs".

Additional resources

- [Performance tuning guide for Multicloud Object Gateway](#) .
- [Installing the Data Protection Application with the **kubevirt** and **openshift** plugins](#)
- [Running tasks in pods using jobs](#) .
- [Configuring the OpenShift API for Data Protection \(OADP\) with multiple backup storage locations](#)

5.12. CONFIGURING OADP WITH ODF

5.12.1. Configuring the OpenShift API for Data Protection with OpenShift Data Foundation

You install the OpenShift API for Data Protection (OADP) with OpenShift Data Foundation by installing the OADP Operator and configuring a backup location and a snapshot location. Then, you install the Data Protection Application.



NOTE

Starting from OADP 1.0.4, all OADP 1.0.z versions can only be used as a dependency of the Migration Toolkit for Containers Operator and are not available as a standalone Operator.

You can configure [Multicloud Object Gateway](#) or any AWS S3-compatible object storage as a backup location.

You can create a **Secret** CR for the backup location and install the Data Protection Application. For more details, see [Installing the OADP Operator](#).

To install the OADP Operator in a restricted network environment, you must first disable the default software catalog sources and mirror the Operator catalog. For details, see [Using Operator Lifecycle Manager in disconnected environments](#).

5.12.1.1. About backup and snapshot locations and their secrets

Review backup location, snapshot location, and secret configuration requirements for the **DataProtectionApplication** custom resource (CR). This helps you understand storage options and credential management for data protection operations.

5.12.1.1.1. Backup locations

You can specify one of the following AWS S3-compatible object storage solutions as a backup location:

- Multicloud Object Gateway (MCG)
- Red Hat Container Storage
- Ceph RADOS Gateway; also known as Ceph Object Gateway
- Red Hat OpenShift Data Foundation
- MinIO

Velero backs up OpenShift Container Platform resources, Kubernetes objects, and internal images as an archive file on object storage.

5.12.1.1.2. Snapshot locations

If you use your cloud provider's native snapshot API to back up persistent volumes, you must specify the cloud provider as the snapshot location.

If you use Container Storage Interface (CSI) snapshots, you do not need to specify a snapshot location because you will create a **VolumeSnapshotClass** CR to register the CSI driver.

If you use File System Backup (FSB), you do not need to specify a snapshot location because FSB backs up the file system on object storage.

5.12.1.1.3. Secrets

If the backup and snapshot locations use the same credentials or if you do not require a snapshot location, you create a default **Secret**.

If the backup and snapshot locations use different credentials, you create two secret objects:

- Custom **Secret** for the backup location, which you specify in the **DataProtectionApplication** CR.
- Default **Secret** for the snapshot location, which is not referenced in the **DataProtectionApplication** CR.



IMPORTANT

The Data Protection Application requires a default **Secret**. Otherwise, the installation will fail.

If you do not want to specify backup or snapshot locations during the installation, you can create a default **Secret** with an empty **credentials-velero** file.

Additional resources

- [Creating an Object Bucket Claim using the OpenShift Web Console](#) .

5.12.1.1.4. Creating a default Secret

You create a default **Secret** if your backup and snapshot locations use the same credentials or if you do not require a snapshot location.

The default name of the **Secret** is **cloud-credentials**, unless your backup storage provider has a default plugin, such as **aws**, **azure**, or **gcp**. In that case, the default name is specified in the provider-specific OADP installation procedure.



NOTE

The **DataProtectionApplication** custom resource (CR) requires a default **Secret**. Otherwise, the installation will fail. If the name of the backup location **Secret** is not specified, the default name is used.

If you do not want to use the backup location credentials during the installation, you can create a **Secret** with the default name by using an empty **credentials-velero** file.

Prerequisites

- Your object storage and cloud storage, if any, must use the same credentials.
- You must configure object storage for Velero.

Procedure

1. Create a **credentials-velero** file for the backup storage location in the appropriate format for your cloud provider.
See the following example:

```
[default]
aws_access_key_id=<AWS_ACCESS_KEY_ID>
aws_secret_access_key=<AWS_SECRET_ACCESS_KEY>
```

2. Create a **Secret** custom resource (CR) with the default name:

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file cloud=credentials-velero
```

The **Secret** is referenced in the **spec.backupLocations.credential** block of the **DataProtectionApplication** CR when you install the Data Protection Application.

5.12.1.1.5. Creating secrets for different credentials

If your backup and snapshot locations use different credentials, you must create two **Secret** objects:

- Backup location **Secret** with a custom name. The custom name is specified in the **spec.backupLocations** block of the **DataProtectionApplication** custom resource (CR).
- Snapshot location **Secret** with the default name, **cloud-credentials**. This **Secret** is not specified in the **DataProtectionApplication** CR.

Procedure

1. Create a **credentials-velero** file for the snapshot location in the appropriate format for your cloud provider.
2. Create a **Secret** for the snapshot location with the default name:

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file cloud=credentials-velero
```

3. Create a **credentials-velero** file for the backup location in the appropriate format for your object storage.
4. Create a **Secret** for the backup location with a custom name:

```
$ oc create secret generic <custom_secret> -n openshift-adp --from-file cloud=credentials-velero
```

5. Add the **Secret** with the custom name to the **DataProtectionApplication** CR, as in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  ...
  backupLocations:
    - velero:
        provider: <provider>
        default: true
        credential:
          key: cloud
          name: <custom_secret> 1
        objectStorage:
          bucket: <bucket_name>
          prefix: <prefix>
```

- 1 Backup location **Secret** with custom name.

5.12.1.2. Configuring the Data Protection Application

You can configure the Data Protection Application by setting Velero resource allocations or enabling self-signed CA certificates.

5.12.1.2.1. Setting Velero CPU and memory resource allocations

You set the CPU and memory resource allocations for the **Velero** pod by editing the **DataProtectionApplication** custom resource (CR) manifest.

Prerequisites

- You must have the OpenShift API for Data Protection (OADP) Operator installed.

Procedure

- Edit the values in the **spec.configuration.velero.podConfig.ResourceAllocations** block of the **DataProtectionApplication** CR manifest, as in the following example:

–

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  # ...
  configuration:
    velero:
      podConfig:
        nodeSelector: <node_selector>
        resourceAllocations:
          limits:
            cpu: "1"
            memory: 1024Mi
          requests:
            cpu: 200m
            memory: 256Mi

```

where:

nodeSelector

Specifies the node selector to be supplied to Velero podSpec.

resourceAllocations

Specifies the resource allocations listed for average usage.



NOTE

Kopia is an option in OADP 1.3 and later releases. You can use Kopia for file system backups, and Kopia is your only option for Data Mover cases with the built-in Data Mover.

Kopia is more resource intensive than Restic, and you might need to adjust the CPU and memory requirements accordingly.

Use the **nodeSelector** field to select which nodes can run the node agent. The **nodeSelector** field is the simplest recommended form of node selection constraint. Any label specified must match the labels on each node.

For more details, see [Configuring node agents and node labels](#).

5.12.1.2.1.1. Adjusting Ceph CPU and memory requirements based on collected data

The following recommendations are based on observations of performance made in the scale and performance lab. The changes are specifically related to Red Hat OpenShift Data Foundation (ODF). If working with ODF, consult the appropriate tuning guides for official recommendations.

5.12.1.2.1.1.1. CPU and memory requirement for configurations

Backup and restore operations require large amounts of CephFS **PersistentVolumes** (PVs). To avoid Ceph MDS pods restarting with an **out-of-memory** (OOM) error, the following configuration is suggested:

Configuration types	Request	Max limit
CPU	Request changed to 3	Max limit to 3
Memory	Request changed to 8 Gi	Max limit to 128 Gi

5.12.1.2.2. Enabling self-signed CA certificates

You must enable a self-signed CA certificate for object storage by editing the **DataProtectionApplication** custom resource (CR) manifest to prevent a **certificate signed by unknown authority** error.

Prerequisites

- You must have the OpenShift API for Data Protection (OADP) Operator installed.

Procedure

- Edit the **spec.backupLocations.velero.objectStorage.caCert** parameter and **spec.backupLocations.velero.config** parameters of the **DataProtectionApplication** CR manifest:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
  # ...
  backupLocations:
    - name: default
      velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket>
          prefix: <prefix>
          caCert: <base64_encoded_cert_string>
        config:
          insecureSkipTLSVerify: "false"
  # ...
```

where:

caCert

Specifies the Base64-encoded CA certificate string.

insecureSkipTLSVerify

Specifies the **insecureSkipTLSVerify** configuration. The configuration can be set to either **"true"** or **"false"**. If set to **"true"**, SSL/TLS security is disabled. If set to **"false"**, SSL/TLS security is enabled.

5.12.1.2.3. Using CA certificates with the velero command aliased for Velero deployment

You might want to use the Velero CLI without installing it locally on your system by creating an alias for it.

Prerequisites

- You must be logged in to the OpenShift Container Platform cluster as a user with the **cluster-admin** role.
- You must have the OpenShift CLI (**oc**) installed.

Procedure

1. To use an aliased Velero command, run the following command:

```
$ alias velero='oc -n openshift-adp exec deployment/velero -c velero -it -- ./velero'
```

2. Check that the alias is working by running the following command:

```
$ velero version
```

```
Client:
Version: v1.12.1-OADP
Git commit: -
Server:
Version: v1.12.1-OADP
```

3. To use a CA certificate with this command, you can add a certificate to the Velero deployment by running the following commands:

```
$ CA_CERT=$(oc -n openshift-adp get dataprotectionapplications.oadp.openshift.io <dpa-name> -o jsonpath='{.spec.backupLocations[0].velero.objectStorage.caCert}')
```

```
$ [[ -n $CA_CERT ]] && echo "$CA_CERT" | base64 -d | oc exec -n openshift-adp -i deploy/velero -c velero -- bash -c "cat > /tmp/your-cacert.txt" || echo "DPA BSL has no caCert"
```

```
$ velero describe backup <backup_name> --details --cacert /tmp/<your_cacert>.txt
```

4. To fetch the backup logs, run the following command:

```
$ velero backup logs <backup_name> --cacert /tmp/<your_cacert.txt>
```

You can use these logs to view failures and warnings for the resources that you cannot back up.

5. If the Velero pod restarts, the **/tmp/your-cacert.txt** file disappears, and you must re-create the **/tmp/your-cacert.txt** file by re-running the commands from the previous step.
6. You can check if the **/tmp/your-cacert.txt** file still exists, in the file location where you stored it, by running the following command:

```
$ oc exec -n openshift-adp -i deploy/velero -c velero -- bash -c "ls /tmp/your-cacert.txt" /tmp/your-cacert.txt
```

In a future release of OpenShift API for Data Protection (OADP), we plan to mount the certificate to the Velero pod so that this step is not required.

5.12.1.3. Installing the Data Protection Application

You install the Data Protection Application (DPA) by creating an instance of the **DataProtectionApplication** API.

Prerequisites

- You must install the OADP Operator.
- You must configure object storage as a backup location.
- If you use snapshots to back up PVs, your cloud provider must support either a native snapshot API or Container Storage Interface (CSI) snapshots.
- If the backup and snapshot locations use the same credentials, you must create a **Secret** with the default name, **cloud-credentials**.
- If the backup and snapshot locations use different credentials, you must create two **Secrets**:
 - **Secret** with a custom name for the backup location. You add this **Secret** to the **DataProtectionApplication** CR.
 - **Secret** with another custom name for the snapshot location. You add this **Secret** to the **DataProtectionApplication** CR.



NOTE

If you do not want to specify backup or snapshot locations during the installation, you can create a default **Secret** with an empty **credentials-velero** file. If there is no default **Secret**, the installation will fail.

Procedure

1. Click **Ecosystem** → **Installed Operators** and select the OADP Operator.
2. Under **Provided APIs**, click **Create instance** in the **DataProtectionApplication** box.
3. Click **YAML View** and update the parameters of the **DataProtectionApplication** manifest:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  configuration:
    velero:
      defaultPlugins:
        - aws
        - kubevirt
        - csi
        - openshift
```



```

resourceTimeout: 10m
nodeAgent:
  enable: true
  uploaderType: kopia
  podConfig:
    nodeSelector: <node_selector>
backupLocations:
  - velero:
    provider: gcp
    default: true
    credential:
      key: cloud
      name: <default_secret>
    objectStorage:
      bucket: <bucket_name>
      prefix: <prefix>

```

where:

namespace

Specifies the default namespace for OADP which is **openshift-adp**. The namespace is a variable and is configurable.

aws

Specifies that an object store plugin corresponding to your storage locations is required. For all S3 providers, the required plugin is **aws**. For Azure and Google Cloud object stores, the **azure** or **gcp** plugin is required.

kubevirt

Optional: The **kubevirt** plugin is used with OpenShift Virtualization.

csi

Specifies the **csi** default plugin if you use CSI snapshots to back up PVs. The **csi** plugin uses the [Velero CSI beta snapshot APIs](#). You do not need to configure a snapshot location.

openshift

Specifies that the **openshift** plugin is mandatory.

resourceTimeout

Specifies how many minutes to wait for several Velero resources such as Velero CRD availability, volumeSnapshot deletion, and backup repository availability, before timeout occurs. The default is 10m.

nodeAgent

Specifies the administrative agent that routes the administrative requests to servers.

enable

Set this value to **true** if you want to enable **nodeAgent** and perform File System Backup.

uploaderType

Specifies the uploader type. Enter **kopia** or **restic** as your uploader. You cannot change the selection after the installation. For the Built-in DataMover you must use Kopia. The **nodeAgent** deploys a daemon set, which means that the **nodeAgent** pods run on each working node. You can configure File System Backup by adding **spec.defaultVolumesToFsBackup: true** to the **Backup** CR.

nodeSelector

Specifies the nodes on which Kopia or Restic are available. By default, Kopia or Restic run on all nodes.

provider

Specifies the backup provider.

name

Specifies the correct default name for the **Secret**, for example, **cloud-credentials-gcp**, if you use a default plugin for the backup provider. If specifying a custom name, then the custom name is used for the backup location. If you do not specify a **Secret** name, the default name is used.

bucket

Specifies a bucket as the backup storage location. If the bucket is not a dedicated bucket for Velero backups, you must specify a prefix.

prefix

Specifies a prefix for Velero backups, for example, **velero**, if the bucket is used for multiple purposes.

- 4. Click **Create**.

Verification

- 1. Verify the installation by viewing the OpenShift API for Data Protection (OADP) resources by running the following command:

```
$ oc get all -n openshift-adp
```

NAME	READY	STATUS	RESTARTS	AGE
pod/oadp-operator-controller-manager-67d9494d47-6l8z8	2/2	Running	0	2m8s
pod/node-agent-9cq4q	1/1	Running	0	94s
pod/node-agent-m4lts	1/1	Running	0	94s
pod/node-agent-pv4kr	1/1	Running	0	95s
pod/velero-588db7f655-n842v	1/1	Running	0	95s

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP
service/oadp-operator-controller-manager-metrics-service	ClusterIP	172.30.70.140	
<none>	8443/TCP	2m8s	
service/openshift-adp-velero-metrics-svc	ClusterIP	172.30.10.0	<none>
8085/TCP	8h		

NAME	DESIRED	CURRENT	READY	UP-TO-DATE	AVAILABLE	NODE
selector						
daemonset.apps/node-agent	3	3	3	3	<none>	96s

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
deployment.apps/oadp-operator-controller-manager	1/1	1	1	2m9s
deployment.apps/velero	1/1	1	1	96s

NAME	DESIRED	CURRENT	READY	AGE
replicaset.apps/oadp-operator-controller-manager-67d9494d47	1	1	1	2m9s
replicaset.apps/velero-588db7f655	1	1	1	96s

2. Verify that the **DataProtectionApplication** (DPA) is reconciled by running the following command:

```
$ oc get dpa dpa-sample -n openshift-adp -o jsonpath='{.status}'
```

```
{"conditions":[{"lastTransitionTime":"2023-10-27T01:23:57Z","message":"Reconcile complete","reason":"Complete","status":"True","type":"Reconciled"]}]}
```

3. Verify the **type** is set to **Reconciled**.
4. Verify the backup storage location and confirm that the **PHASE** is **Available** by running the following command:

```
$ oc get backupstoragelocations.velero.io -n openshift-adp
```

NAME	PHASE	LAST VALIDATED	AGE	DEFAULT
dpa-sample-1	Available	1s	3d16h	true

5.12.1.4. Configuring the DPA with client burst and QPS settings

The burst setting determines how many requests can be sent to the **velero** server before the limit is applied. After the burst limit is reached, the queries per second (QPS) setting determines how many additional requests can be sent per second.

You can set the burst and QPS values of the **velero** server by configuring the Data Protection Application (DPA) with the burst and QPS values. You can use the **dpa.configuration.velero.client-burst** and **dpa.configuration.velero.client-qps** fields of the DPA to set the burst and QPS values.

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **client-burst** and the **client-qps** fields in the DPA as shown in the following example:

Example Data Protection Application

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: test-dpa
  namespace: openshift-adp
spec:
  backupLocations:
    - name: default
  velero:
    config:
      insecureSkipTLSVerify: "true"
      profile: "default"
      region: <bucket_region>
      s3ForcePathStyle: "true"
```

```
s3Url: <bucket_url>
credential:
  key: cloud
  name: cloud-credentials
default: true
objectStorage:
  bucket: <bucket_name>
  prefix: velero
provider: aws
configuration:
  nodeAgent:
    enable: true
    uploaderType: restic
velero:
  client-burst: 500
  client-qps: 300
  defaultPlugins:
    - openshift
    - aws
    - kubevirt
```

where:

client-burst

Specifies the **client-burst** value. In this example, the **client-burst** field is set to 500.

client-qps

Specifies the **client-qps** value. In this example, the **client-qps** field is set to 300.

5.12.1.5. Configuring node agents and node labels

The Data Protection Application (DPA) uses the **nodeSelector** field to select which nodes can run the node agent. The **nodeSelector** field is the recommended form of node selection constraint.

Procedure

1. Run the node agent on any node that you choose by adding a custom label:

```
$ oc label node/<node_name> node-role.kubernetes.io/nodeAgent=""
```



NOTE

Any label specified must match the labels on each node.

2. Use the same custom label in the **DPA.spec.configuration.nodeAgent.podConfig.nodeSelector** field, which you used for labeling nodes:

```
configuration:
  nodeAgent:
    enable: true
  podConfig:
    nodeSelector:
      node-role.kubernetes.io/nodeAgent: ""
```

■

The following example is an anti-pattern of **nodeSelector** and does not work unless both labels, **node-role.kubernetes.io/infra: ""** and **node-role.kubernetes.io/worker: ""**, are on the node:

```
configuration:
  nodeAgent:
    enable: true
  podConfig:
    nodeSelector:
      node-role.kubernetes.io/infra: ""
      node-role.kubernetes.io/worker: ""
```

5.12.1.6. Configuring node agent load affinity

You can schedule the node agent pods on specific nodes by using the **spec.podConfig.nodeSelector** object of the **DataProtectionApplication** (DPA) custom resource (CR).

See the following example in which you can schedule the node agent pods on nodes with the label **label.io/role: cpu-1** and **other-label.io/other-role: cpu-2**.

```
...
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
    podConfig:
      nodeSelector:
        label.io/role: cpu-1
        other-label.io/other-role: cpu-2
  ...
```

You can add more restrictions on the node agent pods scheduling by using the **nodeagent.loadAffinity** object in the DPA spec.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the DPA spec **nodeagent.loadAffinity** object as shown in the following example. In the example, you ensure that the node agent pods are scheduled only on nodes with the label **label.io/role: cpu-1** and the label **label.io/hostname** matching with either **node1** or **node2**.

```
...
spec:
  configuration:
    nodeAgent:
      enable: true
```

```

loadAffinity:
  - nodeSelector:
      matchLabels:
        label.io/role: cpu-1
      matchExpressions:
        - key: label.io/hostname
          operator: In
          values:
            - node1
            - node2
            ...

```

where:

loadAffinity

Specifies the **loadAffinity** object by adding the **matchLabels** and **matchExpressions** objects.

matchExpressions

Specifies the **matchExpressions** object to add restrictions on the node agent pods scheduling.

5.12.1.7. Node agent load affinity guidelines

Use the following guidelines to configure the node agent **loadAffinity** object in the **DataProtectionApplication** (DPA) custom resource (CR).

- Use the **spec.nodeagent.podConfig.nodeSelector** object for simple node matching.
- Use the **loadAffinity.nodeSelector** object without the **podConfig.nodeSelector** object for more complex scenarios.
- You can use both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects, but the **loadAffinity** object must be equal or more restrictive as compared to the **podConfig** object. In this scenario, the **podConfig.nodeSelector** labels must be a subset of the labels used in the **loadAffinity.nodeSelector** object.
- You cannot use the **matchExpressions** and **matchLabels** fields if you have configured both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects in the DPA.
- See the following example to configure both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects in the DPA.

```

...
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
      loadAffinity:
        - nodeSelector:
            matchLabels:
              label.io/location: 'US'
              label.io/gpu: 'no'

```

```

podConfig:
  nodeSelector:
    label.io/gpu: 'no'

```

5.12.1.8. Configuring node agent load concurrency

You can control the maximum number of node agent operations that can run simultaneously on each node within your cluster.

You can configure it using one of the following fields of the Data Protection Application (DPA):

- **globalConfig**: Defines a default concurrency limit for the node agent across all nodes.
- **perNodeConfig**: Specifies different concurrency limits for specific nodes based on **nodeSelector** labels. This provides flexibility for environments where certain nodes might have different resource capacities or roles.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.

Procedure

1. If you want to use load concurrency for specific nodes, add labels to those nodes:

```
$ oc label node/<node_name> label.io/instance-type='large'
```

2. Configure the load concurrency fields for your DPA instance:

```

configuration:
  nodeAgent:
    enable: true
    uploaderType: kopia
    loadConcurrency:
      globalConfig: 1
      perNodeConfig:
        - nodeSelector:
            matchLabels:
              label.io/instance-type: large
          number: 3

```

where:

globalConfig

Specifies the global concurrent number. The default value is 1, which means there is no concurrency and only one load is allowed. The **globalConfig** value does not have a limit.

label.io/instance-type

Specifies the label for per-node concurrency.

number

Specifies the per-node concurrent number. You can specify many per-node concurrent numbers, for example, based on the instance type and size. The range of per-node concurrent number is the same as the global concurrent number. If the configuration file

contains a per-node concurrent number and a global concurrent number, the per-node concurrent number takes precedence.

5.12.1.9. Configuring the node agent as a non-root and non-privileged user

To enhance the node agent security, you can configure the OADP Operator node agent daemonset to run as a non-root and non-privileged user by using the **`spec.configuration.velero.disableFsBackup`** setting in the **DataProtectionApplication** (DPA) custom resource (CR).

By setting the **`spec.configuration.velero.disableFsBackup`** setting to **`true`**, the node agent security context sets the root file system to read-only and sets the **`privileged`** flag to **`false`**.



NOTE

Setting **`spec.configuration.velero.disableFsBackup`** to **`true`** enhances the node agent security by removing the need for privileged containers and enforcing a read-only root file system.

However, it also disables File System Backup (FSB) with Kopia. If your workloads rely on FSB for backing up volumes that do not support native snapshots, then you should evaluate whether the **`disableFsBackup`** configuration fits your use case.

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **`disableFsBackup`** field in the DPA as shown in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: ts-dpa
  namespace: openshift-adp
spec:
  backupLocations:
  - velero:
      credential:
        key: cloud
        name: cloud-credentials
      default: true
      objectStorage:
        bucket: <bucket_name>
        prefix: velero
        provider: gcp
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
    velero:
      defaultPlugins:
      - csi
```



```
- gcp
- openshift
disableFsBackup: true
```

where:

nodeAgent

Specifies to enable the node agent in the DPA.

disableFsBackup

Specifies to set the **disableFsBackup** field to **true**.

Verification

1. Verify that the node agent security context is set to run as non-root and the root file system is **readOnly** by running the following command:

```
$ oc get daemonset node-agent -o yaml
```

The example output is as following:

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  ...
  name: node-agent
  namespace: openshift-adp
  ...
spec:
  ...
  template:
    metadata:
      ...
    spec:
      containers:
        ...
        securityContext:
          allowPrivilegeEscalation: false
          capabilities:
            drop:
              - ALL
          privileged: false
          readOnlyRootFilesystem: true
          ...
      nodeSelector:
        kubernetes.io/os: linux
      os:
        name: linux
      restartPolicy: Always
      schedulerName: default-scheduler
      securityContext:
        runAsNonRoot: true
      seccompProfile:
        type: RuntimeDefault
```

```

serviceAccount: velero
serviceAccountName: velero
....

```

where:

allowPrivilegeEscalation

Specifies that the **allowPrivilegeEscalation** field is false.

privileged

Specifies that the **privileged** field is false.

readOnlyRootFilesystem

Specifies that the root file system is read-only.

runAsNonRoot

Specifies that the node agent is run as a non-root user.

5.12.1.10. Configuring repository maintenance

OADP repository maintenance is a background job, you can configure it independently of the node agent pods. This means that you can schedule the repository maintenance pod on a node where the node agent is or is not running.

You can use the repository maintenance job affinity configurations in the **DataProtectionApplication** (DPA) custom resource (CR) only if you use Kopia as the backup repository.

You have the option to configure the load affinity at the global level affecting all repositories. Or you can configure the load affinity for each repository. You can also use a combination of global and per-repository configuration.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the **loadAffinity** object in the DPA spec by using either one or both of the following methods:
 - Global configuration: Configure load affinity for all repositories as shown in the following example:

```

...
spec:
  configuration:
    repositoryMaintenance:
      global:
        podResources:
          cpuRequest: "100m"
          cpuLimit: "200m"
          memoryRequest: "100Mi"

```

```

memoryLimit: "200Mi"
loadAffinity:
  - nodeSelector:
      matchLabels:
        label.io/gpu: 'no'
      matchExpressions:
        - key: label.io/location
          operator: In
          values:
            - US
            - EU

```

where:

repositoryMaintenance

Specifies the **repositoryMaintenance** object as shown in the example.

global

Specifies the **global** object to configure load affinity for all repositories.

- Per-repository configuration: Configure load affinity per repository as shown in the following example:

```

...
spec:
  configuration:
    repositoryMaintenance:
      myrepositoryname:
        loadAffinity:
          - nodeSelector:
              matchLabels:
                label.io/cpu: 'yes'

```

where:

myrepositoryname

Specifies the **repositoryMaintenance** object for each repository.

5.12.1.11. Configuring Velero load affinity

With each OADP deployment, there is one Velero pod and its main purpose is to schedule Velero workloads. To schedule the Velero pod, you can use the **velero.podConfig.nodeSelector** and the **velero.loadAffinity** objects in the **DataProtectionApplication** (DPA) custom resource (CR) spec.

Use the **podConfig.nodeSelector** object to assign the Velero pod to specific nodes. You can also configure the **velero.loadAffinity** object for pod-level affinity and anti-affinity.

The OpenShift scheduler applies the rules and performs the scheduling of the Velero pod deployment.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.

- You have configured the DPA CR.

Procedure

- Configure the **velero.podConfig.nodeSelector** and the **velero.loadAffinity** objects in the DPA spec as shown in the following examples:

- **velero.podConfig.nodeSelector** object configuration:

```
...
spec:
  configuration:
    velero:
      podConfig:
        nodeSelector:
          some-label.io/custom-node-role: backup-core
```

- **velero.loadAffinity** object configuration:

```
...
spec:
  configuration:
    velero:
      loadAffinity:
        - nodeSelector:
            matchLabels:
              label.io/gpu: 'no'
            matchExpressions:
              - key: label.io/location
                operator: In
            values:
              - US
              - EU
```

5.12.1.12. Overriding the imagePullPolicy setting in the DPA

In OADP 1.4.0 or earlier, the Operator sets the **imagePullPolicy** field of the Velero and node agent pods to **Always** for all images.

In OADP 1.4.1 or later, the Operator first checks if each image has the **sha256** or **sha512** digest and sets the **imagePullPolicy** field accordingly:

- If the image has the digest, the Operator sets **imagePullPolicy** to **IfNotPresent**.
- If the image does not have the digest, the Operator sets **imagePullPolicy** to **Always**.

You can also override the **imagePullPolicy** field by using the **spec.imagePullPolicy** field in the Data Protection Application (DPA).

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **spec.imagePullPolicy** field in the DPA as shown in the following example:

Example Data Protection Application

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: test-dpa
  namespace: openshift-adp
spec:
  backupLocations:
  - name: default
    velero:
      config:
        insecureSkipTLSVerify: "true"
        profile: "default"
        region: <bucket_region>
        s3ForcePathStyle: "true"
        s3Url: <bucket_url>
      credential:
        key: cloud
        name: cloud-credentials
      default: true
      objectStorage:
        bucket: <bucket_name>
        prefix: velero
      provider: aws
    configuration:
      nodeAgent:
        enable: true
        uploaderType: kopia
      velero:
        defaultPlugins:
        - openshift
        - aws
        - kubevirt
        - csi
  imagePullPolicy: Never

```

where:

imagePullPolicy

Specifies the value for **imagePullPolicy**. In this example, the **imagePullPolicy** field is set to **Never**.

5.12.1.12.1. Creating an Object Bucket Claim for disaster recovery on OpenShift Data Foundation

If you use cluster storage for your Multicloud Object Gateway (MCG) bucket **backupStorageLocation** on OpenShift Data Foundation, create an Object Bucket Claim (OBC) using the OpenShift Web Console.

**WARNING**

Failure to configure an Object Bucket Claim (OBC) might lead to backups not being available.

**NOTE**

Unless specified otherwise, "NooBaa" refers to the open source project that provides lightweight object storage, while "Multicloud Object Gateway (MCG)" refers to the Red Hat distribution of NooBaa.

For more information on the MCG, see [Accessing the Multicloud Object Gateway with your applications](#).

Procedure

- Create an Object Bucket Claim (OBC) using the OpenShift web console as described in [Creating an Object Bucket Claim using the OpenShift Web Console](#).

5.12.1.12.2. Enabling CSI in the DataProtectionApplication CR

You enable the Container Storage Interface (CSI) in the **DataProtectionApplication** custom resource (CR) in order to back up persistent volumes with CSI snapshots.

Prerequisites

- The cloud provider must support CSI snapshots.

Procedure

- Edit the **DataProtectionApplication** CR, as in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - csi
```

where:

csi

Specifies the **csi** default plugin.

5.12.1.12.3. Disabling the node agent in DataProtectionApplication

If you are not using **Restic**, **Kopia**, or **DataMover** for your backups, you can disable the **nodeAgent** field in the **DataProtectionApplication** custom resource (CR). Before you disable **nodeAgent**, ensure the OADP Operator is idle and not running any backups.

Procedure

1. To disable the **nodeAgent**, set the **enable** flag to **false**. See the following example:

Example DataProtectionApplication CR

```
# ...
configuration:
  nodeAgent:
    enable: false
    uploaderType: kopia
# ...
```

where:

enable

Enables the node agent.

2. To enable the **nodeAgent**, set the **enable** flag to **true**. See the following example:

Example DataProtectionApplication CR

```
# ...
configuration:
  nodeAgent:
    enable: true
    uploaderType: kopia
# ...
```

where:

enable

Enables the node agent.

You can set up a job to enable and disable the **nodeAgent** field in the **DataProtectionApplication** CR. For more information, see "Running tasks in pods using jobs".

Additional resources

- [Installing the Data Protection Application with the **kubevirt** and **openshift** plugins](#)
- [Running tasks in pods using jobs](#).
- [Configuring the OpenShift API for Data Protection \(OADP\) with multiple backup storage locations](#)

5.13. CONFIGURING OADP WITH OPENSIFT VIRTUALIZATION

5.13.1. Configuring the OpenShift API for Data Protection with OpenShift Virtualization

You can install the OpenShift API for Data Protection (OADP) with OpenShift Virtualization by installing the OADP Operator and configuring a backup location. Then, you can install the Data Protection Application.

Back up and restore virtual machines by using the [OpenShift API for Data Protection](#).



NOTE

OpenShift API for Data Protection with OpenShift Virtualization supports the following backup and restore storage options:

- Container Storage Interface (CSI) backups
- Container Storage Interface (CSI) backups with DataMover

The following storage options are excluded:

- File system backup and restore
- Volume snapshot backups and restores

For more information, see [Backing up applications with File System Backup: Kopia or Restic](#).

To install the OADP Operator in a restricted network environment, you must first disable the default software catalog sources and mirror the Operator catalog. See [Using Operator Lifecycle Manager in disconnected environments](#) for details.

5.13.1.1. Installing and configuring OADP with OpenShift Virtualization

As a cluster administrator, you install OADP by installing the OADP Operator.

The latest version of the OADP Operator installs [Velero 1.16](#).

Prerequisites

- Access to the cluster as a user with the **cluster-admin** role.

Procedure

1. Install the OADP Operator according to the instructions for your storage provider.
2. Install the Data Protection Application (DPA) with the **kubevirt** and **openshift** OADP plugins.
3. Back up virtual machines by creating a **Backup** custom resource (CR).

**WARNING**

Red Hat support is limited to only the following options:

- CSI backups
- CSI backups with DataMover.

You restore the **Backup** CR by creating a **Restore** CR.

Additional resources

- [OADP plugins](#)
- [Backup custom resource \(CR\)](#)
- [Restore CR](#)
- [Using Operator Lifecycle Manager in disconnected environments](#)

5.13.1.2. Installing the Data Protection Application

You install the Data Protection Application (DPA) by creating an instance of the **DataProtectionApplication** API.

Prerequisites

- You must install the OADP Operator.
- You must configure object storage as a backup location.
- If you use snapshots to back up PVs, your cloud provider must support either a native snapshot API or Container Storage Interface (CSI) snapshots.
- If the backup and snapshot locations use the same credentials, you must create a **Secret** with the default name, **cloud-credentials**.

**NOTE**

If you do not want to specify backup or snapshot locations during the installation, you can create a default **Secret** with an empty **credentials-velero** file. If there is no default **Secret**, the installation will fail.

Procedure

1. Click **Ecosystem** → **Installed Operators** and select the OADP Operator.
2. Under **Provided APIs**, click **Create instance** in the **DataProtectionApplication** box.
3. Click **YAML View** and update the parameters of the **DataProtectionApplication** manifest:

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  configuration:
    velero:
      defaultPlugins:
        - kubevirt
        - gcp
        - csi
        - openshift
      resourceTimeout: 10m
  nodeAgent:
    enable: true
    uploaderType: kopia
    podConfig:
      nodeSelector: <node_selector>
  backupLocations:
    - velero:
        provider: gcp
        default: true
        credential:
          key: cloud
          name: <default_secret>
        objectStorage:
          bucket: <bucket_name>
          prefix: <prefix>

```

where:

namespace

Specifies the default namespace for OADP which is **openshift-adp**. The namespace is a variable and is configurable.

kubevirt

Specifies that the **kubevirt** plugin is mandatory for OpenShift Virtualization.

gcp

Specifies the plugin for the backup provider, for example, **gcp**, if it exists.

csi

Specifies that the **csi** plugin is mandatory for backing up PVs with CSI snapshots. The **csi** plugin uses the [Velero CSI beta snapshot APIs](#). You do not need to configure a snapshot location.

openshift

Specifies that the **openshift** plugin is mandatory.

resourceTimeout

Specifies how many minutes to wait for several Velero resources such as Velero CRD availability, volumeSnapshot deletion, and backup repository availability, before timeout occurs. The default is 10m.

nodeAgent

Specifies the administrative agent that routes the administrative requests to servers.

enable

Set this value to **true** if you want to enable **nodeAgent** and perform File System Backup.

uploaderType

Specifies the uploader type. Enter **kopia** as your uploader to use the Built-in DataMover. The **nodeAgent** deploys a daemon set, which means that the **nodeAgent** pods run on each working node. You can configure File System Backup by adding **spec.defaultVolumesToFsBackup: true** to the **Backup** CR.

nodeSelector

Specifies the nodes on which Kopia are available. By default, Kopia runs on all nodes.

provider

Specifies the backup provider.

name

Specifies the correct default name for the **Secret**, for example, **cloud-credentials-gcp**, if you use a default plugin for the backup provider. If specifying a custom name, then the custom name is used for the backup location. If you do not specify a **Secret** name, the default name is used.

bucket

Specifies a bucket as the backup storage location. If the bucket is not a dedicated bucket for Velero backups, you must specify a prefix.

prefix

Specifies a prefix for Velero backups, for example, **velero**, if the bucket is used for multiple purposes.

4. Click **Create**.

Verification

1. Verify the installation by viewing the OpenShift API for Data Protection (OADP) resources by running the following command:

```
$ oc get all -n openshift-adp
```

NAME	READY	STATUS	RESTARTS	AGE
pod/oadp-operator-controller-manager-67d9494d47-6l8z8	2/2	Running	0	2m8s
pod/node-agent-9cq4q	1/1	Running	0	94s
pod/node-agent-m4lts	1/1	Running	0	94s
pod/node-agent-pv4kr	1/1	Running	0	95s
pod/velero-588db7f655-n842v	1/1	Running	0	95s

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP
service/oadp-operator-controller-manager-metrics-service	ClusterIP	172.30.70.140	
<none>	8443/TCP	2m8s	
service/openshift-adp-velero-metrics-svc	ClusterIP	172.30.10.0	<none>
8085/TCP	8h		

NAME	DESIRED	CURRENT	READY	UP-TO-DATE	AVAILABLE	NODE
SELECTOR	AGE					

```

daemonset.apps/node-agent 3      3      3      3      3      <none>      96s

NAME                                READY  UP-TO-DATE  AVAILABLE  AGE
deployment.apps/oadp-operator-controller-manager 1/1    1            1          2m9s
deployment.apps/velero              1/1    1            1          96s

NAME                                DESIRED  CURRENT  READY  AGE
replicaset.apps/oadp-operator-controller-manager-67d9494d47 1        1        1      2m9s
replicaset.apps/velero-588db7f655 1        1        1      96s

```

2. Verify that the **DataProtectionApplication** (DPA) is reconciled by running the following command:

```
$ oc get dpa dpa-sample -n openshift-adp -o jsonpath='{.status}'
```

```

{"conditions":[{"lastTransitionTime":"2023-10-27T01:23:57Z","message":"Reconcile
complete","reason":"Complete","status":"True","type":"Reconciled"}]}

```

3. Verify the **type** is set to **Reconciled**.
4. Verify the backup storage location and confirm that the **PHASE** is **Available** by running the following command:

```
$ oc get backupstoragelocations.velero.io -n openshift-adp
```

```

NAME          PHASE    LAST VALIDATED  AGE    DEFAULT
dpa-sample-1  Available 1s             3d16h true

```



WARNING

If you run a backup of a Microsoft Windows virtual machine (VM) immediately after the VM reboots, the backup might fail with a **PartiallyFailed** error. This is because, immediately after a VM boots, the Microsoft Windows Volume Shadow Copy Service (VSS) and Guest Agent (GA) service are not ready. The VSS and GA service being unready causes the backup to fail. In such a case, retry the backup a few minutes after the VM boots.

5.13.1.3. Backing up a single VM

If you have a namespace with multiple virtual machines (VMs), and want to back up only one of them, you can use the label selector to filter the VM that needs to be included in the backup. You can filter the VM by using the **app: vmname** label.

Prerequisites

- You have installed the OADP Operator.
- You have multiple VMs running in a namespace.

- You have added the **kubevirt** plugin in the **DataProtectionApplication** (DPA) custom resource (CR).
- You have configured the **BackupStorageLocation** CR in the **DataProtectionApplication** CR and **BackupStorageLocation** is available.

Procedure

1. Configure the **Backup** CR as shown in the following example:

Example Backup CR

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: vmbackupsingle
  namespace: openshift-adp
spec:
  snapshotMoveData: true
  includedNamespaces:
    - <vm_namespace> ❶
  labelSelector:
    matchLabels:
      app: <vm_app_name> ❷
  storageLocation: <backup_storage_location_name> ❸
```

- ❶ Specify the name of the namespace where you have created the VMs.
- ❷ Specify the VM name that needs to be backed up.
- ❸ Specify the name of the **BackupStorageLocation** CR.

2. To create a **Backup** CR, run the following command:

```
$ oc apply -f <backup_cr_file_name> ❶
```

- ❶ Specify the name of the **Backup** CR file.

5.13.1.4. Restoring a single VM

After you have backed up a single virtual machine (VM) by using the label selector in the **Backup** custom resource (CR), you can create a **Restore** CR and point it to the backup. This restore operation restores a single VM.

Prerequisites

- You have installed the OADP Operator.
- You have backed up a single VM by using the label selector.

Procedure

1. Configure the **Restore** CR as shown in the following example:

Example Restore CR

```
apiVersion: velero.io/v1
kind: Restore
metadata:
  name: vmrestoresingle
  namespace: openshift-adp
spec:
  backupName: vmbackupsingle 1
  restorePVs: true
```

- 1 Specifies the name of the backup of a single VM.

2. To restore the single VM, run the following command:

```
$ oc apply -f <restore_cr_file_name> 1
```

- 1 Specify the name of the **Restore** CR file.



NOTE

When you restore a backup of VMs, you might notice that the Ceph storage capacity allocated for the restore is higher than expected. This behavior is observed only during the **kubevirt** restore and if the volume type of the VM is **block**.

Use the **rbd sparsify** tool to reclaim space on target volumes. For more details, see [Reclaiming space on target volumes](#).

5.13.1.5. Restoring a single VM from a backup of multiple VMs

If you have a backup containing multiple virtual machines (VMs), and you want to restore only one VM, you can use the **LabelSelectors** section in the **Restore** CR to select the VM to restore. To ensure that the persistent volume claim (PVC) attached to the VM is correctly restored, and the restored VM is not stuck in a **Provisioning** status, use both the **app: <vm_name>** and the **kubevirt.io/created-by** labels. To match the **kubevirt.io/created-by** label, use the UID of **DataVolume** of the VM.

Prerequisites

- You have installed the OADP Operator.
- You have labeled the VMs that need to be backed up.
- You have a backup of multiple VMs.

Procedure

1. Before you take a backup of many VMs, ensure that the VMs are labeled by running the following command:

■

```
$ oc label vm <vm_name> app=<vm_name> -n openshift-adp
```

2. Configure the label selectors in the **Restore** CR as shown in the following example:

Example Restore CR

```
apiVersion: velero.io/v1
kind: Restore
metadata:
  name: singlevmrestore
  namespace: openshift-adp
spec:
  backupName: multiplevmbackup
  restorePVs: true
  LabelSelectors:
    - matchLabels:
        kubevirt.io/created-by: <datavolume_uid> ❶
    - matchLabels:
        app: <vm_name> ❷
```

- ❶ Specify the UID of **DataVolume** of the VM that you want to restore. For example, **b6...53a-ddd7-4d9d-9407-a0c...e5**.
- ❷ Specify the name of the VM that you want to restore. For example, **test-vm**.

3. To restore a VM, run the following command:

```
$ oc apply -f <restore_cr_file_name> ❶
```

- ❶ Specify the name of the **Restore** CR file.

5.13.1.6. Configuring the DPA with client burst and QPS settings

The burst setting determines how many requests can be sent to the **velero** server before the limit is applied. After the burst limit is reached, the queries per second (QPS) setting determines how many additional requests can be sent per second.

You can set the burst and QPS values of the **velero** server by configuring the Data Protection Application (DPA) with the burst and QPS values. You can use the **dpa.configuration.velero.client-burst** and **dpa.configuration.velero.client-qps** fields of the DPA to set the burst and QPS values.

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **client-burst** and the **client-qps** fields in the DPA as shown in the following example:

Example Data Protection Application

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: test-dpa
  namespace: openshift-adp
spec:
  backupLocations:
    - name: default
      velero:
        config:
          insecureSkipTLSVerify: "true"
          profile: "default"
          region: <bucket_region>
          s3ForcePathStyle: "true"
          s3Url: <bucket_url>
        credential:
          key: cloud
          name: cloud-credentials
        default: true
        objectStorage:
          bucket: <bucket_name>
          prefix: velero
          provider: aws
      configuration:
        nodeAgent:
          enable: true
          uploaderType: restic
        velero:
          client-burst: 500
          client-qps: 300
          defaultPlugins:
            - openshift
            - aws
            - kubevirt

```

where:

client-burst

Specifies the **client-burst** value. In this example, the **client-burst** field is set to 500.

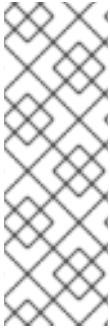
client-qps

Specifies the **client-qps** value. In this example, the **client-qps** field is set to 300.

5.13.1.7. Configuring the node agent as a non-root and non-privileged user

To enhance the node agent security, you can configure the OADP Operator node agent daemonset to run as a non-root and non-privileged user by using the **spec.configuration.velero.disableFsBackup** setting in the **DataProtectionApplication** (DPA) custom resource (CR).

By setting the **spec.configuration.velero.disableFsBackup** setting to **true**, the node agent security context sets the root file system to read-only and sets the **privileged** flag to **false**.



NOTE

Setting **spec.configuration.velero.disableFsBackup** to **true** enhances the node agent security by removing the need for privileged containers and enforcing a read-only root file system.

However, it also disables File System Backup (FSB) with Kopia. If your workloads rely on FSB for backing up volumes that do not support native snapshots, then you should evaluate whether the **disableFsBackup** configuration fits your use case.

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **disableFsBackup** field in the DPA as shown in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: ts-dpa
  namespace: openshift-adp
spec:
  backupLocations:
  - velero:
      credential:
        key: cloud
        name: cloud-credentials
      default: true
      objectStorage:
        bucket: <bucket_name>
        prefix: velero
      provider: gcp
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
    velero:
      defaultPlugins:
      - csi
      - gcp
      - openshift
      disableFsBackup: true
```

where:

nodeAgent

Specifies to enable the node agent in the DPA.

disableFsBackup

Specifies to set the **disableFsBackup** field to **true**.

Verification

1. Verify that the node agent security context is set to run as non-root and the root file system is **readOnly** by running the following command:

```
$ oc get daemonset node-agent -o yaml
```

The example output is as following:

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  ...
  name: node-agent
  namespace: openshift-adp
  ...
spec:
  ...
  template:
    metadata:
      ...
    spec:
      containers:
        ...
        securityContext:
          allowPrivilegeEscalation: false
          capabilities:
            drop:
              - ALL
          privileged: false
          readOnlyRootFilesystem: true
          ...
      nodeSelector:
        kubernetes.io/os: linux
      os:
        name: linux
      restartPolicy: Always
      schedulerName: default-scheduler
      securityContext:
        runAsNonRoot: true
        seccompProfile:
          type: RuntimeDefault
      serviceAccount: velero
      serviceAccountName: velero
      ....
```

where:

allowPrivilegeEscalation

Specifies that the **allowPrivilegeEscalation** field is false.

privileged

Specifies that the **privileged** field is false.

readOnlyRootFilesystem

Specifies that the root file system is read-only.

runAsNonRoot

Specifies that the node agent is run as a non-root user.

5.13.1.8. Configuring node agents and node labels

The Data Protection Application (DPA) uses the **nodeSelector** field to select which nodes can run the node agent. The **nodeSelector** field is the recommended form of node selection constraint.

Procedure

1. Run the node agent on any node that you choose by adding a custom label:

```
$ oc label node/<node_name> node-role.kubernetes.io/nodeAgent=""
```



NOTE

Any label specified must match the labels on each node.

2. Use the same custom label in the **DPA.spec.configuration.nodeAgent.podConfig.nodeSelector** field, which you used for labeling nodes:

```
configuration:
  nodeAgent:
    enable: true
  podConfig:
    nodeSelector:
      node-role.kubernetes.io/nodeAgent: ""
```

The following example is an anti-pattern of **nodeSelector** and does not work unless both labels, **node-role.kubernetes.io/infra: ""** and **node-role.kubernetes.io/worker: ""**, are on the node:

```
configuration:
  nodeAgent:
    enable: true
  podConfig:
    nodeSelector:
      node-role.kubernetes.io/infra: ""
      node-role.kubernetes.io/worker: ""
```

5.13.1.9. Configuring node agent load affinity

You can schedule the node agent pods on specific nodes by using the **spec.podConfig.nodeSelector** object of the **DataProtectionApplication** (DPA) custom resource (CR).

See the following example in which you can schedule the node agent pods on nodes with the label **label.io/role: cpu-1** and **other-label.io/other-role: cpu-2**.

```
...
spec:
  configuration:
    nodeAgent:
      enable: true
```

```

uploaderType: kopia
podConfig:
  nodeSelector:
    label.io/role: cpu-1
    other-label.io/other-role: cpu-2
  ...

```

You can add more restrictions on the node agent pods scheduling by using the **nodeagent.loadAffinity** object in the DPA spec.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the DPA spec **nodeagent.loadAffinity** object as shown in the following example. In the example, you ensure that the node agent pods are scheduled only on nodes with the label **label.io/role: cpu-1** and the label **label.io/hostname** matching with either **node1** or **node2**.

```

...
spec:
  configuration:
    nodeAgent:
      enable: true
      loadAffinity:
        - nodeSelector:
            matchLabels:
              label.io/role: cpu-1
            matchExpressions:
              - key: label.io/hostname
                operator: In
                values:
                  - node1
                  - node2
        ...

```

where:

loadAffinity

Specifies the **loadAffinity** object by adding the **matchLabels** and **matchExpressions** objects.

matchExpressions

Specifies the **matchExpressions** object to add restrictions on the node agent pods scheduling.

5.13.1.10. Node agent load affinity guidelines

Use the following guidelines to configure the node agent **loadAffinity** object in the **DataProtectionApplication** (DPA) custom resource (CR).

- Use the **spec.nodeagent.podConfig.nodeSelector** object for simple node matching.
- Use the **loadAffinity.nodeSelector** object without the **podConfig.nodeSelector** object for more complex scenarios.
- You can use both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects, but the **loadAffinity** object must be equal or more restrictive as compared to the **podConfig** object. In this scenario, the **podConfig.nodeSelector** labels must be a subset of the labels used in the **loadAffinity.nodeSelector** object.
- You cannot use the **matchExpressions** and **matchLabels** fields if you have configured both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects in the DPA.
- See the following example to configure both **podConfig.nodeSelector** and **loadAffinity.nodeSelector** objects in the DPA.

```
...
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
      loadAffinity:
        - nodeSelector:
            matchLabels:
              label.io/location: 'US'
              label.io/gpu: 'no'
      podConfig:
        nodeSelector:
          label.io/gpu: 'no'
```

5.13.1.11. Configuring node agent load concurrency

You can control the maximum number of node agent operations that can run simultaneously on each node within your cluster.

You can configure it using one of the following fields of the Data Protection Application (DPA):

- **globalConfig**: Defines a default concurrency limit for the node agent across all nodes.
- **perNodeConfig**: Specifies different concurrency limits for specific nodes based on **nodeSelector** labels. This provides flexibility for environments where certain nodes might have different resource capacities or roles.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.

Procedure

1. If you want to use load concurrency for specific nodes, add labels to those nodes:

■

```
$ oc label node/<node_name> label.io/instance-type='large'
```

2. Configure the load concurrency fields for your DPA instance:

```
configuration:
  nodeAgent:
    enable: true
    uploaderType: kopia
    loadConcurrency:
      globalConfig: 1
      perNodeConfig:
        - nodeSelector:
            matchLabels:
              label.io/instance-type: large
            number: 3
```

where:

globalConfig

Specifies the global concurrent number. The default value is 1, which means there is no concurrency and only one load is allowed. The **globalConfig** value does not have a limit.

label.io/instance-type

Specifies the label for per-node concurrency.

number

Specifies the per-node concurrent number. You can specify many per-node concurrent numbers, for example, based on the instance type and size. The range of per-node concurrent number is the same as the global concurrent number. If the configuration file contains a per-node concurrent number and a global concurrent number, the per-node concurrent number takes precedence.

5.13.12. Configuring repository maintenance

OADP repository maintenance is a background job, you can configure it independently of the node agent pods. This means that you can schedule the repository maintenance pod on a node where the node agent is or is not running.

You can use the repository maintenance job affinity configurations in the **DataProtectionApplication** (DPA) custom resource (CR) only if you use Kopia as the backup repository.

You have the option to configure the load affinity at the global level affecting all repositories. Or you can configure the load affinity for each repository. You can also use a combination of global and per-repository configuration.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the **loadAffinity** object in the DPA spec by using either one or both of the following methods:
 - Global configuration: Configure load affinity for all repositories as shown in the following example:

```
...
spec:
  configuration:
    repositoryMaintenance:
      global:
        podResources:
          cpuRequest: "100m"
          cpuLimit: "200m"
          memoryRequest: "100Mi"
          memoryLimit: "200Mi"
        loadAffinity:
          - nodeSelector:
              matchLabels:
                label.io/gpu: 'no'
            matchExpressions:
              - key: label.io/location
                operator: In
                values:
                  - US
                  - EU
```

where:

repositoryMaintenance

Specifies the **repositoryMaintenance** object as shown in the example.

global

Specifies the **global** object to configure load affinity for all repositories.

- Per-repository configuration: Configure load affinity per repository as shown in the following example:

```
...
spec:
  configuration:
    repositoryMaintenance:
      myrepositoryname:
        loadAffinity:
          - nodeSelector:
              matchLabels:
                label.io/cpu: 'yes'
```

where:

myrepositoryname

Specifies the **repositoryMaintenance** object for each repository.

5.13.13. Configuring Velero load affinity

With each OADP deployment, there is one Velero pod and its main purpose is to schedule Velero workloads. To schedule the Velero pod, you can use the **velero.podConfig.nodeSelector** and the **velero.loadAffinity** objects in the **DataProtectionApplication** (DPA) custom resource (CR) spec.

Use the **podConfig.nodeSelector** object to assign the Velero pod to specific nodes. You can also configure the **velero.loadAffinity** object for pod-level affinity and anti-affinity.

The OpenShift scheduler applies the rules and performs the scheduling of the Velero pod deployment.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- You have installed the OADP Operator.
- You have configured the DPA CR.

Procedure

- Configure the **velero.podConfig.nodeSelector** and the **velero.loadAffinity** objects in the DPA spec as shown in the following examples:
 - **velero.podConfig.nodeSelector** object configuration:

```
...
spec:
  configuration:
    velero:
      podConfig:
        nodeSelector:
          some-label.io/custom-node-role: backup-core
```

- **velero.loadAffinity** object configuration:

```
...
spec:
  configuration:
    velero:
      loadAffinity:
        - nodeSelector:
            matchLabels:
              label.io/gpu: 'no'
            matchExpressions:
              - key: label.io/location
                operator: In
            values:
              - US
              - EU
```

5.13.1.14. Overriding the imagePullPolicy setting in the DPA

In OADP 1.4.0 or earlier, the Operator sets the **imagePullPolicy** field of the Velero and node agent pods to **Always** for all images.

In OADP 1.4.1 or later, the Operator first checks if each image has the **sha256** or **sha512** digest and sets the **imagePullPolicy** field accordingly:

- If the image has the digest, the Operator sets **imagePullPolicy** to **IfNotPresent**.
- If the image does not have the digest, the Operator sets **imagePullPolicy** to **Always**.

You can also override the **imagePullPolicy** field by using the **spec.imagePullPolicy** field in the Data Protection Application (DPA).

Prerequisites

- You have installed the OADP Operator.

Procedure

- Configure the **spec.imagePullPolicy** field in the DPA as shown in the following example:

Example Data Protection Application

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: test-dpa
  namespace: openshift-adp
spec:
  backupLocations:
    - name: default
      velero:
        config:
          insecureSkipTLSVerify: "true"
          profile: "default"
          region: <bucket_region>
          s3ForcePathStyle: "true"
          s3Url: <bucket_url>
        credential:
          key: cloud
          name: cloud-credentials
        default: true
        objectStorage:
          bucket: <bucket_name>
          prefix: velero
        provider: aws
      configuration:
        nodeAgent:
          enable: true
          uploaderType: kopia
        velero:
          defaultPlugins:
            - openshift
            - aws
            - kubevirt
            - csi
  imagePullPolicy: Never
```

where:

imagePullPolicy

Specifies the value for **imagePullPolicy**. In this example, the **imagePullPolicy** field is set to **Never**.

5.13.1.15. About incremental back up support

OADP supports incremental backups of **block** and **Filesystem** persistent volumes for both containerized, and OpenShift Virtualization workloads. The following table summarizes the support for File System Backup (FSB), Container Storage Interface (CSI), and CSI Data Mover:

Table 5.4. OADP backup support matrix for containerized workloads

Volume mode	FSB - Restic	FSB - Kopia	CSI	CSI Data Mover
Filesystem	S ^[1] , I ^[2]	S ^[1] , I ^[2]	S ^[1]	S ^[1] , I ^[2]
Block	N ^[3]	N ^[3]	S ^[1]	S ^[1] , I ^[2]

Table 5.5. OADP backup support matrix for OpenShift Virtualization workloads

Volume mode	FSB - Restic	FSB - Kopia	CSI	CSI Data Mover
Filesystem	N ^[3]	N ^[3]	S ^[1]	S ^[1] , I ^[2]
Block	N ^[3]	N ^[3]	S ^[1]	S ^[1] , I ^[2]

- 1. Backup supported
- 2. Incremental backup supported
- 3. Not supported



NOTE

The CSI Data Mover backups use Kopia regardless of **uploaderType**.



IMPORTANT

Red Hat only supports the combination of OADP versions 1.3.0 and later, and OpenShift Virtualization versions 4.14 and later.

OADP versions before 1.3.0 are not supported for back up and restore of OpenShift Virtualization.

5.14. CONFIGURING OADP WITH MULTIPLE BACKUP STORAGE LOCATIONS

5.14.1. Configuring the OpenShift API for Data Protection (OADP) with more than one Backup Storage Location

Configure multiple backup storage locations (BSLs) in the Data Protection Application (DPA) to store backups across different regions or storage providers. This provides flexibility and redundancy for your backup strategy.

OADP supports multiple credentials for configuring more than one BSL, so that you can specify the credentials to use with any BSL.

5.14.1.1. Configuring the DPA with more than one BSL

Configure the **DataProtectionApplication** (DPA) custom resource (CR) with multiple **BackupStorageLocation** (BSL) resources to store backups across different locations using provider-specific credentials. This provides backup distribution and location-specific restore capabilities.

For example, you have configured the following two BSLs:

- Configured one BSL in the DPA and set it as the default BSL.
- Created another BSL independently by using the **BackupStorageLocation** CR.

As you have already set the BSL created through the DPA as the default, you cannot set the independently created BSL again as the default. This means, at any given time, you can set only one BSL as the default BSL.

Prerequisites

- You must install the OADP Operator.
- You must create the secrets by using the credentials provided by the cloud provider.

Procedure

1. Configure the **DataProtectionApplication** CR with more than one **BackupStorageLocation** CR. See the following example:

Example DPA

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
#...
backupLocations:
- name: aws
  velero:
    provider: aws
    default: true
    objectStorage:
      bucket: <bucket_name>
      prefix: <prefix>
    config:
      region: <region_name>
      profile: "default"
    credential:
      key: cloud
```

```

    name: cloud-credentials
- name: odf
  velero:
    provider: aws
    default: false
    objectStorage:
      bucket: <bucket_name>
      prefix: <prefix>
    config:
      profile: "default"
      region: <region_name>
      s3Url: <url>
      insecureSkipTLSVerify: "true"
      s3ForcePathStyle: "true"
    credential:
      key: cloud
      name: <custom_secret_name_odf>
#...
```

where:

name: aws

Specifies a name for the first BSL.

default: true

Indicates that this BSL is the default BSL. If a BSL is not set in the **Backup CR**, the default BSL is used. You can set only one BSL as the default.

<bucket_name>

Specifies the bucket name.

<prefix>

Specifies a prefix for Velero backups. For example, **velero**.

<region_name>

Specifies the AWS region for the bucket.

cloud-credentials

Specifies the name of the default **Secret** object that you created.

name: odf

Specifies a name for the second BSL.

<url>

Specifies the URL of the S3 endpoint.

<custom_secret_name_odf>

Specifies the correct name for the **Secret**. For example, **custom_secret_name_odf**. If you do not specify a **Secret** name, the default name is used.

2. Specify the BSL to be used in the backup CR. See the following example.

Example backup CR

```

apiVersion: velero.io/v1
kind: Backup
# ...
```

```
spec:
  includedNamespaces:
  - <namespace>
  storageLocation: <backup_storage_location>
  defaultVolumesToFsBackup: true
```

where:

<namespace>

Specifies the namespace to back up.

<backup_storage_location>

Specifies the storage location.

5.14.1.2. Configuring two backup BSLs with different cloud credentials

Configure two backup storage locations with different cloud credentials to back up applications to multiple storage targets. With this setup, you can distribute backups across different storage providers for redundancy.

Prerequisites

- You must install the OADP Operator.
- You must configure two backup storage locations: AWS S3 and Multicloud Object Gateway (MCG).
- You must have an application with a database deployed on a Red Hat OpenShift cluster.

Procedure

1. Create the first **Secret** for the AWS S3 storage provider with the default name by running the following command:

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file cloud=
<aws_credentials_file_name>
```

where:

<aws_credentials_file_name>

Specifies the name of the cloud credentials file for AWS S3.

2. Create the second **Secret** for MCG with a custom name by running the following command:

```
$ oc create secret generic mcg-secret -n openshift-adp --from-file cloud=
<MCG_credentials_file_name>
```

where:

<MCG_credentials_file_name>

Specifies the name of the cloud credentials file for MCG. Note the name of the **mcg-secret** custom secret.

3. Configure the DPA with the two BSLs as shown in the following example.

Example DPA

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: two-bsl-dpa
  namespace: openshift-adp
spec:
  backupLocations:
  - name: aws
    velero:
      config:
        profile: default
        region: <region_name>
      credential:
        key: cloud
        name: cloud-credentials
      default: true
      objectStorage:
        bucket: <bucket_name>
        prefix: velero
      provider: aws
  - name: mcg
    velero:
      config:
        insecureSkipTLSVerify: "true"
        profile: noobaa
        region: <region_name>
        s3ForcePathStyle: "true"
        s3Url: <s3_url>
      credential:
        key: cloud
        name: mcg-secret
      objectStorage:
        bucket: <bucket_name_mcg>
        prefix: velero
      provider: aws
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
    velero:
      defaultPlugins:
      - openshift
      - aws

```

where:

<region_name>

Specifies the AWS region for the bucket.

<bucket_name>

Specifies the AWS S3 bucket name.

region: <region_name>

Specifies the region, following the naming convention of the documentation of MCG.

<s3_url>

Specifies the URL of the S3 endpoint for MCG.

mcg-secret

Specifies the name of the custom secret for MCG storage.

<bucket_name_mcg>

Specifies the MCG bucket name.

4. Create the DPA by running the following command:

```
$ oc create -f <dpa_file_name>
```

where:

<dpa_file_name>

Specifies the file name of the DPA you configured.

5. Verify that the DPA has reconciled by running the following command:

```
$ oc get dpa -o yaml
```

6. Verify that the BSLs are available by running the following command:

```
$ oc get bsl
```

Example output

```
NAME   PHASE    LAST VALIDATED  AGE    DEFAULT
aws    Available 5s              3m28s true
mcg    Available 5s              3m28s
```

7. Create a backup CR with the default BSL.

**NOTE**

In the following example, the **storageLocation** field is not specified in the backup CR.

Example backup CR

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: test-backup1
  namespace: openshift-adp
spec:
  includedNamespaces:
    - <mysql_namespace>
  defaultVolumesToFsBackup: true
```

where:

<mysql_namespace>

Specifies the namespace for the application installed in the cluster.

8. Create a backup by running the following command:

```
$ oc apply -f <backup_file_name>
```

where:

<backup_file_name>

Specifies the name of the backup CR file.

9. Verify that the backup completed with the default BSL by running the following command:

```
$ oc get backups.velero.io <backup_name> -o yaml
```

where:

<backup_name>

Specifies the name of the backup.

10. Create a backup CR by using MCG as the BSL. In the following example, note that the second **storageLocation** value is specified at the time of backup CR creation.

Example backup CR

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: test-backup1
  namespace: openshift-adp
spec:
  includedNamespaces:
    - <mysql_namespace>
  storageLocation: mcg
  defaultVolumesToFsBackup: true
```

where:

<mysql_namespace>

Specifies the namespace for the application installed in the cluster.

mcg

Specifies the second storage location.

11. Create a second backup by running the following command:

```
$ oc apply -f <backup_file_name>
```

where:

<backup_file_name>

Specifies the name of the backup CR file.

12. Verify that the backup completed with the storage location as MCG by running the following command:

```
$ oc get backups.velero.io <backup_name> -o yaml
```

where:

<backup_name>

Specifies the name of the backup.

Additional resources

- [Creating profiles for different credentials](#)

5.15. CONFIGURING OADP WITH MULTIPLE VOLUME SNAPSHOT LOCATIONS

5.15.1. Configuring the OpenShift API for Data Protection (OADP) with more than one Volume Snapshot Location

Configure multiple Volume Snapshot Locations (VSLs) in the Data Protection Application (DPA) to store volume snapshots across different cloud provider regions. This provides geographic redundancy and regional disaster recovery capabilities.

5.15.1.1. Configuring the DPA with more than one VSL

Configure the **DataProtectionApplication** (DPA) custom resource (CR) with multiple Volume Snapshot Locations (VSLs) using provider-specific credentials in the same region as your persistent volumes. This provides volume snapshot distribution across different storage targets.

Procedure

- Configure the DPA CR with more than one VSL as shown in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
#...
snapshotLocations:
- velero:
  config:
    profile: default
    region: <region>
  credential:
    key: cloud
    name: cloud-credentials
  provider: aws
- velero:
  config:
    profile: default
    region: <region>
  credential:
    key: cloud
```

```
name: <custom_credential>
provider: aws
#...
```

where:

<region>

Specifies the region. The snapshot location must be in the same region as the persistent volumes.

<custom_credential>

Specifies the custom credential name.

5.16. UNINSTALLING OADP

5.16.1. Uninstalling the OpenShift API for Data Protection

You uninstall the OpenShift API for Data Protection (OADP) by deleting the OADP Operator. See [Deleting Operators from a cluster](#) for details.

5.17. OADP BACKING UP

5.17.1. Backing up applications

Frequent backups might consume storage on the backup storage location. Check the frequency of backups, retention time, and the amount of data of the persistent volumes (PVs) if using non-local backups, for example, S3 buckets. Because all taken backup remains until expired, also check the time to live (TTL) setting of the schedule.

You can back up applications by creating a **Backup** custom resource (CR). For more information, see [Creating a Backup CR](#). The following are the different backup types for a **Backup** CR:

- The **Backup** CR creates backup files for Kubernetes resources and internal images on S3 object storage.
- If you use Velero's snapshot feature to back up data stored on the persistent volume, only snapshot related information is stored in the S3 bucket along with the Openshift object data.
- If your cloud provider has a native snapshot API or supports CSI snapshots, the **Backup** CR backs up persistent volumes (PVs) by creating snapshots. For more information about working with CSI snapshots, see [Backing up persistent volumes with CSI snapshots](#).

If the underlying storage or the backup bucket are part of the same cluster, then the data might be lost in case of disaster.

For more information about CSI volume snapshots, see [CSI volume snapshots](#).

- If your cloud provider does not support snapshots or if your applications are on NFS data volumes, you can create backups by using Kopia or Restic. See [Backing up applications with File System Backup: Kopia or Restic](#).



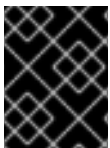
PODVOLUMERESTORE FAILS WITH A .../.SNAPSHOT: READ-ONLY FILE SYSTEM ERROR

The .../.**snapshot** directory is a snapshot copy directory, which is used by several NFS servers. This directory has read-only access by default, so Velero cannot restore to this directory.

Do not give Velero write access to the **.snapshot** directory, and disable client access to this directory.

Additional resources

- [Enable or disable client access to Snapshot copy directory by editing a share](#)
- [Prerequisites for backup and restore with FlashBlade](#)



IMPORTANT

The OpenShift API for Data Protection (OADP) does not support backing up volume snapshots that were created by other software.

5.17.1.1. Previewing resources before running backup and restore

OADP backs up application resources based on the type, namespace, or label. This means that you can view the resources after the backup is complete. Similarly, you can view the restored objects based on the namespace, persistent volume (PV), or label after a restore operation is complete. To preview the resources in advance, you can do a dry run of the backup and restore operations.

Prerequisites

- You have installed the OADP Operator.

Procedure

1. To preview the resources included in the backup before running the actual backup, run the following command:

```
$ velero backup create <backup-name> --snapshot-volumes false ❶
```

- ❶ Specify the value of **--snapshot-volumes** parameter as **false**.

2. To know more details about the backup resources, run the following command:

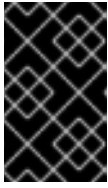
```
$ velero describe backup <backup_name> --details ❶
```

- ❶ Specify the name of the backup.

3. To preview the resources included in the restore before running the actual restore, run the following command:

```
$ velero restore create --from-backup <backup-name> ❶
```

- 1 Specify the name of the backup created to review the backup resources.



IMPORTANT

The **velero restore create** command creates restore resources in the cluster. You must delete the resources created as part of the restore, after you review the resources.

4. To know more details about the restore resources, run the following command:

```
$ velero describe restore <restore_name> --details 1
```

- 1 Specify the name of the restore.

You can create backup hooks to run commands before or after the backup operation. See [Creating backup hooks](#).

You can schedule backups by creating a **Schedule** CR instead of a **Backup** CR. See [Scheduling backups using Schedule CR](#).

5.17.1.2. Known issues

OpenShift Container Platform 4.20 enforces a pod security admission (PSA) policy that can hinder the readiness of pods during a Restic restore process.

This issue has been resolved in the OADP 1.1.6 and OADP 1.2.2 releases, therefore it is recommended that users upgrade to these releases.

For more information, see [Restic restore partially failing on OCP 4.15 due to changed PSA policy](#).

Additional resources

- [Installing Operators on clusters for administrators](#)
- [Installing Operators in namespaces for non-administrators](#)

5.17.2. Creating a Backup CR

You back up Kubernetes resources, internal images, and persistent volumes (PVs) by creating a **Backup** custom resource (CR).

Prerequisites

- You must install the OpenShift API for Data Protection (OADP) Operator.
- The **DataProtectionApplication** CR must be in a **Ready** state.
- Backup location prerequisites:
 - You must have S3 object storage configured for Velero.
 - You must have a backup location configured in the **DataProtectionApplication** CR.

- Snapshot location prerequisites:
 - Your cloud provider must have a native snapshot API or support Container Storage Interface (CSI) snapshots.
 - For CSI snapshots, you must create a **VolumeSnapshotClass** CR to register the CSI driver.
 - You must have a volume location configured in the **DataProtectionApplication** CR.

Procedure

1. Retrieve the **backupStorageLocations** CRs by entering the following command:

```
$ oc get backupstoragelocations.velero.io -n openshift-adp
```

Example output

NAMESPACE	NAME	PHASE	LAST VALIDATED	AGE	DEFAULT
openshift-adp	velero-sample-1	Available	11s	31m	

2. Create a **Backup** CR, as in the following example:

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: <backup>
  labels:
    velero.io/storage-location: default
  namespace: openshift-adp
spec:
  hooks: {}
  includedNamespaces:
    - <namespace> ❶
  includedResources: [] ❷
  excludedResources: [] ❸
  storageLocation: <velero-sample-1> ❹
  ttl: 720h0m0s
  labelSelector: ❺
    matchLabels:
      app: <label_1>
      app: <label_2>
      app: <label_3>
  orLabelSelectors: ❻
    - matchLabels:
      app: <label_1>
      app: <label_2>
      app: <label_3>
```

- ❶ Specify an array of namespaces to back up.
- ❷ Optional: Specify an array of resources to include in the backup. Resources might be shortcuts (for example, 'po' for 'pods') or fully-qualified. If unspecified, all resources are included.

- 3 Optional: Specify an array of resources to exclude from the backup. Resources might be shortcuts (for example, 'po' for 'pods') or fully-qualified.
- 4 Specify the name of the **backupStorageLocations** CR.
- 5 Map of {key,value} pairs of backup resources that have **all** the specified labels.
- 6 Map of {key,value} pairs of backup resources that have **one or more** of the specified labels.

3. Verify that the status of the **Backup** CR is **Completed**:

```
$ oc get backups.velero.io -n openshift-adp <backup> -o jsonpath='{.status.phase}'
```

5.17.3. Backing up persistent volumes with CSI snapshots

You back up persistent volumes with Container Storage Interface (CSI) snapshots by editing the **VolumeSnapshotClass** custom resource (CR) of the cloud storage before you create the **Backup** CR, see [CSI volume snapshots](#).

For more information, see [Creating a Backup CR](#).

5.17.3.1. Backing up persistent volumes with CSI snapshots

Prerequisites

- The cloud provider must support CSI snapshots.
- You must enable CSI in the **DataProtectionApplication** CR.

Procedure

- Add the **metadata.labels.velero.io/csi-volumesnapshot-class: "true"** key-value pair to the **VolumeSnapshotClass** CR:

Example configuration file

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: <volume_snapshot_class_name>
labels:
  velero.io/csi-volumesnapshot-class: "true" 1
annotations:
  snapshot.storage.kubernetes.io/is-default-class: true 2
driver: <csi_driver>
deletionPolicy: <deletion_policy_type> 3
```

- 1 Must be set to **true**.
- 2 If you are restoring this volume in another cluster with the same driver, make sure that you set the **snapshot.storage.kubernetes.io/is-default-class** parameter to **false** instead of setting it to **true**. Otherwise, the restore will partially fail.

3

OADP supports the **Retain** and **Delete** deletion policy types for CSI and Data Mover backup and restore.

Next steps

- You can now create a **Backup** CR.

5.17.4. Backing up applications with File System Backup: Kopia or Restic

You can use OADP to back up and restore Kubernetes volumes attached to pods from the file system of the volumes. This process is called File System Backup (FSB) or Pod Volume Backup (PVB). It is accomplished by using modules from the open source backup tools Restic or Kopia.

If your cloud provider does not support snapshots or if your applications are on NFS data volumes, you can create backups by using FSB.



NOTE

[Restic](#) is installed by the OADP Operator by default. If you prefer, you can install [Kopia](#) instead.

FSB integration with OADP provides a solution for backing up and restoring almost any type of Kubernetes volumes. This integration is an additional capability of OADP and is not a replacement for existing functionality.

You back up Kubernetes resources, internal images, and persistent volumes with Kopia or Restic by editing the **Backup** custom resource (CR).

You do not need to specify a snapshot location in the **DataProtectionApplication** CR.



NOTE

In OADP version 1.3 and later, you can use either Kopia or Restic for backing up applications.

For the Built-in DataMover, you must use Kopia.

In OADP version 1.2 and earlier, you can only use Restic for backing up applications.



IMPORTANT

FSB does not support backing up **hostPath** volumes. For more information, see [FSB limitations](#).



PODVOLUMERESTORE FAILS WITH A .../.SNAPSHOT: READ-ONLY FILE SYSTEM ERROR

The .../.**snapshot** directory is a snapshot copy directory, which is used by several NFS servers. This directory has read-only access by default, so Velero cannot restore to this directory.

Do not give Velero write access to the **.snapshot** directory, and disable client access to this directory.

Additional resources

- [Enable or disable client access to Snapshot copy directory by editing a share](#)
- [Prerequisites for backup and restore with FlashBlade](#)

5.17.4.1. Backing up applications with File System Backup

Prerequisites

- You must install the OpenShift API for Data Protection (OADP) Operator.
- You must not disable the default **nodeAgent** installation by setting **spec.configuration.nodeAgent.enable** to **false** in the **DataProtectionApplication** CR.
- You must select Kopia or Restic as the uploader by setting **spec.configuration.nodeAgent.uploaderType** to **kopia** or **restic** in the **DataProtectionApplication** CR.
- The **DataProtectionApplication** CR must be in a **Ready** state.

Procedure

- Create the **Backup** CR, as in the following example:

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: <backup>
  labels:
    velero.io/storage-location: default
  namespace: openshift-adp
spec:
  defaultVolumesToFsBackup: true 1
...
```

- 1** In OADP version 1.2 and later, add the **defaultVolumesToFsBackup: true** setting within the **spec** block. In OADP version 1.1, add **defaultVolumesToRestic: true**.

5.17.5. Creating backup hooks

When performing a backup, it is possible to specify one or more commands to execute in a container within a pod, based on the pod being backed up.

The commands can be configured to performed before any custom action processing (*Pre* hooks), or after all custom actions have been completed and any additional items specified by the custom action have been backed up (*Post* hooks).

You create backup hooks to run commands in a container in a pod by editing the **Backup** custom resource (CR).

Procedure

- Add a hook to the **spec.hooks** block of the **Backup** CR, as in the following example:

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: <backup>
  namespace: openshift-adp
spec:
  hooks:
    resources:
      - name: <hook_name>
        includedNamespaces:
          - <namespace> 1
        excludedNamespaces: 2
          - <namespace>
        includedResources: []
        - pods 3
        excludedResources: [] 4
        labelSelector: 5
          matchLabels:
            app: velero
            component: server
        pre: 6
          - exec:
              container: <container> 7
              command:
                - /bin/uname 8
                - -a
              onError: Fail 9
              timeout: 30s 10
        post: 11
    ...
```

- 1 Optional: You can specify namespaces to which the hook applies. If this value is not specified, the hook applies to all namespaces.
- 2 Optional: You can specify namespaces to which the hook does not apply.
- 3 Currently, pods are the only supported resource that hooks can apply to.
- 4 Optional: You can specify resources to which the hook does not apply.
- 5 Optional: This hook only applies to objects matching the label. If this value is not specified, the hook applies to all objects.

- 6 Array of hooks to run before the backup.
- 7 Optional: If the container is not specified, the command runs in the first container in the pod.
- 8 This is the entry point for the **init** container being added.
- 9 Allowed values for error handling are **Fail** and **Continue**. The default is **Fail**.
- 10 Optional: How long to wait for the commands to run. The default is **30s**.
- 11 This block defines an array of hooks to run after the backup, with the same parameters as the pre-backup hooks.

5.17.6. Scheduling backups using Schedule CR

The schedule operation allows you to create a backup of your data at a particular time, specified by a Cron expression.

You schedule backups by creating a **Schedule** custom resource (CR) instead of a **Backup** CR.



WARNING

Leave enough time in your backup schedule for a backup to finish before another backup is created.

For example, if a backup of a namespace typically takes 10 minutes, do not schedule backups more frequently than every 15 minutes.

Prerequisites

- You must install the OpenShift API for Data Protection (OADP) Operator.
- The **DataProtectionApplication** CR must be in a **Ready** state.

Procedure

1. Retrieve the **backupStorageLocations** CRs:

```
$ oc get backupStorageLocations -n openshift-adp
```

Example output

```

NAMESPACE   NAME           PHASE    LAST VALIDATED  AGE   DEFAULT
openshift-adp velero-sample-1 Available      11s           31m

```

2. Create a **Schedule** CR, as in the following example:

```
$ cat << EOF | oc apply -f -
```

```

apiVersion: velero.io/v1
kind: Schedule
metadata:
  name: <schedule>
  namespace: openshift-adp
spec:
  schedule: 0 7 * * * 1
  template:
    hooks: {}
    includedNamespaces:
      - <namespace> 2
    storageLocation: <velero-sample-1> 3
    defaultVolumesToFsBackup: true 4
    ttl: 720h0m0s
EOF

```

- 1 **cron** expression to schedule the backup, for example, **0 7 * * *** to perform a backup every day at 7:00.



NOTE

To schedule a backup at specific intervals, enter the **<duration_in_minutes>** in the following format:

```
schedule: "**/10 * * * *"
```

Enter the minutes value between quotation marks (" ").

- 2 Array of namespaces to back up.
- 3 Name of the **backupStorageLocations** CR.
- 4 Optional: In OADP version 1.2 and later, add the **defaultVolumesToFsBackup: true** key-value pair to your configuration when performing backups of volumes with Restic. In OADP version 1.1, add the **defaultVolumesToRestic: true** key-value pair when you back up volumes with Restic.

Verification

- Verify that the status of the **Schedule** CR is **Completed** after the scheduled backup runs:

```
$ oc get schedule -n openshift-adp <schedule> -o jsonpath='{.status.phase}'
```

5.17.7. Deleting backups

You can delete a backup by creating the **DeleteBackupRequest** custom resource (CR) or by running the **velero backup delete** command as explained in the following procedures.

The volume backup artifacts are deleted at different times depending on the backup method:

- Restic: The artifacts are deleted in the next full maintenance cycle, after the backup is deleted.

- Container Storage Interface (CSI): The artifacts are deleted immediately when the backup is deleted.
- Kopia: The artifacts are deleted after three full maintenance cycles of the Kopia repository, after the backup is deleted.

5.17.7.1. Deleting a backup by creating a DeleteBackupRequest CR

You can delete a backup by creating a **DeleteBackupRequest** custom resource (CR).

Prerequisites

- You have run a backup of your application.

Procedure

1. Create a **DeleteBackupRequest** CR manifest file:

```
apiVersion: velero.io/v1
kind: DeleteBackupRequest
metadata:
  name: deletebackuprequest
  namespace: openshift-adp
spec:
  backupName: <backup_name> 1
```

- 1 Specify the name of the backup.

2. Apply the **DeleteBackupRequest** CR to delete the backup:

```
$ oc apply -f <deletebackuprequest_cr_filename>
```

5.17.7.2. Deleting a backup by using the Velero CLI

You can delete a backup by using the Velero CLI.

Prerequisites

- You have run a backup of your application.
- You downloaded the Velero CLI and can access the Velero binary in your cluster.

Procedure

- To delete the backup, run the following Velero command:

```
$ velero backup delete <backup_name> -n openshift-adp 1
```

- 1 Specify the name of the backup.

5.17.7.3. About Kopia repository maintenance

There are two types of Kopia repository maintenance:

Quick maintenance

- Runs every hour to keep the number of index blobs (n) low. A high number of indexes negatively affects the performance of Kopia operations.
- Does not delete any metadata from the repository without ensuring that another copy of the same metadata exists.

Full maintenance

- Runs every 24 hours to perform garbage collection of repository contents that are no longer needed.
- **snapshot-gc**, a full maintenance task, finds all files and directory listings that are no longer accessible from snapshot manifests and marks them as deleted.
- A full maintenance is a resource-costly operation, as it requires scanning all directories in all snapshots that are active in the cluster.

5.17.7.3.1. Kopia maintenance in OADP

The **repo-maintain-job** jobs are executed in the namespace where OADP is installed, as shown in the following example:

pod/repo-maintain-job-173...2527-2nbls	0/1	Completed	0	168m
pod/repo-maintain-job-173....536-fl9tm	0/1	Completed	0	108m
pod/repo-maintain-job-173...2545-55ggx	0/1	Completed	0	48m

You can check the logs of the **repo-maintain-job** for more details about the cleanup and the removal of artifacts in the backup object storage. You can find a note, as shown in the following example, in the **repo-maintain-job** when the next full cycle maintenance is due:

```
not due for full maintenance cycle until 2024-00-00 18:29:4
```



IMPORTANT

Three successful executions of a full maintenance cycle are required for the objects to be deleted from the backup object storage. This means you can expect up to 72 hours for all the artifacts in the backup object storage to be deleted.

5.17.7.4. Deleting a backup repository

After you delete the backup, and after the Kopia repository maintenance cycles to delete the related artifacts are complete, the backup is no longer referenced by any metadata or manifest objects. You can then delete the **backuprepository** custom resource (CR) to complete the backup deletion process.

Prerequisites

- You have deleted the backup of your application.

- You have waited up to 72 hours after the backup is deleted. This time frame allows Kopia to run the repository maintenance cycles.

Procedure

1. To get the name of the backup repository CR for a backup, run the following command:

```
$ oc get backuprepositories.velero.io -n openshift-adp
```

2. To delete the backup repository CR, run the following command:

```
$ oc delete backuprepository <backup_repository_name> -n openshift-adp 1
```

- 1** Specify the name of the backup repository from the earlier step.

5.17.8. About Kopia

Kopia is a fast and secure open-source backup and restore tool that allows you to create encrypted snapshots of your data and save the snapshots to remote or cloud storage of your choice.

Kopia supports network and local storage locations, and many cloud or remote storage locations, including:

- Amazon S3 and any cloud storage that is compatible with S3
- Azure Blob Storage
- Google Cloud Storage platform

Kopia uses content-addressable storage for snapshots:

- Snapshots are always incremental; data that is already included in previous snapshots is not re-uploaded to the repository. A file is only uploaded to the repository again if it is modified.
- Stored data is deduplicated; if multiple copies of the same file exist, only one of them is stored.
- If files are moved or renamed, Kopia can recognize that they have the same content and does not upload them again.

5.17.8.1. OADP integration with Kopia

OADP 1.3 supports Kopia as the backup mechanism for pod volume backup in addition to Restic. You must choose one or the other at installation by setting the **uploaderType** field in the **DataProtectionApplication** custom resource (CR). The possible values are **restic** or **kopia**. If you do not specify an **uploaderType**, OADP 1.3 defaults to using Kopia as the backup mechanism. The data is written to and read from a unified repository.



IMPORTANT

Using the Kopia client to modify the Kopia backup repositories is not supported and can affect the integrity of Kopia backups. OADP does not support directly connecting to the Kopia repository and can offer support only on a best-effort basis.

The following example shows a **DataProtectionApplication** CR configured for using Kopia:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: dpa-sample
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
# ...
```

5.18. OADP RESTORING

5.18.1. Restoring applications

You restore application backups by creating a **Restore** custom resource (CR). See [Creating a Restore CR](#).

You can create restore hooks to run commands in a container in a pod by editing the **Restore** CR. See [Creating restore hooks](#).

5.18.1.1. Previewing resources before running backup and restore

OADP backs up application resources based on the type, namespace, or label. This means that you can view the resources after the backup is complete. Similarly, you can view the restored objects based on the namespace, persistent volume (PV), or label after a restore operation is complete. To preview the resources in advance, you can do a dry run of the backup and restore operations.

Prerequisites

- You have installed the OADP Operator.

Procedure

1. To preview the resources included in the backup before running the actual backup, run the following command:

```
$ velero backup create <backup-name> --snapshot-volumes false ❶
```

- ❶ Specify the value of **--snapshot-volumes** parameter as **false**.

2. To know more details about the backup resources, run the following command:

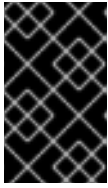
```
$ velero describe backup <backup_name> --details ❶
```

- ❶ Specify the name of the backup.

3. To preview the resources included in the restore before running the actual restore, run the following command:

```
$ velero restore create --from-backup <backup-name> 1
```

- 1 Specify the name of the backup created to review the backup resources.



IMPORTANT

The **velero restore create** command creates restore resources in the cluster. You must delete the resources created as part of the restore, after you review the resources.

4. To know more details about the restore resources, run the following command:

```
$ velero describe restore <restore_name> --details 1
```

- 1 Specify the name of the restore.

5.18.1.2. Creating a Restore CR

You restore a **Backup** custom resource (CR) by creating a **Restore** CR.

When you restore a stateful application that uses the **azurefile-csi** storage class, the restore operation remains in the **Finalizing** phase.

Prerequisites

- You must install the OpenShift API for Data Protection (OADP) Operator.
- The **DataProtectionApplication** CR must be in a **Ready** state.
- You must have a Velero **Backup** CR.
- The persistent volume (PV) capacity must match the requested size at backup time. Adjust the requested size if needed.

Procedure

1. Create a **Restore** CR, as in the following example:

```
apiVersion: velero.io/v1
kind: Restore
metadata:
  name: <restore>
  namespace: openshift-adp
spec:
  backupName: <backup> 1
  includedResources: [] 2
  excludedResources:
    - nodes
    - events
    - events.events.k8s.io
    - backups.velero.io
```



```
- restores.velero.io
- resticrepositories.velero.io
restorePVs: true 3
```

- 1 Name of the **Backup** CR.
- 2 Optional: Specify an array of resources to include in the restore process. Resources might be shortcuts (for example, **po** for **pods**) or fully-qualified. If unspecified, all resources are included.
- 3 Optional: The **restorePVs** parameter can be set to **false** to turn off restore of **PersistentVolumes** from **VolumeSnapshot** of Container Storage Interface (CSI) snapshots or from native snapshots when **VolumeSnapshotLocation** is configured.

2. Verify that the status of the **Restore** CR is **Completed** by entering the following command:

```
$ oc get restores.velero.io -n openshift-adp <restore> -o jsonpath='{.status.phase}'
```

3. Verify that the backup resources have been restored by entering the following command:

```
$ oc get all -n <namespace> 1
```

- 1 Namespace that you backed up.

4. If you restore **DeploymentConfig** with volumes or if you use post-restore hooks, run the **dc-post-restore.sh** cleanup script by entering the following command:

```
$ bash dc-restic-post-restore.sh -> dc-post-restore.sh
```



NOTE

During the restore process, the OADP Velero plug-ins scale down the **DeploymentConfig** objects and restore the pods as standalone pods. This is done to prevent the cluster from deleting the restored **DeploymentConfig** pods immediately on restore and to allow the restore and post-restore hooks to complete their actions on the restored pods. The cleanup script shown below removes these disconnected pods and scales any **DeploymentConfig** objects back up to the appropriate number of replicas.

dc-restic-post-restore.sh → dc-post-restore.sh cleanup script

```
#!/bin/bash
set -e

# if sha256sum exists, use it to check the integrity of the file
if command -v sha256sum >/dev/null 2>&1; then
    CHECKSUM_CMD="sha256sum"
else
    CHECKSUM_CMD="shasum -a 256"
fi
```

```

label_name () {
    if [ "${#1}" -le "63" ]; then
        echo $1
        return
    fi
    sha=$(echo -n $1|$CHECKSUM_CMD)
    echo "${1:0:57}${sha:0:6}"
}

if [[ $# -ne 1 ]]; then
    echo "usage: ${BASH_SOURCE} restore-name"
    exit 1
fi

echo "restore: $1"

label=$(label_name $1)
echo "label: $label"

echo Deleting disconnected restore pods
oc delete pods --all-namespaces -l oadp.openshift.io/disconnected-from-dc=$label

for dc in $(oc get dc --all-namespaces -l oadp.openshift.io/replicas-modified=$label -o
jsonpath='{range .items[*]}{.metadata.namespace}{","}{.metadata.name}{","}'
{.metadata.annotations.oadp\openshift\io/original-replicas}{","}'
{.metadata.annotations.oadp\openshift\io/original-paused}{"\n"}')
do
    IFS=' ' read -ra dc_arr <<< "$dc"
    if [ ${#dc_arr[0]} -gt 0 ]; then
        echo Found deployment ${dc_arr[0]}/${dc_arr[1]}, setting replicas: ${dc_arr[2]}, paused:
        ${dc_arr[3]}
        cat <<EOF | oc patch dc -n ${dc_arr[0]} ${dc_arr[1]} --patch-file /dev/stdin
spec:
  replicas: ${dc_arr[2]}
  paused: ${dc_arr[3]}
EOF
    fi
done

```

5.18.1.3. Creating restore hooks

You create restore hooks to run commands in a container in a pod by editing the **Restore** custom resource (CR).

You can create two types of restore hooks:

- An **init** hook adds an init container to a pod to perform setup tasks before the application container starts.
If you restore a Restic backup, the **restic-wait** init container is added before the restore hook init container.
- An **exec** hook runs commands or scripts in a container of a restored pod.

Procedure

- Add a hook to the **spec.hooks** block of the **Restore** CR, as in the following example:

```

apiVersion: velero.io/v1
kind: Restore
metadata:
  name: <restore>
  namespace: openshift-adp
spec:
  hooks:
    resources:
      - name: <hook_name>
        includedNamespaces:
          - <namespace> ❶
        excludedNamespaces:
          - <namespace>
        includedResources:
          - pods ❷
        excludedResources: []
        labelSelector: ❸
          matchLabels:
            app: velero
            component: server
        postHooks:
          - init:
              initContainers:
                - name: restore-hook-init
                  image: alpine:latest
                  volumeMounts:
                    - mountPath: /restores/pvc1-vm
                      name: pvc1-vm
                  command:
                    - /bin/ash
                    - -c
              timeout: ❹
          - exec:
              container: <container> ❺
              command:
                - /bin/bash ❻
                - -c
                - "psql < /backup/backup.sql"
              waitTimeout: 5m ❼
              execTimeout: 1m ❽
              onError: Continue ❾

```

- ❶ Optional: Array of namespaces to which the hook applies. If this value is not specified, the hook applies to all namespaces.
- ❷ Currently, pods are the only supported resource that hooks can apply to.
- ❸ Optional: This hook only applies to objects matching the label selector.
- ❹ Optional: Timeout specifies the maximum length of time Velero waits for **initContainers** to complete.

- 5 Optional: If the container is not specified, the command runs in the first container in the pod.
- 6 This is the entrypoint for the init container being added.
- 7 Optional: How long to wait for a container to become ready. This should be long enough for the container to start and for any preceding hooks in the same container to complete. If not set, the restore process waits indefinitely.
- 8 Optional: How long to wait for the commands to run. The default is **30s**.
- 9 Allowed values for error handling are **Fail** and **Continue**:
 - **Continue**: Only command failures are logged.
 - **Fail**: No more restore hooks run in any container in any pod. The status of the **Restore** CR will be **PartiallyFailed**.

IMPORTANT

During a File System Backup (FSB) restore operation, a **Deployment** resource referencing an **ImageStream** is not restored properly. The restored pod that runs the FSB, and the **postHook** is terminated prematurely.

This happens because, during the restore operation, OpenShift controller updates the **spec.template.spec.containers[0].image** field in the **Deployment** resource with an updated **ImageStreamTag** hash. The update triggers the rollout of a new pod, terminating the pod on which **velero** runs the FSB and the post restore hook. For more information about image stream trigger, see "Triggering updates on image stream changes".

The workaround for this behavior is a two-step restore process:

1. First, perform a restore excluding the **Deployment** resources, for example:

```
$ velero restore create <RESTORE_NAME> \
  --from-backup <BACKUP_NAME> \
  --exclude-resources=deployment.apps
```

2. After the first restore is successful, perform a second restore by including these resources, for example:

```
$ velero restore create <RESTORE_NAME> \
  --from-backup <BACKUP_NAME> \
  --include-resources=deployment.apps
```

Additional resources

- [Triggering updates on image stream changes](#)

5.19. OADP SELF-SERVICE

5.19.1. OADP Self-Service

Use OADP Self-Service to enable namespace administrators to back up and restore their applications without cluster admin privileges. This helps you delegate backup operations while maintaining administrative control.

5.19.1.1. About OADP Self-Service

From OADP 1.5.0 onward, you do not need the **cluster-admin** role to perform the backup and restore operations. You can use OADP with the namespace **admin** role. The namespace **admin** role has administrator access only to the namespace the user is assigned to.

You can use the Self-Service feature only after the cluster administrator installs the OADP Operator and provides the necessary permissions.

The OADP Self-Service feature provides secure self-service data protection capabilities for users without **cluster-admin** privileges while maintaining proper access controls.

The OADP cluster administrator creates a user with the namespace **admin** role and provides the necessary Role Based Access Controls (RBAC) to the user to perform OADP Self-Service actions. As this user has limited access compared to the **cluster-admin** role, this user is referred to as a namespace admin user.

As a namespace admin user, you can back up and restore applications deployed in your authorized namespace on the cluster.

OADP Self-Service offers the following benefits:

- As a cluster administrator:
 - You allow namespace-scoped backup and restore operations to a namespace admin user. This means, a namespace admin user cannot access a namespace that they are not authorized to.
 - You keep administrator control over non-administrator operations through **DataProtectionApplication** configuration and policies.
- As a namespace admin user:
 - You can create backup and restore custom resources for your authorized namespace.
 - You can create dedicated backup storage locations in your authorized namespace.
 - You have secure access to backup logs and status information.

5.19.1.2. What namespace-scoped backup and restore means

OADP Self-Service ensures that namespace admin users can only operate within their authorized namespace. For example, if you do not have access to a namespace, as a namespace admin user, you cannot back up that namespace.

A namespace admin user cannot access backup and restore data of other users.

The cluster administrator enforces the access control through custom resources (CRs) that securely manage the backup and restore operations.

Additionally, the cluster administrator can control the allowed options within the CRs, restricting certain operations for added security by using **spec** enforcements in the **DataProtectionApplication** (DPA) CR.

Namespace **admin** users can perform the following Self-Service operations:

- Create and manage backups of their authorized namespaces.
- Restore data to their authorized namespaces.
- Configure their own backup storage locations.
- Check backup and restore status.
- Request retrieval of relevant logs.

Additional resources

- [Configuring an htpasswd identity provider](#)

5.19.1.3. OADP Self-Service custom resources

Use OADP Self-Service custom resources to control backup, restore, storage location, and download operations for namespace-scoped applications. This provides namespace administrators with self-service data protection tools.

The OADP Self-Service feature has the following new custom resources (CRs) to perform the backup and restore operations for a namespace admin user:

Table 5.6. Custom resources

CR	Description
NonAdminController (NAC)	Controls and orchestrates the Self-Service operations.
NonAdminBackup (NAB)	Manages namespace-scoped backup operations.
NonAdminRestore (NAR)	Manages namespace-scoped restore operations.
NonAdminBackupStorageLocation (NABSL)	Defines user-specific backup storage location.
NonAdminDownloadRequest (NADR)	Manages namespace-scoped download request operations.

5.19.1.4. How OADP Self-Service works

Review how OADP Self-Service processes backup requests through the **NonAdminController** (NAC) custom resource, which validates namespace administrator requests and creates corresponding **Velero** backup objects.

The diagram describes the following workflow:

1. A namespace admin user creates a **NonAdminBackup** (NAB) custom resource (CR) request.
2. The **NonAdminController** (NAC) CR receives the NAB CR request.
3. The NAC validates the request and updates the NAB CR about the request.
4. The NAC creates the **Velero** backup object.
5. The NAC monitors the **Velero** backup object and cascades the status back to the NAB CR.

Figure 5.1. How OADP Self-Service works



5.19.1.5. OADP Self-Service prerequisites

Configure your cluster environment to enable OADP Self-Service backup and restore operations by meeting the following prerequisites. This helps namespace administrators perform data protection tasks in their assigned namespaces.

- The cluster administrator has configured the OADP **DataProtectionApplication** (DPA) CR to enable Self-Service.
- The cluster administrator has completed the following tasks:
 - Created a namespace **admin** user account.
 - Created a namespace for the namespace **admin** user.
 - Assigned appropriate privileges for the namespace admin user's namespace. This ensures that the namespace admin user is authorized to access and perform backup and restore operations in their assigned namespace.
- Optionally, the cluster administrator can create a **NonAdminBackupStorageLocation** (NABSL) CR for the namespace **admin** user.

5.19.1.6. OADP Self-Service namespace permissions

Assign namespace permissions to namespace administrators to create and manage backup, restore, and storage location resources in their assigned namespaces. This grants namespace administrators the required access for Self-Service data protection operations.

As a cluster administrator, ensure that a namespace admin user has editor roles assigned for the following list of objects in their namespace.

- **nonadminbackups.oadp.openshift.io**

- `nonadminbackupstoragelocations.oadp.openshift.io`
- `nonadminrestores.oadp.openshift.io`
- `nonadmindownloadrequests.oadp.openshift.io`

For more details on the namespace **admin** role, see [Default cluster roles](#).

A cluster administrator can also define their own specifications so that users can have rights similar to **project** or namespace **admin** roles.

5.19.1.6.1. Example RBAC YAML for backup operation

See the following role-based access control (RBAC) YAML file example with namespace permissions for a namespace **admin** user to perform a backup operation.

Example RBAC manifest

```
...
- apiGroups:
  - oadp.openshift.io
  resources:
  - nonadminbackups
  - nonadminrestores
  - nonadminbackupstoragelocations
  - nonadmindownloadrequests
  verbs:
  - create
  - delete
  - get
  - list
  - patch
  - update
  - watch
- apiGroups:
  - oadp.openshift.io
  resources:
  - nonadminbackups/status
  - nonadminrestores/status
  verbs:
  - get
```

5.19.1.7. OADP Self-Service limitations

Review the limitations and unsupported features of OADP Self-Service to understand which operations are restricted for namespace administrators. This helps you plan appropriate backup and restore strategies within the supported functionality.

The following features are not supported by OADP Self-Service:

- Cross cluster backup and restore, or migrations are not supported. These OADP operations are supported for the cluster administrator.

- A namespace **admin** user cannot create a **VolumeSnapshotLocation** (VSL) CR. The cluster administrator creates and configures the VSL in the **DataProtectionApplication** (DPA) CR for a namespace **admin** user.
- The **ResourceModifiers** CR and volume policies are not supported for a namespace **admin** user.
- A namespace **admin** user can request backup or restore logs by using the **NonAdminDownloadRequest** CR, only if the backup or restore is created by a user by using the **NonAdminBackupStorageLocation** CR.
If the backup or restore CRs are created by using the cluster-wide default backup storage location, a namespace **admin** user cannot request the backup or restore logs.
- To ensure secure backup and restore, OADP Self-Service automatically excludes the following CRs from being backed up or restored:
 - **NonAdminBackup**
 - **NonAdminRestore**
 - **NonAdminBackupStorageLocation**
 - **SecurityContextConstraints**
 - **ClusterRole**
 - **ClusterRoleBinding**
 - **CustomResourceDefinition**
 - **PriorityClasses**
 - **VirtualMachineClusterInstanceTypes**
 - **VirtualMachineClusterPreferences**

5.19.1.8. OADP Self-Service backup and restore phases

Review the status phases of **NonAdminBackup** (NAB) and **NonAdminRestore** (NAR) custom resources to track the progress and state of backup and restore operations. This helps you monitor and troubleshoot Self-Service backup and restore requests.

The phase of the CRs only progress forward. Once a phase transitions to the next phase, it cannot revert to a previous phase.

Table 5.7. Phases

Value	Description
New	A creation request of the NAB or NAR CR is accepted by the NAC, but it has not yet been validated by the NAC.

BackingOff	NAB or NAR CR is invalidated by the NAC CR because of an invalid spec of the NAB or NAR CR. The namespace admin user can update the NAB or NAR spec to comply with the policies set by the administrator. After the namespace admin user edits the CRs, the NAC reconciles the CR again.
Created	NAB or NAR CR is validated by the NAC, and the Velero backup or restore object is created.
Deletion	NAB or NAR CR is marked for deletion. The NAC deletes the corresponding Velero backup or restore object. When the Velero object is deleted, the NAB or NAR CR is also deleted.

5.19.1.9. About NonAdminBackupStorageLocation CR

Review the **NonAdminBackupStorageLocation** (NABSL) custom resource (CR) workflows to understand how namespace administrators define backup storage locations through administrator creation, approval, or automatic processes. This helps you choose the appropriate workflow based on security requirements.

To ensure that the NABSL CR is created and used securely, use cluster administrator controls. The cluster administrator manages the NABSL CR to comply with company policies, and compliance requirements.

You can create a NABSL CR by using one of the following workflows:

- **Administrator creation workflow:** In this workflow, the cluster administrator creates the NABSL CR for the namespace admin user. The namespace admin user then references the NABSL in the **NonAdminBackup** CR.
- **Administrator approval workflow:** The cluster administrator must explicitly enable this opt-in feature in the DPA by setting the **nonAdmin.requireApprovalForBSL** field to **true**. The cluster administrator approval process works as follows:
 - a. A namespace admin user creates a NABSL CR. Because the administrator has enforced an approval process in the DPA, it triggers the creation of a **NonAdminBackupStorageLocationRequest** CR in the **openshift-adp** namespace.
 - b. The cluster administrator reviews the request and either approves or rejects the request.
 - If approved, a **Velero BackupStorageLocation** (BSL) is created in the **openshift-adp** namespace, and the NABSL CR status is updated to reflect the approval.
 - If rejected, the status of the NABSL CR is updated to reflect the rejection.
 - c. The cluster administrator can also revoke a previously approved NABSL CR. The **approve** field is set back to **pending** or **reject**. This results in the deletion of the **Velero** BSL, and the namespace admin user is notified of the rejection.
- **Automatic approval workflow:** In this workflow, the cluster administrator does not enforce an approval process for the NABSL CR by setting the **nonAdmin.requireApprovalForBSL** field in

the DPA to **false**. The default value of this field is **false**. Not setting the field results in an automatic approval of the NABSL. Therefore, the namespace admin user can create the NABSL CR from their authorized namespace.



IMPORTANT

For security purposes, use either the administrator creation or the administrator approval workflow. The automatic approval workflow is less secure as it does not require administrator review.

5.19.2. OADP Self-Service cluster admin use cases

Configure and manage OADP Self-Service by enabling the feature, reviewing backup storage location requests, and enforcing policy templates. This helps you provide Self-Service backup capabilities while maintaining administrative control.

5.19.2.1. Enabling and disabling OADP Self-Service

Enable or disable the OADP Self-Service feature to allow namespace administrators to manage their own backup and restore operations without cluster admin privileges. This helps you delegate backup responsibilities while maintaining administrative control.



NOTE

You can install only one instance of the **NonAdminController** (NAC) CR in the cluster. If you install multiple instances of the NAC CR, you get the following error:

message: only a single instance of Non-Admin Controller can be installed across the entire cluster. Non-Admin controller is already configured and installed in openshift-adp namespace.

Prerequisites

- You are logged in to the cluster with the **cluster-admin** role.
- You have installed the OADP Operator.
- You have configured the DPA.

Procedure

- To enable OADP Self-Service, edit the DPA CR to configure the **nonAdmin.enable** section. See the following example configuration:

Example DataProtectionApplication CR

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: oadp-backup
  namespace: openshift-adp
spec:
  configuration:
```

```

nodeAgent:
  enable: true
  uploaderType: kopia
velero:
  defaultPlugins:
    - aws
    - openshift
    - csi
  defaultSnapshotMoveData: true
nonAdmin:
  enable: true
backupLocations:
  - velero:
      config:
        profile: "default"
        region: noobaa
        s3Url: https://s3.openshift-storage.svc
        s3ForcePathStyle: "true"
        insecureSkipTLSVerify: "true"
      provider: aws
      default: true
      credential:
        key: cloud
        name: <cloud_credentials>
      objectStorage:
        bucket: <bucket_name>
        prefix: oadp

```

where:

nonAdmin

Specifies the section in the **spec** section of the DPA to enable or disable the Self-Service feature.

enable

Specifies whether to enable the Self-Service feature. Set to **true** to enable the feature. Set to **false** to disable the feature.

Verification

- To verify that the **NonAdminController** (NAC) pod is running in the OADP namespace, run the following command:

```
$ oc get pod -n openshift-adp -l control-plane=non-admin-controller
```

Example output

```

NAME                                READY STATUS  RESTARTS  AGE
non-admin-controller-5d....f5-p..9p  1/1   Running    0         99m

```

5.19.2.2. Enabling NonAdminBackupStorageLocation administrator approval workflow

Enable the administrator approval workflow for **NonAdminBackupStorageLocation** custom resource to review backup storage location requests from namespace administrators before they are applied. This helps you maintain control over backup storage configurations.

Prerequisites

- You are logged in to the cluster with the **cluster-admin** role.
- You have installed the OADP Operator.
- You have enabled OADP Self-Service in the **DataProtectionApplication** CR.

Procedure

- To enable the NABSL administrator approval workflow, edit the DPA CR by using the following example configuration:

Example **DataProtectionApplication** CR

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: oadp-backup
  namespace: openshift-adp
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
  velero:
    defaultPlugins:
      - aws
      - openshift
      - csi
    noDefaultBackupLocation: true
  nonAdmin:
    enable: true
    requireApprovalForBSL: true
```

where:

noDefaultBackupLocation

Specifies that there is no default backup storage location configured in the DPA CR. Set to **true** to enable the namespace admin user to create a NABSL CR and send the CR request for approval.

requireApprovalForBSL

Specifies whether the NABSL administrator approval workflow is enabled. Set to **true** to enable the approval workflow.

5.19.2.3. Approving a **NonAdminBackupStorageLocation** request

Approve **NonAdminBackupStorageLocation** (NABSL) custom resource requests from namespace administrators to grant access to their specified backup storage locations. This enables self-service backup and restore operations for namespace resources.

Prerequisites

- You are logged in to the cluster with the **cluster-admin** role.
- You have installed the OADP Operator.
- You have enabled OADP Self-Service in the **DataProtectionApplication** (DPA) CR.
- You have enabled the NABSL CR approval workflow in the DPA.

Procedure

1. To see the NABSL CR requests that are in queue for administrator approval, run the following command:

```
$ oc -n openshift-adp get NonAdminBackupStorageLocationRequests
```

Example output

NAME	REQUEST-PHASE	REQUEST-NAMESPACE	REQUEST-NAME	AGE
non-admin-bsl-test-.....175	Approved	non-admin-bsl-test	incorrect-bucket-nabsl	4m57s
non-admin-bsl-test-.....196	Approved	non-admin-bsl-test	perfect-nabsl	5m26s
non-admin-bsl-test-s....e1a	Rejected	non-admin-bsl-test	suspicious-sample	2m56s
non-admin-bsl-test-.....5e0	Pending	non-admin-bsl-test	waitingapproval-nabsl	4m20s

2. To approve the NABSL CR request, set the **approvalDecision** field to **approve** by running the following command:

```
$ oc patch nabslrequest <nabsl_name> -n openshift-adp --type=merge -p '{"spec": {"approvalDecision": "approve"}}'
```

Replace **<nabsl_name>** with the name of the **NonAdminBackupStorageLocationRequest** CR.

Verification

- Verify that the Velero backup storage location is created and the phase is **Available** by running the following command:

```
$ oc get velero.io.backupstoragelocation
```

Example output

NAME	PHASE	LAST VALIDATED	AGE	DEFAULT
test-nac-test-bsl-cd...930	Available	62s	62s	

5.19.2.4. Rejecting a NonAdminBackupStorageLocation request

Reject **NonAdminBackupStorageLocation** (NABSL) custom resource (CR) requests from namespace administrators to deny access to backup storage locations that do not meet requirements. This helps you maintain security and compliance standards.

Prerequisites

- You are logged in to the cluster with the **cluster-admin** role.
- You have installed the OADP Operator.
- You have enabled OADP Self-Service in the **DataProtectionApplication** (DPA) CR.
- You have enabled the NABSL CR approval workflow in the DPA.

Procedure

1. To see the NABSL CR requests that are in queue for administrator approval, run the following command:

```
$ oc -n openshift-adp get NonAdminBackupStorageLocationRequests
```

Example output

```
$ oc get nabslrequest
NAME                                REQUEST-PHASE  REQUEST-NAMESPACE  REQUEST-NAME
AGE
non-admin-bsl-test-.....175  Approved      non-admin-bsl-test  incorrect-bucket-nabsl
4m57s
non-admin-bsl-test-.....196  Approved      non-admin-bsl-test  perfect-nabsl          5m26s
non-admin-bsl-test-s....e1a  Rejected      non-admin-bsl-test  suspicious-sample
2m56s
non-admin-bsl-test-.....5e0  Pending       non-admin-bsl-test  waitingapproval-nabsl
4m20s
```

2. To reject the NABSL CR request, set the **approvalDecision** field to **reject** by running the following command:

```
$ oc patch nabslrequest <nabsl_name> -n openshift-adp --type=merge -p '{"spec":
{"approvalDecision": "reject"}}'
```

Replace **<nabsl_name>** with the name of the **NonAdminBackupStorageLocationRequest** CR.

5.19.2.5. OADP Self-Service administrator DPA spec enforcement

Enforce policy templates in the **DataProtectionApplication** (DPA) custom resource (CR) to control **NonAdminBackup**, **NonAdminRestore**, and **NonAdminBackupStorageLocation** custom resources created by namespace administrators. This helps you maintain compliance standards.

The cluster administrator can enforce a company, or a compliance policy by using the following fields in the **DataProtectionApplication** (DPA) CR:

enforceBSLSpec

To enforce a policy on the **NonAdminBackupStorageLocation** CR.

enforceBackupSpec

To enforce a policy on the **NonAdminBackup** CR.

enforceRestoreSpec

To enforce a policy on the **NonAdminRestore** CR.

By using the enforceable fields, administrators can ensure that the NABSL, NAB, and NAR CRs created by a namespace admin user, comply with the administrator defined policy.

5.19.2.6. Self-Service administrator spec enforcement for NABSL

Enforce specific fields in **NonAdminBackupStorageLocation** (NABSL) custom resource (CR) to control storage bucket, credentials, configuration, access mode, and validation settings used by namespace administrators. This helps you maintain organizational policies.

You can enforce the following fields for a NABSL:

- **objectStorage**
- **credential**
- **config**
- **accessMode**
- **validationFrequency**

For example, if you want to enforce a namespace admin user to use a specific storage bucket, you can set up the **DataProtectionApplication** (DPA) CR as following:

Example DataProtectionApplication CR

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  nonAdmin:
    enable: true
    enforceBSLSpec:
      config:
        checksumAlgorithm: ""
        profile: default
        region: us-west-2
      objectStorage:
        bucket: my-company-bucket
        prefix: velero
        provider: aws
```

where:

enforceBSLSpec

Specifies the section to enforce policies for the **NonAdminBackupStorageLocation** CR.

config

Specifies the configuration to enforce for the NABSL. In this example, it enforces the use of an AWS S3 bucket in the **us-west-2** region.

objectStorage

Specifies the object storage settings to use a company bucket named **my-company-bucket**.

When a namespace admin user creates a NABSL, they must follow the template set up in the DPA. Otherwise, the **status.phase** field on the NABSL CR is set to **BackingOff** and the NABSL fails to create.

5.19.2.7. Self-Service administrator spec enforcement for NAB

Enforce specific fields in **NonAdminBackup** (NAB) custom resource (CR) to control timeout settings, resource policies, label selectors, snapshot configurations, and time-to-live values used by namespace administrators. This helps you maintain backup standards.

You can enforce the following fields for a NAB CR:

- **csiSnapshotTimeout**
- **itemOperationTimeout**
- **resourcePolicy**
- **includedResources**
- **excludedResources**
- **orderedResources**
- **includeClusterResources**
- **excludedClusterScopedResources**
- **excludedNamespaceScopedResources**
- **includedNamespaceScopedResources**
- **labelSelector**
- **orLabelSelectors**
- **snapshotVolumes**
- **ttl**
- **snapshotMoveData**
- **uploaderConfig.parallelFilesUpload**

If you want to enforce a **ttl** value and a Data Mover backup for a namespace admin user, you can set up the **DataProtectionApplication** (DPA) CR as shown in the following example:

Example DataProtectionApplication CR

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  nonAdmin:
    enable: true
    enforceBackupSpec:
      snapshotMoveData: true
      ttl: 158h0m0s

```

where:

enforceBackupSpec

Specifies the section to enforce policies for the **NonAdminBackup** CR.

snapshotMoveData

Specifies whether to enforce Data Mover. Set to **true** to enforce Data Mover backups.

ttl

Specifies the time-to-live value to enforce for backups. In this example, it is set to **158h0m0s**.

When a namespace admin user creates a NAB CR, they must follow the template set up in the DPA. Otherwise, the **status.phase** field on the NAB CR is set to **BackingOff** and the NAB CR fails to create.

5.19.2.8. Self-Service administrator spec enforcement for NAR

Enforce specific fields in **NonAdminRestore** (NAR) custom resource (CR) to control timeout settings, resource policies, label selectors, persistent volume restoration, and node port configurations used by namespace administrators. This helps you maintain restore standards.

You can enforce the following fields for a NAR CR:

- **itemOperationTimeout**
- **uploaderConfig**
- **includedResources**
- **excludedResources**
- **restoreStatus**
- **includeClusterResources**
- **labelSelector**
- **orLabelSelectors**
- **restorePVs**
- **preserveNodePorts**

5.19.3. OADP Self-Service namespace admin use cases

Use OADP Self-Service as a namespace administrator to create backup storage locations, perform backup and restore operations, and review operation logs for your authorized namespaces. This helps you to manage data protection independently without cluster admin access.

5.19.3.1. Creating a NonAdminBackupStorageLocation CR

Create a **NonAdminBackupStorageLocation** (NABSL) custom resource (CR) to define backup storage locations in your authorized namespace. With this feature, you can store backups in a cloud storage that meets your application requirements.

Prerequisites

- You are logged in to the cluster as a namespace admin user.
- The cluster administrator has installed the OADP Operator.
- The cluster administrator has configured the **DataProtectionApplication** (DPA) CR to enable OADP Self-Service.
- The cluster administrator has created a namespace for you and has authorized you to operate from that namespace.

Procedure

1. Create a **Secret** CR by using the cloud credentials file content for your cloud provider. Run the following command:

```
$ oc create secret generic cloud-credentials -n test-nac-ns --from-file <cloud_key_name>=<cloud_credentials_file>
```

where:

<cloud_key_name>

Specifies the cloud provider key name. In this example, the **Secret** name is **cloud-credentials** and the authorized namespace name is **test-nac-ns**.

<cloud_credentials_file>

Specifies the cloud credentials file name.

2. To create a **NonAdminBackupStorageLocation** CR, create a YAML manifest file with the following configuration:

Example NonAdminBackupStorageLocation CR

```
apiVersion: oadp.openshift.io/v1alpha1
kind: NonAdminBackupStorageLocation
metadata:
  name: test-nabsl
  namespace: test-nac-ns
spec:
  backupStorageLocationSpec:
    config:
      profile: default
      region: <region_name>
    credential:
```

```

    key: cloud
    name: cloud-credentials
  objectStorage:
    bucket: <bucket_name>
    prefix: velero
    provider: aws

```

where:

namespace

Specifies the namespace you are authorized to operate from. For example, **test-nac-ns**.

<region_name>

Specifies the region name for your cloud provider.

<bucket_name>

Specifies the bucket name for storing backups.

3. To apply the NABSL CR configuration, run the following command:

```
$ oc apply -f <nabsl_cr_filename>
```

Replace **<nabsl_cr_filename>** with the file name containing the NABSL CR configuration.

Verification

1. To verify that the NABSL CR is in the **New** phase and is pending administrator approval, run the following command:

```
$ oc get nabsl test-nabsl -o yaml
```

Example output

```

apiVersion: oadp.openshift.io/v1alpha1
kind: NonAdminBackupStorageLocation
...
status:
  conditions:
    - lastTransitionTime: "2025-02-26T09:07:15Z"
      message: NonAdminBackupStorageLocation spec validation successful
      reason: BslSpecValidation
      status: "True"
      type: Accepted
    - lastTransitionTime: "2025-02-26T09:07:15Z"
      message: NonAdminBackupStorageLocationRequest approval pending
      reason: BslSpecApprovalPending
      status: "False"
      type: ClusterAdminApproved
  phase: New
  veleroBackupStorageLocation:
    nacuid: test-nac-test-bsl-c...d4389a1930
    name: test-nac-test-bsl-cd....1930
    namespace: openshift-adp

```

where:

message

Contains the **NonAdminBackupStorageLocationRequest approval pending** message.

phase

Specifies the status of the phase. In this example, the phase is **New**.

2. After the cluster administrator approves the **NonAdminBackupStorageLocationRequest** CR request, verify that the NABSL CR is successfully created by running the following command:

```
$ oc get nabsl test-nabsl -o yaml
```

Example output

```
apiVersion: oadp.openshift.io/v1alpha1
kind: NonAdminBackupStorageLocation
metadata:
  creationTimestamp: "2025-02-19T09:30:34Z"
  finalizers:
    - nonadminbackupstoragelocation.oadp.openshift.io/finalizer
  generation: 1
  name: test-nabsl
  namespace: test-nac-ns
  resourceVersion: "159973"
  uid: 4a..80-3260-4ef9-a3..5a-00...d1922
spec:
  backupStorageLocationSpec:
    credential:
      key: cloud
      name: cloud-credentials
    objectStorage:
      bucket: oadp...51rrdqj
      prefix: velero
      provider: aws
status:
  conditions:
    - lastTransitionTime: "2025-02-19T09:30:34Z"
      message: NonAdminBackupStorageLocation spec validation successful
      reason: BslSpecValidation
      status: "True"
      type: Accepted
    - lastTransitionTime: "2025-02-19T09:30:34Z"
      message: Secret successfully created in the OADP namespace
      reason: SecretCreated
      status: "True"
      type: SecretSynced
    - lastTransitionTime: "2025-02-19T09:30:34Z"
      message: BackupStorageLocation successfully created in the OADP namespace
      reason: BackupStorageLocationCreated
      status: "True"
      type: BackupStorageLocationSynced
  phase: Created
  veleroBackupStorageLocation:
    nacuuid: test-nac-..f933a-4ec1-4f6a-8099-ee...b8b26
```

```

name: test-nac-test-nabsl-36...11ab8b26
namespace: openshift-adp
status:
  lastSyncedTime: "2025-02-19T11:47:10Z"
  lastValidationTime: "2025-02-19T11:47:31Z"
  phase: Available

```

where:

message: NonAdminBackupStorageLocation spec validation successful

Specifies that the NABSL **spec** is validated and approved by the cluster administrator.

message: Secret successfully created in the OADP namespace

Specifies that the **secret** object is successfully created in the **openshift-adp** namespace.

message: BackupStorageLocation successfully created in the OADP namespace

Specifies that the associated **Velero BackupStorageLocation** is successfully created in the **openshift-adp** namespace.

nacuuid

Specifies the NAC that is orchestrating the NABSL CR.

name

Specifies the name of the associated **Velero** backup storage location object.

phase: Available

Specifies that the NABSL is ready for use.

5.19.3.2. Creating a NonAdminBackup CR

Create a **NonAdminBackup** (NAB) custom resource (CR) to back up application resources in your authorized namespace. This helps you to protect your application data and configuration without requiring cluster administrator privileges.

After you create a NAB CR, the CR undergoes the following phases:

- The initial phase for the CR is **New**.
- The CR creation request goes to the **NonAdminController** (NAC) for reconciliation and validation.
- Upon successful validation and creation of the **Velero** backup object, the **status.phase** field of the NAB CR is updated to the next phase, which is, **Created**.

Review the following important points when creating a NAB CR:

- The **NonAdminBackup** CR creates the **Velero** backup object securely so that other namespace admin users cannot access the CR.
- As a namespace admin user, you can only specify your authorized namespace in the NAB CR. You get an error when you specify a namespace you are not authorized to use.

Prerequisites

- You are logged in to the cluster as a namespace admin user.

- The cluster administrator has installed the OADP Operator.
- The cluster administrator has configured the **DataProtectionApplication** (DPA) CR to enable OADP Self-Service.
- The cluster administrator has created a namespace for you and has authorized you to operate from that namespace.
- Optional: You can create and use a **NonAdminBackupStorageLocation** (NABSL) CR to store the backup data. If you do not use a NABSL CR, then the backup is stored in the default backup storage location configured in the DPA.

Procedure

1. To create a **NonAdminBackup** CR, create a YAML manifest file with the following configuration:

Example NonAdminBackup CR

```
apiVersion: oadp.openshift.io/v1alpha1
kind: NonAdminBackup
metadata:
  name: test-nab
spec:
  backupSpec:
    defaultVolumesToFsBackup: true
    snapshotMoveData: false
    storageLocation: test-bsl
```

where:

name

Specifies a name for the NAB CR. For example, **test-nab**.

defaultVolumesToFsBackup

Specifies whether to use File System Backup (FSB). Set to **true** to use FSB.

snapshotMoveData

Specifies whether to back up data volumes by using the Data Mover. Set to **true** to use Data Mover. This example uses FSB for backup.

storageLocation

Specifies a NABSL CR as a storage location. If you do not set a **storageLocation**, then the default backup storage location configured in the DPA is used.

2. To apply the NAB CR configuration, run the following command:

```
$ oc apply -f <nab_cr_filename>
```

Replace **<nab_cr_filename>** with the file name containing the NAB CR configuration.

Verification

- To verify that the NAB CR is successfully created, run the following command:

■

```
$ oc get nab test-nab -o yaml
```

Example output

```
apiVersion: oadp.openshift.io/v1alpha1
kind: NonAdminBackup
metadata:
  creationTimestamp: "2025-03-06T10:02:56Z"
  finalizers:
    - nonadminbackup.oadp.openshift.io/finalizer
  generation: 2
  name: test-nab
  namespace: test-nac-ns
  resourceVersion: "134316"
  uid: c5...4c8a8
spec:
  backupSpec:
    csiSnapshotTimeout: 0s
    defaultVolumesToFsBackup: true
    hooks: {}
    itemOperationTimeout: 0s
    metadata: {}
    storageLocation: test-bsl
    ttl: 0s
status:
  conditions:
    - lastTransitionTime: "202...56Z"
      message: backup accepted
      reason: BackupAccepted
      status: "True"
      type: Accepted
    - lastTransitionTime: "202..T10:02:56Z"
      message: Created Velero Backup object
      reason: BackupScheduled
      status: "True"
      type: Queued
  dataMoverDataUploads: {}
  fileSystemPodVolumeBackups:
    completed: 2
    total: 2
  phase: Created
  queueInfo:
    estimatedQueuePosition: 0
  veleroBackup:
    nacuuid: test-nac-test-nab-d2...a9b14
    name: test-nac-test-nab-d2...b14
    namespace: openshift-adp
    spec:
      csiSnapshotTimeout: 10m0s
      defaultVolumesToFsBackup: true
      excludedResources:
        - nonadminbackups
        - nonadminrestores
        - nonadminbackupstoragelocations
        - securitycontextconstraints
        - clusterroles
```



```

- clusterrolebindings
- priorityclasses
- customresourcedefinitions
- virtualmachineclusterinstancetypes
- virtualmachineclusterpreferences
hooks: {}
includedNamespaces:
- test-nac-ns
itemOperationTimeout: 4h0m0s
metadata: {}
snapshotMoveData: false
storageLocation: test-nac-test-bsl-bf..02b70a
ttl: 720h0m0s
status:
  completionTimestamp: "2025-0..3:13Z"
  expiration: "2025..2:56Z"
  formatVersion: 1.1.0
  hookStatus: {}
  phase: Completed
  progress:
    itemsBackedUp: 46
    totalItems: 46
  startTimestamp: "2025-..56Z"
  version: 1
  warnings: 1

```

where:

namespace

Specifies the namespace name that the **NonAdminController** CR sets on the **Velero** backup object to back up.

message: backup accepted

Specifies that the NAC has reconciled and validated the NAB CR and has created the **Velero** backup object.

fileSystemPodVolumeBackups

Specifies the number of volumes that are backed up by using FSB.

phase: Created

Specifies that the NAB CR is in the **Created** phase.

estimatedQueuePosition

Specifies the queue position of the backup object. There can be multiple backups in process, and each backup object is assigned a queue position. When the backup is complete, the queue position is set to **0**.

nacuuid

Specifies that the NAC creates the **Velero** backup object and sets the value for the **nacuuid** field.

name

Specifies the name of the associated **Velero** backup object.

status

Specifies the status of the **Velero** backup object.

phase: Completed

Specifies that the **Velero** backup object is in the **Completed** phase and the backup is successful.

5.19.3.3. Deleting a NonAdminBackup CR

As a namespace admin user, you can delete a **NonAdminBackup** (NAB) custom resource (CR).

Prerequisites

- You are logged in to the cluster as a namespace admin user.
- The cluster administrator has installed the OADP Operator.
- The cluster administrator has configured the **DataProtectionApplication** (DPA) CR to enable OADP Self-Service.
- The cluster administrator has created a namespace for you and has authorized you to operate from that namespace.
- You have created a NAB CR in your authorized namespace.

Procedure

1. Edit the **NonAdminBackup** CR YAML manifest file by running the following command:

```
$ oc edit <nab_cr> -n <authorized_namespace>
```

where:

<nab_cr>

Specifies the name of the NAB CR to be deleted.

<authorized_namespace>

Specifies the name of your authorized namespace.

2. Update the NAB CR YAML manifest file and add the **deleteBackup** flag as shown in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: NonAdminBackup
metadata:
  name: <nab_cr>
spec:
  backupSpec:
    includedNamespaces:
      - <authorized_namespace>
    deleteBackup: true
```

where:

<nab_cr>

Specify the name of the NAB CR to be deleted.

<authorized_namespace>

Specify the name of your authorized namespace.

deleteBackup: true

Add the **deleteBackup** flag and set it to **true**.

Verification

- Verify that the NAB CR is deleted by running the following command:

```
$ oc get nab <nab_cr>
```

<nab_cr> is the name of the NAB CR you deleted.

You should see an output as shown in the following example:

```
Error from server (NotFound): nonadminbackups.oadp.openshift.io "test-nab" not found
```

5.19.3.4. Creating a NonAdminRestore CR

Create a **NonAdminRestore** (NAR) custom resource (CR) to restore application resources from a backup to your authorized namespace. This provides an ability to recover your application data and configuration without requiring cluster administrator privileges.

Prerequisites

- You are logged in to the cluster as a namespace admin user.
- The cluster administrator has installed the OADP Operator.
- The cluster administrator has configured the **DataProtectionApplication** (DPA) CR to enable OADP Self-Service.
- The cluster administrator has created a namespace for you and has authorized you to operate from that namespace.
- You have a backup of your application by creating a **NonAdminBackup** (NAB) CR.

Procedure

1. To create a **NonAdminRestore** CR, create a YAML manifest file with the following configuration:

Example NonAdminRestore CR

```
apiVersion: oadp.openshift.io/v1alpha1
kind: NonAdminRestore
metadata:
  name: test-nar
spec:
  restoreSpec:
    backupName: test-nab
```

where:

name

Specifies a name for the NAR CR. For example, **test-nar**.

backupName

Specifies the name of the NAB CR you want to restore from. For example, **test-nab**.

2. To apply the NAR CR configuration, run the following command:

```
$ oc apply -f <nar_cr_filename>
```

Replace **<nar_cr_filename>** with the file name containing the NAR CR configuration.

Verification

1. To verify that the NAR CR is successfully created, run the following command:

```
$ oc get nar test-nar -o yaml
```

Example output

```
apiVersion: oadp.openshift.io/v1alpha1
kind: NonAdminRestore
metadata:
  creationTimestamp: "2025-...:15Z"
  finalizers:
    - nonadminrestore.oadp.openshift.io/finalizer
  generation: 2
  name: test-nar
  namespace: test-nac-ns
  resourceVersion: "156517"
  uid: f9f5...63ef34
spec:
  restoreSpec:
    backupName: test-nab
    hooks: {}
    itemOperationTimeout: 0s
status:
  conditions:
    - lastTransitionTime: "2025..15Z"
      message: restore accepted
      reason: RestoreAccepted
      status: "True"
      type: Accepted
    - lastTransitionTime: "2025-03-06T11:22:15Z"
      message: Created Velero Restore object
      reason: RestoreScheduled
      status: "True"
      type: Queued
  dataMoverDataDownloads: {}
  fileSystemPodVolumeRestores:
    completed: 2
    total: 2
  phase: Created
  queueInfo:
    estimatedQueuePosition: 0
```

```

veleroRestore:
  nacuuid: test-nac-test-nar-c...1ba
  name: test-nac-test-nar-c7...1ba
  namespace: openshift-adp
  status:
    completionTimestamp: "2025...22:44Z"
    hookStatus: {}
    phase: Completed
    progress:
      itemsRestored: 28
      totalItems: 28
    startTimestamp: "2025..15Z"
    warnings: 7

```

where:

message: restore accepted

Specifies that the **NonAdminController** (NAC) CR has reconciled and validated the NAR CR.

fileSystemPodVolumeRestores

Specifies the number of volumes that are restored.

phase: Created

Specifies that the NAR CR is in the **Created** phase.

estimatedQueuePosition

Specifies the queue position of the restore object. There can be multiple restores in process, and each restore is assigned a queue position. When the restore is complete, the queue position is set to **0**.

nacuuid

Specifies that the NAC creates the **Velero** restore object and sets the **nacuuid** value.

name

Specifies the name of the associated **Velero** restore object.

phase: Completed

Specifies that the **Velero** restore object is in the **Completed** phase and the restore is successful.

5.19.3.5. About NonAdminDownloadRequest CR

Review backup and restore logs by using the **NonAdminDownloadRequest** (NADR) custom resource (CR). This helps you troubleshoot backup and restore issues without cluster administrator assistance.

The NADR CR provides information that is equivalent to what a cluster administrator can access by using the **velero backup describe --details** command.

After the NADR CR request is validated, a secure download URL is generated to access the requested information.

You can download the following NADR resources:

Table 5.8. NADR resources

Resource type	Description	Equivalent to
BackupResourceList	List of resources included in the backup	velero backup describe --details (resource listing)
BackupContents	Contents of files backed up	Part of backup details
BackupLog	Logs from the backup operation	velero backup logs
BackupVolumeSnapshots	Information about volume snapshots	velero backup describe --details (snapshots section)
BackupItemOperations	Information about item operations performed during backup	velero backup describe --details (operations section)
RestoreLog	Logs from the restore operation	velero restore logs
RestoreResults	Detailed results of the restore	velero restore describe --details

5.19.3.6. Reviewing NAB and NAR logs

Create a **NonAdminDownloadRequest** (NADR) custom resource (CR) to access and review detailed logs for **NonAdminBackup** (NAB) and **NonAdminRestore** (NAR) operations. This helps you troubleshoot backup and restore issues independently.



NOTE

You can review the NAB logs only if you are using a **NonAdminBackupStorageLocation** (NABSL) CR as a backup storage location for the backup.

Prerequisites

- You are logged in to the cluster as a namespace admin user.
- The cluster administrator has installed the OADP Operator.
- The cluster administrator has configured the **DataProtectionApplication** (DPA) CR to enable OADP Self-Service.
- The cluster administrator has created a namespace for you and has authorized you to operate from that namespace.
- You have a backup of your application by creating a NAB CR.
- You have restored the application by creating a NAR CR.

Procedure

1. To review NAB CR logs, create a **NonAdminDownloadRequest** CR and specify the NAB CR name as shown in the following example:

Example NonAdminDownloadRequest CR

```
apiVersion: oadp.openshift.io/v1alpha1
kind: NonAdminDownloadRequest
metadata:
  name: test-nadr-backup
spec:
  target:
    kind: BackupLog
    name: test-nab
```

where:

kind

Specifies **BackupLog** as the value for the **kind** field of the NADR CR.

name

Specifies the name of the NAB CR.

2. Verify that the NADR CR is processed by running the following command:

```
$ oc get nadr test-nadr-backup -o yaml
```

Example output

```
apiVersion: oadp.openshift.io/v1alpha1
kind: NonAdminDownloadRequest
metadata:
  creationTimestamp: "2025-03-06T10:05:22Z"
  generation: 1
  name: test-nadr-backup
  namespace: test-nac-ns
  resourceVersion: "134866"
  uid: 520...8d9
spec:
  target:
    kind: BackupLog
    name: test-nab
status:
  conditions:
    - lastTransitionTime: "202...5:22Z"
      message: ""
      reason: Success
      status: "True"
      type: Processed
  phase: Created
velero:
  status:
    downloadURL: https://...
    expiration: "202...22Z"
    phase: Processed
```

where:

downloadURL

The **status.downloadURL** field contains the download URL of the NAB logs. You can use the **downloadURL** to download and review the NAB logs.

phase

The **status.phase** is **Processed**.

- Download and analyze the backup information by using the **status.downloadURL** URL.
- To review NAR CR logs, create a **NonAdminDownloadRequest** CR and specify the NAR CR name as shown in the following example:

Example NonAdminDownloadRequest CR

```
apiVersion: oadp.openshift.io/v1alpha1
kind: NonAdminDownloadRequest
metadata:
  name: test-nadr-restore
spec:
  target:
    kind: RestoreLog
    name: test-nar
```

where:

kind

Specifies **RestoreLog** as the value for the **kind** field of the NADR CR.

name

Specifies the name of the NAR CR.

- Verify that the NADR CR is processed by running the following command:

```
$ oc get nadr test-nadr-restore -o yaml
```

Example output

```
apiVersion: oadp.openshift.io/v1alpha1
kind: NonAdminDownloadRequest
metadata:
  creationTimestamp: "2025-03-06T11:26:01Z"
  generation: 1
  name: test-nadr-restore
  namespace: test-nac-ns
  resourceVersion: "157842"
  uid: f3e...7862f
spec:
  target:
    kind: RestoreLog
    name: test-nar
status:
  conditions:
```



```
- lastTransitionTime: "202...01Z"
  message: ""
  reason: Success
  status: "True"
  type: Processed
phase: Created
velero:
  status:
    downloadURL: https://...
    expiration: "202...01Z"
    phase: Processed
```

where:

downloadURL

The **status.downloadURL** field contains the download URL of the NAR logs. You can use the **downloadURL** to download and review the NAR logs.

phase

The **status.phase** is **Processed**.

- Download and analyze the restore information by using the **status.downloadURL** URL.

5.19.4. OADP Self-Service troubleshooting

Resolve common errors and issues when using OADP Self-Service by following troubleshooting procedures for backup storage locations and backup operations. This helps you quickly identify and fix problems independently.

5.19.4.1. Resolving error **NonAdminBackupStorageLocation not found in the namespace**

Resolve the **NonAdminBackupStorageLocation not found in the namespace** error by using a backup storage location that belongs to the same namespace as your backup. This helps ensure successful backup operations.

Consider the following scenario of a namespace **admin** backup:

- You have created two **NonAdminBackupStorageLocations** (NABs) custom resources (CRs) in two different namespaces, for example, **nabsl-1** in **namespace-1** and **nabsl-2** in **namespace-2**.
- You are taking a backup of **namespace-1** and use **nabsl-2** in the **NonAdminBackup** (NAB) CR.

In this scenario, after creating the NAB CR, you get the following error. The cause of the error is that the NABSL CR does not belong to the namespace that you are trying to back up.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: NonAdminBackup
...
status:
  conditions:
    - lastTransitionTime: "2025-02-20T10:13:00Z"
      message: 'NonAdminBackupStorageLocation not found in the namespace:
NonAdminBackupStorageLocation.oadp.openshift.io
"nabsl2" not found'
```

```

reason: InvalidBackupSpec
status: "False"
type: Accepted
phase: BackingOff

```

Procedure

- Use the NABSL that belongs to the same namespace that you are trying to back up. In this scenario, you must use **nabsl-1** in the NAB CR to back up **namespace-1**.

5.19.4.2. Resolving error `NonAdminBackupStorageLocation cannot be set as default`

Resolve the error that occurs when you set a **NonAdminBackupStorageLocation** (NABSL) custom resource (CR) as the default backup storage location. This helps you resolve validation errors and configure backup storage locations correctly.

As a non-admin user, if you have created a NABSL CR in your authorized namespace, you cannot set the NABSL CR as the default backup storage location.

If you set the NABSL CR as the default backup storage location, the NABSL CR fails to validate and the **NonAdminController** (NAC) gives an error message.

`NonAdminBackupStorageLocation cannot be used as a default BSL`

Procedure

- To successfully validate and reconcile the NABSL CR, set the **default** field to **false** in the NABSL CR:

```

apiVersion: oadp.openshift.io/v1alpha1
kind: NonAdminBackupStorageLocation
...
spec:
  backupStorageLocationSpec:
    credential:
      key: cloud
      name: cloud-credentials-gcp
    default: false
    objectStorage:
      bucket: oad..718
      prefix: velero
      provider: gcp

```

where:

default

Specifies that the **default** backup storage location is set to **false**.

5.20. OADP AND ROSA

5.20.1. Backing up applications on ROSA clusters using OADP

You can use OpenShift API for Data Protection (OADP) with Red Hat OpenShift Service on AWS (ROSA) clusters to back up and restore application data.

ROSA is a fully-managed, turnkey application platform that allows you to deliver value to your customers by building and deploying applications.

ROSA provides seamless integration with a wide range of Amazon Web Services (AWS) compute, database, analytics, machine learning, networking, mobile, and other services to speed up the building and delivery of differentiating experiences to your customers.

You can subscribe to the service directly from your AWS account.

After you create your clusters, you can operate your clusters with the OpenShift Container Platform web console or through [Red Hat OpenShift Cluster Manager](#). You can also use ROSA with OpenShift APIs and command-line interface (CLI) tools.

For additional information about ROSA installation, see [Installing Red Hat OpenShift Service on AWS \(ROSA\) interactive walkthrough](#).

Before installing OpenShift API for Data Protection (OADP), you must set up role and policy credentials for OADP so that it can use the Amazon Web Services API.

This process is performed in the following two stages:

1. Prepare AWS credentials
2. Install the OADP Operator and give it an IAM role

5.20.1.1. Preparing AWS credentials for OADP

An Amazon Web Services account must be prepared and configured to accept an OpenShift API for Data Protection (OADP) installation.

Procedure

1. Create the following environment variables by running the following commands:



IMPORTANT

Change the cluster name to match your cluster, and ensure you are logged into the cluster as an administrator. Ensure that all fields are outputted correctly before continuing.

```
$ export CLUSTER_NAME=my-cluster
```

- **my-cluster**: Replace **my-cluster** with your cluster name.

```
$ export ROSA_CLUSTER_ID=$(rosa describe cluster -c ${CLUSTER_NAME} --output json | jq -r .id)
```

```
$ export REGION=$(rosa describe cluster -c ${CLUSTER_NAME} --output json | jq -r .region.id)
```

```

$ export OIDC_ENDPOINT=$(oc get authentication.config.openshift.io cluster -o
jsonpath='{.spec.serviceAccountIssuer}' | sed 's|^https://|')

$ export AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)

$ export CLUSTER_VERSION=$(rosa describe cluster -c ${CLUSTER_NAME} -o json | jq -r
.version.raw_id | cut -f -2 -d '.')

$ export ROLE_NAME="${CLUSTER_NAME}-openshift-oadp-aws-cloud-credentials"

$ export SCRATCH="/tmp/${CLUSTER_NAME}/oadp"

$ mkdir -p ${SCRATCH}

$ echo "Cluster ID: ${ROSA_CLUSTER_ID}, Region: ${REGION}, OIDC Endpoint:
${OIDC_ENDPOINT}, AWS Account ID: ${AWS_ACCOUNT_ID}"

```

2. On the AWS account, create an IAM policy to allow access to AWS S3:

- a. Check to see if the policy exists by running the following command:

```

$ POLICY_ARN=$(aws iam list-policies --query "Policies[?
PolicyName=='RosaOadpVer1'].{ARN:Arn}" --output text)

```

- **RosaOadp**: Replace **RosaOadp** with your policy name.

- b. Enter the following command to create the policy JSON file and then create the policy:



NOTE

If the policy ARN is not found, the command creates the policy. If the policy ARN already exists, the **if** statement intentionally skips the policy creation.

```

$ if [[ -z "${POLICY_ARN}" ]]; then
  cat << EOF > ${SCRATCH}/policy.json
  {
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
          "s3:CreateBucket",
          "s3:DeleteBucket",
          "s3:PutBucketTagging",
          "s3:GetBucketTagging",
          "s3:PutEncryptionConfiguration",
          "s3:GetEncryptionConfiguration",
          "s3:PutLifecycleConfiguration",
          "s3:GetLifecycleConfiguration",
          "s3:GetBucketLocation",
          "s3:ListBucket",

```

```

        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject",
        "s3:ListBucketMultipartUploads",
        "s3:AbortMultipartUpload",
        "s3:ListMultipartUploadParts",
        "ec2:DescribeSnapshots",
        "ec2:DescribeVolumes",
        "ec2:DescribeVolumeAttribute",
        "ec2:DescribeVolumesModifications",
        "ec2:DescribeVolumeStatus",
        "ec2:CreateTags",
        "ec2:CreateVolume",
        "ec2:CreateSnapshot",
        "ec2:DeleteSnapshot"
    ],
    "Resource": "*"
}
]]
EOF

```

```

POLICY_ARN=$(aws iam create-policy --policy-name "RosaOadpVer1" \
--policy-document file:///${SCRATCH}/policy.json --query Policy.Arn \
--tags Key=rosa_openshift_version,Value=${CLUSTER_VERSION}
Key=rosa_role_prefix,Value=ManagedOpenShift
Key=operator_namespace,Value=openshift-oadp Key=operator_name,Value=openshift-
oadp \
--output text)
fi

```

- **SCRATCH: SCRATCH** is a name for a temporary directory created for the environment variables.

c. View the policy ARN by running the following command:

```
$ echo ${POLICY_ARN}
```

3. Create an IAM role trust policy for the cluster:

a. Create the trust policy file by running the following command:

```

$ cat <<EOF > ${SCRATCH}/trust-policy.json
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {
      "Federated": "arn:aws:iam::${AWS_ACCOUNT_ID}:oidc-
provider/${OIDC_ENDPOINT}"
    },
    "Action": "sts:AssumeRoleWithWebIdentity",
    "Condition": {
      "StringEquals": {
        "${OIDC_ENDPOINT}:sub": [
          "system:serviceaccount:openshift-adp:openshift-adp-controller-manager",
          "system:serviceaccount:openshift-adp:velero"
        ]
      }
    }
  }]
}

```

```
}  
}  
}}  
}  
EOF
```

- b. Create the role by running the following command:

```
$ ROLE_ARN=$(aws iam create-role --role-name \  
"${ROLE_NAME}" \  
--assume-role-policy-document file://${SCRATCH}/trust-policy.json \  
--tags Key=rosa_cluster_id,Value=${ROSA_CLUSTER_ID} \  
Key=rosa_openshift_version,Value=${CLUSTER_VERSION} \  
Key=rosa_role_prefix,Value=ManagedOpenShift \  
Key=operator_namespace,Value=openshift-adp \  
Key=operator_name,Value=openshift-oadp \  
--query Role.Arn --output text)
```

- c. View the role ARN by running the following command:

```
$ echo ${ROLE_ARN}
```

4. Attach the IAM policy to the IAM role by running the following command:

```
$ aws iam attach-role-policy --role-name "${ROLE_NAME}" \  
--policy-arn ${POLICY_ARN}
```

5.20.1.2. Installing the OADP Operator and providing the IAM role

AWS Security Token Service (AWS STS) is a global web service that provides short-term credentials for IAM or federated users. OpenShift Container Platform with STS is the recommended credential mode. This document describes how to install OpenShift API for Data Protection (OADP) on clusters with AWS STS.

IMPORTANT

Restic is unsupported.

Kopia file system backup (FSB) is supported when backing up file systems that do not support Container Storage Interface (CSI) snapshots.

Example file systems include the following:

- Amazon Elastic File System (EFS)
- Network File System (NFS)
- **emptyDir** volumes
- Local volumes

For backing up volumes, OADP on ROSA with AWS STS recommends native snapshots and Container Storage Interface (CSI) snapshots. Data Mover backups are supported, but can be slower than native snapshots.

In an Amazon ROSA cluster that uses STS authentication, restoring backed-up data in a different AWS region is not supported.

Prerequisites

- An OpenShift Container Platform cluster with the required access and tokens. For instructions, see the previous procedure *Preparing AWS credentials for OADP*. If you plan to use two different clusters for backing up and restoring, you must prepare AWS credentials, including **ROLE_ARN**, for each cluster.

Procedure

1. Create an OpenShift Container Platform secret from your AWS token file by entering the following commands:

- a. Create the credentials file:

```
$ cat <<EOF > ${SCRATCH}/credentials
[default]
role_arn = ${ROLE_ARN}
web_identity_token_file = /var/run/secrets/openshift/serviceaccount/token
region = <aws_region> 1
EOF
```

- 1** Replace **<aws_region>** with the AWS region to use for the STS endpoint.

- b. Create a namespace for OADP:

```
$ oc create namespace openshift-adp
```

- c. Create the OpenShift Container Platform secret:

```
$ oc -n openshift-adp create secret generic cloud-credentials \
--from-file=${SCRATCH}/credentials
```

■

**NOTE**

In OpenShift Container Platform versions 4.15 and later, the OADP Operator supports a new standardized STS workflow through the Operator Lifecycle Manager (OLM) and Cloud Credentials Operator (CCO). In this workflow, you do not need to create the above secret, you only need to supply the role ARN during the installation of OLM-managed operators using the OpenShift Container Platform web console, for more information see *Installing from software catalog using the web console*.

The preceding secret is created automatically by CCO.

2. Install the OADP Operator:
 - a. In the OpenShift Container Platform web console, browse to **Ecosystem → Software Catalog**.
 - b. Search for the **OADP Operator**.
 - c. In the **role_ARN** field, paste the `role_arn` that you created previously and click **Install**.
3. Create AWS cloud storage using your AWS credentials by entering the following command:

```
$ cat << EOF | oc create -f -
  apiVersion: oadp.openshift.io/v1alpha1
  kind: CloudStorage
  metadata:
    name: ${CLUSTER_NAME}-oadp
    namespace: openshift-adp
  spec:
    creationSecret:
      key: credentials
      name: cloud-credentials
    enableSharedConfig: true
    name: ${CLUSTER_NAME}-oadp
    provider: aws
    region: $REGION
EOF
```

4. Check your application's storage default storage class by entering the following command:

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

Example output

NAME	STATUS	VOLUME	CAPACITY	ACCESS MODES
STORAGECLASS	AGE			
applog	Bound	pvc-351791ae-b6ab-4e8b-88a4-30f73caf5ef8	1Gi	RWO gp3-
csi	4d19h			
mysql	Bound	pvc-16b8e009-a20a-4379-accb-bc81fedd0621	1Gi	RWO gp3-
csi	4d19h			

5. Get the storage class by running the following command:


```
$ oc get storageclass
```

Example output

NAME	PROVISIONER	RECLAIMPOLICY	VOLUMEBINDINGMODE	ALLOWVOLUMEEXPANSION	AGE
gp2	kubernetes.io/aws-efs	Delete	WaitForFirstConsumer	true	4d21h
gp2-csi	ebs.csi.aws.com	Delete	WaitForFirstConsumer	true	4d21h
gp3	ebs.csi.aws.com	Delete	WaitForFirstConsumer	true	4d21h
gp3-csi (default)	ebs.csi.aws.com	Delete	WaitForFirstConsumer	true	4d21h

NOTE

The following storage classes will work:

- gp3-csi
- gp2-csi
- gp3
- gp2

If the application or applications that are being backed up are all using persistent volumes (PVs) with Container Storage Interface (CSI), it is advisable to include the CSI plugin in the OADP DPA configuration.

6. Create the **DataProtectionApplication** resource to configure the connection to the storage where the backups and volume snapshots are stored:
 - a. If you are using only CSI volumes, deploy a Data Protection Application by entering the following command:

```
$ cat << EOF | oc create -f -
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: ${CLUSTER_NAME}-dpa
  namespace: openshift-adp
spec:
  backupImages: true ❶
  features:
    dataMover:
      enable: false
  backupLocations:
  - bucket:
      cloudStorageRef:
        name: ${CLUSTER_NAME}-oadp
      credential:
        key: credentials
```

```

    name: cloud-credentials
    prefix: velero
    default: true
    config:
      region: ${REGION}
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - aws
        - csi
      nodeAgent: ❷
      enable: false
      uploaderType: kopia ❸
EOF

```

- ❶ ROSA supports internal image backup. Set this field to **false** if you do not want to use image backup.
- ❷ See the important note regarding the **nodeAgent** attribute at the end of this procedure.
- ❸ The type of uploader. The built-in Data Mover uses Kopia as the default uploader mechanism regardless of the value of the **uploaderType** field.

- b. If you are using CSI or non-CSI volumes, deploy a Data Protection Application by entering the following command:

```

$ cat << EOF | oc create -f -
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: ${CLUSTER_NAME}-dpa
  namespace: openshift-adp
spec:
  backupImages: true ❶
  backupLocations:
    - bucket:
        cloudStorageRef:
          name: ${CLUSTER_NAME}-oadp
        credential:
          key: credentials
          name: cloud-credentials
          prefix: velero
          default: true
          config:
            region: ${REGION}
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - aws
      nodeAgent: ❷
      enable: false
      uploaderType: restic

```

```

snapshotLocations:
- velero:
  config:
    credentialsFile: /tmp/credentials/openshift-adp/cloud-credentials-credentials 3
    enableSharedConfig: "true" 4
    profile: default 5
    region: ${REGION} 6
    provider: aws
EOF

```

- 1** ROSA supports internal image backup. Set this field to **false** if you do not want to use image backup.
- 2** See the important note regarding the **nodeAgent** attribute at the end of this procedure.
- 3** The **credentialsFile** field is the mounted location of the bucket credential on the pod.
- 4** The **enableSharedConfig** field allows the **snapshotLocations** to share or reuse the credential defined for the bucket.
- 5** Use the profile name set in the AWS credentials file.
- 6** Specify **region** as your AWS region. This must be the same as the cluster region.

You are now ready to back up and restore OpenShift Container Platform applications, as described in *Backing up applications*.

IMPORTANT

The **enable** parameter of **restic** is set to **false** in this configuration, because OADP does not support Restic in ROSA environments.

If you use OADP 1.2, replace this configuration:

```

nodeAgent:
  enable: false
  uploaderType: restic

```

with the following configuration:

```

restic:
  enable: false

```

If you want to use two different clusters for backing up and restoring, the two clusters must have the same AWS S3 storage names in both the cloud storage CR and the OADP **DataProtectionApplication** configuration.

5.20.1.3. Updating the IAM role ARN in the OADP Operator subscription

While installing the OADP Operator on a ROSA Security Token Service (STS) cluster, if you provide an incorrect IAM role Amazon Resource Name (ARN), the **openshift-adp-controller** pod gives an error.

The credential requests that are generated contain the wrong IAM role ARN. To update the credential requests object with the correct IAM role ARN, you can edit the OADP Operator subscription and patch the IAM role ARN with the correct value. By editing the OADP Operator subscription, you do not have to uninstall and reinstall OADP to update the IAM role ARN.

Prerequisites

- You have a Red Hat OpenShift Service on AWS STS cluster with the required access and tokens.
- You have installed OADP on the ROSA STS cluster.

Procedure

1. To verify that the OADP subscription has the wrong IAM role ARN environment variable set, run the following command:

```
$ oc get sub -o yaml redhat-oadp-operator
```

Example subscription

```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  annotations:
    creationTimestamp: "2025-01-15T07:18:31Z"
    generation: 1
  labels:
    operators.coreos.com/redhat-oadp-operator.openshift-adp: ""
name: redhat-oadp-operator
namespace: openshift-adp
resourceVersion: "77363"
uid: 5ba00906-5ad2-4476-ae7b-ffa90986283d
spec:
  channel: stable-1.4
  config:
    env:
      - name: ROLEARN
        value: arn:aws:iam::11111111:role/wrong-role-arn 1
installPlanApproval: Manual
name: redhat-oadp-operator
source: prestage-operators
sourceNamespace: openshift-marketplace
startingCSV: oadp-operator.v1.4.2
```

1. Verify the value of **ROLEARN** you want to update.

2. Update the **ROLEARN** field of the subscription with the correct role ARN by running the following command:

```
$ oc patch subscription redhat-oadp-operator -p '{"spec": {"config": {"env": [{"name": "ROLEARN", "value": "<role_arn>"}]}}}' --type='merge'
```

where:

<role_arn>

Specifies the IAM role ARN to be updated. For example, **arn:aws:iam::160...
..6956:role/oadprosa.....8wlf**.

3. Verify that the **secret** object is updated with correct role ARN value by running the following command:

```
$ oc get secret cloud-credentials -o jsonpath='{.data.credentials}' | base64 -d
```

Example output

```
[default]
sts_regional_endpoints = regional
role_arn = arn:aws:iam::160.....6956:role/oadprosa.....8wlf
web_identity_token_file = /var/run/secrets/openshift/serviceaccount/token
```

4. Configure the **DataProtectionApplication** custom resource (CR) manifest file as shown in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: test-rosa-dpa
  namespace: openshift-adp
spec:
  backupLocations:
  - bucket:
      config:
        region: us-east-1
      cloudStorageRef:
        name: <cloud_storage> 1
      credential:
        name: cloud-credentials
        key: credentials
      prefix: velero
      default: true
  configuration:
    velero:
      defaultPlugins:
      - aws
      - openshift
```

1 Specify the **CloudStorage** CR.

5. Create the **DataProtectionApplication** CR by running the following command:

```
$ oc create -f <dpa_manifest_file>
```

6. Verify that the **DataProtectionApplication** CR is reconciled and the **status** is set to **"True"** by running the following command:

■

```
$ oc get dpa -n openshift-adp -o yaml
```

Example DataProtectionApplication

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
status:
  conditions:
  - lastTransitionTime: "2023-07-31T04:48:12Z"
    message: Reconcile complete
    reason: Complete
    status: "True"
    type: Reconciled
```

7. Verify that the **BackupStorageLocation** CR is in an available state by running the following command:

```
$ oc get backupstoragelocations.velero.io -n openshift-adp
```

Example BackupStorageLocation

NAME	PHASE	LAST VALIDATED	AGE	DEFAULT
ts-dpa-1	Available	3s	6s	true

Additional resources

- [Installing from the software catalog using the web console](#) .
- [Backing up applications](#)

5.20.1.4. Example: Backing up workload on OADP ROSA STS, with an optional cleanup

5.20.1.4.1. Performing a backup with OADP and OpenShift Container Platform

The following example **hello-world** application has no persistent volumes (PVs) attached. Perform a backup by using OpenShift API for Data Protection (OADP) with OpenShift Container Platform.

Either Data Protection Application (DPA) configuration will work.

Procedure

1. Create a workload to back up by running the following commands:

```
$ oc create namespace hello-world
```

```
$ oc new-app -n hello-world --image=docker.io/openshift/hello-openshift
```

2. Expose the route by running the following command:

```
$ oc expose service/hello-openshift -n hello-world
```

3. Check that the application is working by running the following command:

```
$ curl `oc get route/hello-openshift -n hello-world -o jsonpath='{.spec.host}'`
```

You should see an output similar to the following example:

```
Hello OpenShift!
```

4. Back up the workload by running the following command:

```
$ cat << EOF | oc create -f -
  apiVersion: velero.io/v1
  kind: Backup
  metadata:
    name: hello-world
    namespace: openshift-adp
  spec:
    includedNamespaces:
      - hello-world
    storageLocation: ${CLUSTER_NAME}-dpa-1
    ttl: 720h0m0s
EOF
```

5. Wait until the backup is complete, and then run the following command:

```
$ watch "oc -n openshift-adp get backup hello-world -o json | jq .status"
```

You should see an output similar to the following example:

```
{
  "completionTimestamp": "2022-09-07T22:20:44Z",
  "expiration": "2022-10-07T22:20:22Z",
  "formatVersion": "1.1.0",
  "phase": "Completed",
  "progress": {
    "itemsBackedUp": 58,
    "totalItems": 58
  },
  "startTimestamp": "2022-09-07T22:20:22Z",
  "version": 1
}
```

6. Delete the demo workload by running the following command:

```
$ oc delete ns hello-world
```

7. Restore the workload from the backup by running the following command:

```
$ cat << EOF | oc create -f -
  apiVersion: velero.io/v1
  kind: Restore
  metadata:
    name: hello-world
    namespace: openshift-adp
```

```
spec:
  backupName: hello-world
EOF
```

- Wait for the Restore to finish by running the following command:

```
$ watch "oc -n openshift-adp get restore hello-world -o json | jq .status"
```

You should see an output similar to the following example:

```
{
  "completionTimestamp": "2022-09-07T22:25:47Z",
  "phase": "Completed",
  "progress": {
    "itemsRestored": 38,
    "totalItems": 38
  },
  "startTimestamp": "2022-09-07T22:25:28Z",
  "warnings": 9
}
```

- Check that the workload is restored by running the following command:

```
$ oc -n hello-world get pods
```

You should see an output similar to the following example:

```
NAME                                READY STATUS RESTARTS AGE
hello-openshift-9f885f7c6-kdjpi 1/1   Running 0      90s
```

- Check the JSONPath by running the following command:

```
$ curl `oc get route/hello-openshift -n hello-world -o jsonpath='{.spec.host}'`
```

You should see an output similar to the following example:

```
Hello OpenShift!
```



NOTE

For troubleshooting tips, see the [troubleshooting documentation](#).

5.20.1.4.2. Cleaning up a cluster after a backup with OADP and ROSA STS

If you need to uninstall the OpenShift API for Data Protection (OADP) Operator together with the backups and the S3 bucket from this example, follow these instructions.

Procedure

- Delete the workload by running the following command:

```
$ oc delete ns hello-world
```


2. Delete the Data Protection Application (DPA) by running the following command:

```
$ oc -n openshift-adp delete dpa ${CLUSTER_NAME}-dpa
```

3. Delete the cloud storage by running the following command:

```
$ oc -n openshift-adp delete cloudstorage ${CLUSTER_NAME}-oadp
```



WARNING

If this command hangs, you might need to delete the finalizer by running the following command:

```
$ oc -n openshift-adp patch cloudstorage ${CLUSTER_NAME}-oadp -p '{"metadata":{"finalizers":null}}' --type=merge
```

4. If the Operator is no longer required, remove it by running the following command:

```
$ oc -n openshift-adp delete subscription oadp-operator
```

5. Remove the namespace from the Operator:

```
$ oc delete ns openshift-adp
```

6. If the backup and restore resources are no longer required, remove them from the cluster by running the following command:

```
$ oc delete backups.velero.io hello-world
```

7. To delete backup, restore and remote objects in AWS S3 run the following command:

```
$ velero backup delete hello-world
```

8. If you no longer need the Custom Resource Definitions (CRD), remove them from the cluster by running the following command:

```
$ for CRD in `oc get crds | grep velero | awk '{print $1}'`; do oc delete crd $CRD; done
```

9. Delete the AWS S3 bucket by running the following commands:

```
$ aws s3 rm s3://${CLUSTER_NAME}-oadp --recursive
```

```
$ aws s3api delete-bucket --bucket ${CLUSTER_NAME}-oadp
```

10. Detach the policy from the role by running the following command:

```
$ aws iam detach-role-policy --role-name "${ROLE_NAME}" --policy-arn "${POLICY_ARN}"
```

11. Delete the role by running the following command:

```
$ aws iam delete-role --role-name "${ROLE_NAME}"
```

5.21. OADP AND AWS STS

5.21.1. Backing up applications on AWS STS using OADP

You install the OpenShift API for Data Protection (OADP) with Amazon Web Services (AWS) by installing the OADP Operator. The Operator installs [Velero 1.16](#).



NOTE

Starting from OADP 1.0.4, all OADP 1.0.z versions can only be used as a dependency of the Migration Toolkit for Containers Operator and are not available as a standalone Operator.

You configure AWS for Velero, create a default **Secret**, and then install the Data Protection Application. For more details, see [Installing the OADP Operator](#).

To install the OADP Operator in a restricted network environment, you must first disable the default software catalog sources and mirror the Operator catalog. See [Using Operator Lifecycle Manager in disconnected environments](#) for details.

You can install OADP on an AWS Security Token Service (STS) (AWS STS) cluster manually. Amazon AWS provides AWS STS as a web service that enables you to request temporary, limited-privilege credentials for users. You use STS to provide trusted users with temporary access to resources via API calls, your AWS console, or the AWS command-line interface (CLI).

Before installing OpenShift API for Data Protection (OADP), you must set up role and policy credentials for OADP so that it can use the Amazon Web Services API.

This process is performed in the following two stages:

1. Prepare AWS credentials.
2. Install the OADP Operator and give it an IAM role.

5.21.1.1. Preparing AWS STS credentials for OADP

An Amazon Web Services account must be prepared and configured to accept an OpenShift API for Data Protection (OADP) installation. Prepare the AWS credentials by using the following procedure.

Procedure

1. Define the **cluster_name** environment variable by running the following command:

```
$ export CLUSTER_NAME= <AWS_cluster_name> 1
```

- 1 The variable can be set to any value.

2. Retrieve all of the details of the **cluster** such as the **AWS_ACCOUNT_ID**, **OIDC_ENDPOINT** by running the following command:

```
$ export CLUSTER_VERSION=$(oc get clusterversion version -o
jsonpath='{.status.desired.version}{"\n"}')
```

```
$ export AWS_CLUSTER_ID=$(oc get clusterversion version -o jsonpath='{.spec.clusterID}
{"\n"}')
```

```
$ export OIDC_ENDPOINT=$(oc get authentication.config.openshift.io cluster -o
jsonpath='{.spec.serviceAccountIssuer}' | sed 's|^https://|'|')
```

```
$ export REGION=$(oc get infrastructures cluster -o
jsonpath='{.status.platformStatus.aws.region}' --allow-missing-template-keys=false || echo
us-east-2)
```

```
$ export AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
```

```
$ export ROLE_NAME="${CLUSTER_NAME}-openshift-oadp-aws-cloud-credentials"
```

3. Create a temporary directory to store all of the files by running the following command:

```
$ export SCRATCH="/tmp/${CLUSTER_NAME}/oadp"
mkdir -p ${SCRATCH}
```

4. Display all of the gathered details by running the following command:

```
$ echo "Cluster ID: ${AWS_CLUSTER_ID}, Region: ${REGION}, OIDC Endpoint:
${OIDC_ENDPOINT}, AWS Account ID: ${AWS_ACCOUNT_ID}"
```

5. On the AWS account, create an IAM policy to allow access to AWS S3:

- a. Check to see if the policy exists by running the following commands:

```
$ export POLICY_NAME="OadpVer1"
```

- **POLICY_NAME:** The variable can be set to any value.

```
$ POLICY_ARN=$(aws iam list-policies --query "Policies[?
PolicyName=='$POLICY_NAME'].{ARN:Arn}" --output text)
```

- b. Enter the following command to create the policy JSON file and then create the policy:



NOTE

If the policy ARN is not found, the command creates the policy. If the policy ARN already exists, the **if** statement intentionally skips the policy creation.

```
$ if [[ -z "${POLICY_ARN}" ]]; then
cat << EOF > ${SCRATCH}/policy.json
{
```

```

"Version": "2012-10-17",
"Statement": [
{
  "Effect": "Allow",
  "Action": [
    "s3:CreateBucket",
    "s3:DeleteBucket",
    "s3:PutBucketTagging",
    "s3:GetBucketTagging",
    "s3:PutEncryptionConfiguration",
    "s3:GetEncryptionConfiguration",
    "s3:PutLifecycleConfiguration",
    "s3:GetLifecycleConfiguration",
    "s3:GetBucketLocation",
    "s3:ListBucket",
    "s3:GetObject",
    "s3:PutObject",
    "s3:DeleteObject",
    "s3:ListBucketMultipartUploads",
    "s3:AbortMultipartUpload",
    "s3:ListMultipartUploadParts",
    "ec2:DescribeSnapshots",
    "ec2:DescribeVolumes",
    "ec2:DescribeVolumeAttribute",
    "ec2:DescribeVolumesModifications",
    "ec2:DescribeVolumeStatus",
    "ec2:CreateTags",
    "ec2:CreateVolume",
    "ec2:CreateSnapshot",
    "ec2:DeleteSnapshot"
  ],
  "Resource": "*"
}
]}
EOF

```

```

POLICY_ARN=$(aws iam create-policy --policy-name $POLICY_NAME \
--policy-document file:///${SCRATCH}/policy.json --query Policy.Arn \
--tags Key=openshift_version,Value=${CLUSTER_VERSION} \
Key=operator_namespace,Value=openshift-adp Key=operator_name,Value=oadp \
--output text)
fi

```

- **SCRATCH**: The name for a temporary directory created for storing the files.

c. View the policy ARN by running the following command:

```
$ echo ${POLICY_ARN}
```

6. Create an IAM role trust policy for the cluster:

a. Create the trust policy file by running the following command:

```

$ cat <<EOF > ${SCRATCH}/trust-policy.json
{
  "Version": "2012-10-17",

```

```

    "Statement": [{
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::${AWS_ACCOUNT_ID}:oidc-
provider/${OIDC_ENDPOINT}"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "${OIDC_ENDPOINT}:sub": [
            "system:serviceaccount:openshift-adp:openshift-adp-controller-manager",
            "system:serviceaccount:openshift-adp:velero"
          ]
        }
      }
    ]
  }
}
EOF

```

- b. Create an IAM role trust policy for the cluster by running the following command:

```

$ ROLE_ARN=$(aws iam create-role --role-name \
"${ROLE_NAME}" \
--assume-role-policy-document file://${SCRATCH}/trust-policy.json \
--tags Key=cluster_id,Value=${AWS_CLUSTER_ID}
Key=openshift_version,Value=${CLUSTER_VERSION}
Key=operator_namespace,Value=openshift-adp Key=operator_name,Value=oadp --
query Role.Arn --output text)

```

- c. View the role ARN by running the following command:

```

$ echo ${ROLE_ARN}

```

7. Attach the IAM policy to the IAM role by running the following command:

```

$ aws iam attach-role-policy --role-name "${ROLE_NAME}" --policy-arn ${POLICY_ARN}

```

5.21.1.1.1. Setting Velero CPU and memory resource allocations

You set the CPU and memory resource allocations for the **Velero** pod by editing the **DataProtectionApplication** custom resource (CR) manifest.

Prerequisites

- You must have the OpenShift API for Data Protection (OADP) Operator installed.

Procedure

- Edit the values in the **spec.configuration.velero.podConfig.ResourceAllocations** block of the **DataProtectionApplication** CR manifest, as in the following example:

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>

```

```
spec:
# ...
configuration:
  velero:
    podConfig:
      nodeSelector: <node_selector>
      resourceAllocations:
        limits:
          cpu: "1"
          memory: 1024Mi
        requests:
          cpu: 200m
          memory: 256Mi
```

where:

nodeSelector

Specifies the node selector to be supplied to Velero podSpec.

resourceAllocations

Specifies the resource allocations listed for average usage.



NOTE

Kopia is an option in OADP 1.3 and later releases. You can use Kopia for file system backups, and Kopia is your only option for Data Mover cases with the built-in Data Mover.

Kopia is more resource intensive than Restic, and you might need to adjust the CPU and memory requirements accordingly.

5.21.1.2. Installing the OADP Operator and providing the IAM role

AWS Security Token Service (AWS STS) is a global web service that provides short-term credentials for IAM or federated users. This document describes how to install OpenShift API for Data Protection (OADP) on an AWS STS cluster manually.

IMPORTANT

Restic is unsupported.

Kopia file system backup (FSB) is supported when backing up file systems that do not support Container Storage Interface (CSI) snapshots.

Example file systems include the following:

- Amazon Elastic File System (EFS)
- Network File System (NFS)
- **emptyDir** volumes
- Local volumes

For backing up volumes, OADP on AWS STS recommends native snapshots and Container Storage Interface (CSI) snapshots. Data Mover backups are supported, but can be slower than native snapshots.

In an AWS cluster that uses STS authentication, restoring backed-up data in a different AWS region is not supported.

Prerequisites

- An OpenShift Container Platform AWS STS cluster with the required access and tokens. For instructions, see the previous procedure *Preparing AWS credentials for OADP*. If you plan to use two different clusters for backing up and restoring, you must prepare AWS credentials, including **ROLE_ARN**, for each cluster.

Procedure

1. Create an OpenShift Container Platform secret from your AWS token file by entering the following commands:

- a. Create the credentials file:

```
$ cat <<EOF > ${SCRATCH}/credentials
[default]
role_arn = ${ROLE_ARN}
web_identity_token_file = /var/run/secrets/openshift/serviceaccount/token
region = <aws_region> 1
EOF
```

- 1** Replace **<aws_region>** with the AWS region to use for the STS endpoint.

- b. Create a namespace for OADP:

```
$ oc create namespace openshift-adp
```

- c. Create the OpenShift Container Platform secret:

```
$ oc -n openshift-adp create secret generic cloud-credentials \
--from-file=${SCRATCH}/credentials
```

■

**NOTE**

In OpenShift Container Platform versions 4.14 and later, the OADP Operator supports a new standardized STS workflow through the Operator Lifecycle Manager (OLM) and Cloud Credentials Operator (CCO). In this workflow, you do not need to create the above secret, you only need to supply the role ARN during the installation of OLM-managed operators using the OpenShift Container Platform web console, for more information see *Installing from the software catalog using the web console*.

The preceding secret is created automatically by CCO.

2. Install the OADP Operator:
 - a. In the OpenShift Container Platform web console, browse to **Ecosystem → Software Catalog**.
 - b. Search for the **OADP Operator**.
 - c. In the **role_ARN** field, paste the `role_arn` that you created previously and click **Install**.
3. Create AWS cloud storage using your AWS credentials by entering the following command:

```
$ cat << EOF | oc create -f -
  apiVersion: oadp.openshift.io/v1alpha1
  kind: CloudStorage
  metadata:
    name: ${CLUSTER_NAME}-oadp
    namespace: openshift-adp
  spec:
    creationSecret:
      key: credentials
      name: cloud-credentials
    enableSharedConfig: true
    name: ${CLUSTER_NAME}-oadp
    provider: aws
    region: $REGION
EOF
```

4. Check your application's storage default storage class by entering the following command:

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

Example output

NAME	STATUS	VOLUME	CAPACITY	ACCESS MODES
STORAGECLASS	AGE			
applog	Bound	pvc-351791ae-b6ab-4e8b-88a4-30f73caf5ef8	1Gi	RWO gp3-
csi	4d19h			
mysql	Bound	pvc-16b8e009-a20a-4379-accb-bc81fedd0621	1Gi	RWO gp3-
csi	4d19h			

5. Get the storage class by running the following command:


```
$ oc get storageclass
```

Example output

NAME	PROVISIONER	RECLAIMPOLICY	VOLUMEBINDINGMODE	ALLOWVOLUMEEXPANSION	AGE
gp2	kubernetes.io/aws-efs	Delete	WaitForFirstConsumer	true	4d21h
gp2-csi	ebs.csi.aws.com	Delete	WaitForFirstConsumer	true	4d21h
gp3	ebs.csi.aws.com	Delete	WaitForFirstConsumer	true	4d21h
gp3-csi (default)	ebs.csi.aws.com	Delete	WaitForFirstConsumer	true	4d21h

NOTE

The following storage classes will work:

- gp3-csi
- gp2-csi
- gp3
- gp2

If the application or applications that are being backed up are all using persistent volumes (PVs) with Container Storage Interface (CSI), it is advisable to include the CSI plugin in the OADP DPA configuration.

6. Create the **DataProtectionApplication** resource to configure the connection to the storage where the backups and volume snapshots are stored:
 - a. If you are using only CSI volumes, deploy a Data Protection Application by entering the following command:

```
$ cat << EOF | oc create -f -
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: ${CLUSTER_NAME}-dpa
  namespace: openshift-adp
spec:
  backupImages: true ❶
  features:
    dataMover:
      enable: false
  backupLocations:
  - bucket:
      cloudStorageRef:
        name: ${CLUSTER_NAME}-oadp
      credential:
        key: credentials
```

```

    name: cloud-credentials
    prefix: velero
    default: true
    config:
      region: ${REGION}
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - aws
        - csi
      nodeAgent: ❷
      enable: false
      uploaderType: kopia ❸
EOF

```

- ❶ Set this field to **false** if you do not want to use image backup.
- ❷ See the important note regarding the **nodeAgent** attribute at the end of this procedure.
- ❸ The type of uploader. The built-in Data Mover uses Kopia as the default uploader mechanism regardless of the value of the **uploaderType** field.

- b. If you are using CSI or non-CSI volumes, deploy a Data Protection Application by entering the following command:

```

$ cat << EOF | oc create -f -
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: ${CLUSTER_NAME}-dpa
  namespace: openshift-adp
spec:
  backupImages: true ❶
  features:
    dataMover:
      enable: false
  backupLocations:
    - bucket:
        cloudStorageRef:
          name: ${CLUSTER_NAME}-oadp
        credential:
          key: credentials
          name: cloud-credentials
        prefix: velero
        default: true
        config:
          region: ${REGION}
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - aws
      nodeAgent: ❷

```

```

enable: false
uploaderType: restic
snapshotLocations:
- velero:
  config:
    credentialsFile: /tmp/credentials/openshift-adp/cloud-credentials-credentials 3
    enableSharedConfig: "true" 4
    profile: default 5
    region: ${REGION} 6
    provider: aws
EOF

```

- 1** Set this field to **false** if you do not want to use image backup.
- 2** See the important note regarding the **nodeAgent** attribute at the end of this procedure.
- 3** The **credentialsFile** field is the mounted location of the bucket credential on the pod.
- 4** The **enableSharedConfig** field allows the **snapshotLocations** to share or reuse the credential defined for the bucket.
- 5** Use the profile name set in the AWS credentials file.
- 6** Specify **region** as your AWS region. This must be the same as the cluster region.

You are now ready to back up and restore OpenShift Container Platform applications, as described in *Backing up applications*.

IMPORTANT

If you use OADP 1.2, replace this configuration:

```

nodeAgent:
  enable: false
  uploaderType: restic

```

with the following configuration:

```

restic:
  enable: false

```

If you want to use two different clusters for backing up and restoring, the two clusters must have the same AWS S3 storage names in both the cloud storage CR and the OADP **DataProtectionApplication** configuration.

Additional resources

- [Installing from the software catalog using the web console](#)
- [Backing up applications](#)

5.21.1.3. Backing up workload on OADP AWS STS, with an optional cleanup

5.21.1.3.1. Performing a backup with OADP and AWS STS

The following example **hello-world** application has no persistent volumes (PVs) attached. Perform a backup with OpenShift API for Data Protection (OADP) with Amazon Web Services (AWS) (AWS STS).

Either Data Protection Application (DPA) configuration will work.

1. Create a workload to back up by running the following commands:

```
$ oc create namespace hello-world
```

```
$ oc new-app -n hello-world --image=docker.io/openshift/hello-openshift
```

2. Expose the route by running the following command:

```
$ oc expose service/hello-openshift -n hello-world
```

3. Check that the application is working by running the following command:

```
$ curl `oc get route/hello-openshift -n hello-world -o jsonpath='{.spec.host}'`
```

Example output

```
Hello OpenShift!
```

4. Back up the workload by running the following command:

```
$ cat << EOF | oc create -f -
  apiVersion: velero.io/v1
  kind: Backup
  metadata:
    name: hello-world
    namespace: openshift-adp
  spec:
    includedNamespaces:
      - hello-world
    storageLocation: ${CLUSTER_NAME}-dpa-1
    ttl: 720h0m0s
EOF
```

5. Wait until the backup has completed and then run the following command:

```
$ watch "oc -n openshift-adp get backup hello-world -o json | jq .status"
```

Example output

```
{
  "completionTimestamp": "2022-09-07T22:20:44Z",
  "expiration": "2022-10-07T22:20:22Z",
  "formatVersion": "1.1.0",
```

```

    "phase": "Completed",
    "progress": {
      "itemsBackedUp": 58,
      "totalItems": 58
    },
    "startTimestamp": "2022-09-07T22:20:22Z",
    "version": 1
  }

```

6. Delete the demo workload by running the following command:

```
$ oc delete ns hello-world
```

7. Restore the workload from the backup by running the following command:

```

$ cat << EOF | oc create -f -
  apiVersion: velero.io/v1
  kind: Restore
  metadata:
    name: hello-world
    namespace: openshift-adp
  spec:
    backupName: hello-world
EOF

```

8. Wait for the Restore to finish by running the following command:

```
$ watch "oc -n openshift-adp get restore hello-world -o json | jq .status"
```

Example output

```

{
  "completionTimestamp": "2022-09-07T22:25:47Z",
  "phase": "Completed",
  "progress": {
    "itemsRestored": 38,
    "totalItems": 38
  },
  "startTimestamp": "2022-09-07T22:25:28Z",
  "warnings": 9
}

```

9. Check that the workload is restored by running the following command:

```
$ oc -n hello-world get pods
```

Example output

NAME	READY	STATUS	RESTARTS	AGE
hello-openshift-9f885f7c6-kdjpj	1/1	Running	0	90s

10. Check the JSONPath by running the following command:

```
$ curl `oc get route/hello-openshift -n hello-world -o jsonpath='{.spec.host}'`
```

Example output

```
Hello OpenShift!
```



NOTE

For troubleshooting tips, see the OADP team's [troubleshooting documentation](#).

5.21.1.3.2. Cleaning up a cluster after a backup with OADP and AWS STS

If you need to uninstall the OpenShift API for Data Protection (OADP) Operator together with the backups and the S3 bucket from this example, follow these instructions.

Procedure

1. Delete the workload by running the following command:

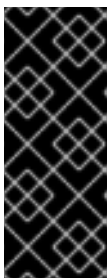
```
$ oc delete ns hello-world
```

2. Delete the Data Protection Application (DPA) by running the following command:

```
$ oc -n openshift-adp delete dpa ${CLUSTER_NAME}-dpa
```

3. Delete the cloud storage by running the following command:

```
$ oc -n openshift-adp delete cloudstorage ${CLUSTER_NAME}-oadp
```



IMPORTANT

If this command hangs, you might need to delete the finalizer by running the following command:

```
$ oc -n openshift-adp patch cloudstorage ${CLUSTER_NAME}-oadp -p '{"metadata":{"finalizers":null}}' --type=merge
```

4. If the Operator is no longer required, remove it by running the following command:

```
$ oc -n openshift-adp delete subscription oadp-operator
```

5. Remove the namespace from the Operator by running the following command:

```
$ oc delete ns openshift-adp
```

6. If the backup and restore resources are no longer required, remove them from the cluster by running the following command:

```
$ oc delete backups.velero.io hello-world
```

- To delete backup, restore and remote objects in AWS S3, run the following command:

```
$ velero backup delete hello-world
```

- If you no longer need the Custom Resource Definitions (CRD), remove them from the cluster by running the following command:

```
$ for CRD in `oc get crds | grep velero | awk '{print $1}'`; do oc delete crd $CRD; done
```

- Delete the AWS S3 bucket by running the following commands:

```
$ aws s3 rm s3://${CLUSTER_NAME}-oadp --recursive
```

```
$ aws s3api delete-bucket --bucket ${CLUSTER_NAME}-oadp
```

- Detach the policy from the role by running the following command:

```
$ aws iam detach-role-policy --role-name "${ROLE_NAME}" --policy-arn "${POLICY_ARN}"
```

- Delete the role by running the following command:

```
$ aws iam delete-role --role-name "${ROLE_NAME}"
```

5.22. OADP AND 3SCALE

5.22.1. Backing up and restoring 3scale API Management by using OADP

With Red Hat 3scale API Management, you can manage your APIs for internal or external users. You can deploy 3scale components on-premise, in the cloud, as a managed service, or in any combination based on your requirements.

With OpenShift API for Data Protection (OADP), you can safeguard 3scale API Management deployments by backing up application resources, persistent volumes, and configurations.



NOTE

You can use the OpenShift API for Data Protection (OADP) Operator to back up and restore your 3scale API Management on-cluster storage databases without affecting your running services

You can configure OADP to perform the following operations with 3scale API Management:

- Create a backup of 3scale components by following the steps in [Backing up 3scale API Management](#).
- Restore the components to scale up the 3scale operator and deployment by following the steps in [Restoring 3scale API Management](#).

5.22.2. Backing up 3scale API Management by using OADP

You can back up Red Hat 3scale API Management components by backing up the 3scale operator, and databases such as MySQL and Redis.

Prerequisites

- You installed and configured Red Hat 3scale API Management. For more information, see [Installing 3scale API Management on OpenShift](#) and [Red Hat 3scale API Management](#).

5.22.2.1. Creating the Data Protection Application

You can create a Data Protection Application (DPA) custom resource (CR) for Red Hat 3scale API Management.

Procedure

- Create a YAML file with the following configuration:

Example dpa.yaml file

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: dpa-sample
  namespace: openshift-adp
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift
        - aws
        - csi
      resourceTimeout: 10m
    nodeAgent:
      enable: true
      uploaderType: kopia
  backupLocations:
    - name: default
      velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket_name> 1
          prefix: <prefix> 2
          config:
            region: <region> 3
            profile: "default"
            s3ForcePathStyle: "true"
            s3Url: <s3_url> 4
          credential:
            key: cloud
            name: cloud-credentials
```

- Specify a bucket as the backup storage location. If the bucket is not a dedicated bucket for Velero backups, you must specify a prefix.

- 2 Specify a prefix for Velero backups, for example, `velero`, if the bucket is used for multiple purposes.
- 3 Specify a region for backup storage location.
- 4 Specify the URL of the object store that you are using to store backups.

2. Create the DPA CR by running the following command:

```
$ oc create -f dpa.yaml
```

Additional resources

- [Installing the Data Protection Application](#)

5.22.2.2. Backing up the 3scale API Management operator, secret, and APIManager

You can back up the Red Hat 3scale API Management operator resources, and both the **Secret** and APIManager custom resource (CR).

Prerequisites

- You created the Data Protection Application (DPA).

Procedure

1. Back up your 3scale operator CRs, such as **operatorgroup**, **namespaces**, and **subscriptions**, by creating a YAML file with the following configuration:

Example backup.yaml file

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: operator-install-backup 1
  namespace: openshift-adp
spec:
  csiSnapshotTimeout: 10m0s
  defaultVolumesToFsBackup: false
  includedNamespaces:
    - threescale 2
  includedResources:
    - operatorgroups
    - subscriptions
    - namespaces
  itemOperationTimeout: 1h0m0s
  snapshotMoveData: false
  ttl: 720h0m0s
```

- 1 The value of the **metadata.name** parameter in the backup is the same value used in the **metadata.backupName** parameter used when restoring the 3scale operator.
- 2 Namespace where the 3scale operator is installed.

**NOTE**

You can also back up and restore **ReplicationControllers**, **Deployment**, and **Pod** objects to ensure that all manually set environments are backed up and restored. This does not affect the flow of restoration.

2. Create a backup CR by running the following command:

```
$ oc create -f backup.yaml
```

Example output

```
backup.velero.io/operator-install-backup created
```

3. Back up the **Secret** CR by creating a YAML file with the following configuration:

Example backup-secret.yaml file

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: operator-resources-secrets 1
  namespace: openshift-adp
spec:
  csiSnapshotTimeout: 10m0s
  defaultVolumesToFsBackup: false
  includedNamespaces:
    - threescale
  includedResources:
    - secrets
  itemOperationTimeout: 1h0m0s
  labelSelector:
    matchLabels:
      app: 3scale-api-management
  snapshotMoveData: false
  snapshotVolumes: false
  ttl: 720h0m0s
```

- 1 The value of the **metadata.name** parameter in the backup is the same value used in the **metadata.backupName** parameter used when restoring the **Secret**.

4. Create the **Secret** backup CR by running the following command:

```
$ oc create -f backup-secret.yaml
```

Example output

```
backup.velero.io/operator-resources-secrets created
```

5. Back up the APIManager CR by creating a YAML file with the following configuration:

Example backup-apimanager.yaml file

```

apiVersion: velero.io/v1
kind: Backup
metadata:
  name: operator-resources-apim 1
  namespace: openshift-adp
spec:
  csiSnapshotTimeout: 10m0s
  defaultVolumesToFsBackup: false
  includedNamespaces:
    - threescale
  includedResources:
    - apimanagers
  itemOperationTimeout: 1h0m0s
  snapshotMoveData: false
  snapshotVolumes: false
  storageLocation: ts-dpa-1
  ttl: 720h0m0s
  volumeSnapshotLocations:
    - ts-dpa-1

```

- 1** The value of the **metadata.name** parameter in the backup is the same value used in the **metadata.backupName** parameter used when restoring the APIManager.

6. Create the APIManager CR by running the following command:

```
$ oc create -f backup-apimanager.yaml
```

Example output

```
backup.velero.io/operator-resources-apim created
```

Additional resources

- [Creating a Backup CR](#)

5.22.2.3. Backing up a MySQL database

You can back up a MySQL database by creating and attaching a persistent volume claim (PVC) to include the dumped data in the specified path.

Prerequisites

- You have backed up the Red Hat 3scale API Management operator.

Procedure

1. Create a YAML file with the following configuration for adding an additional PVC:

Example ts_pvc.yaml file

```

kind: PersistentVolumeClaim
apiVersion: v1

```

```

metadata:
  name: example-claim
  namespace: threescale
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: gp3-csi
  volumeMode: Filesystem

```

2. Create the additional PVC by running the following command:

```
$ oc create -f ts_pvc.yml
```

3. Attach the PVC to the system database pod by editing the **system-mysql** deployment to use the MySQL dump:

```
$ oc edit deployment system-mysql -n threescale
```

```

volumeMounts:
  - name: example-claim
    mountPath: /var/lib/mysqldump/data
  - name: mysql-storage
    mountPath: /var/lib/mysql/data
  - name: mysql-extra-conf
    mountPath: /etc/my-extra.d
  - name: mysql-main-conf
    mountPath: /etc/my-extra
  ...
  serviceAccount: amp
volumes:
  - name: example-claim
    persistentVolumeClaim:
      claimName: example-claim 1
  ...

```

- 1** The PVC that contains the dumped data.

4. Create a YAML file with following configuration to back up the MySQL database:

Example mysql.yaml file

```

apiVersion: velero.io/v1
kind: Backup
metadata:
  name: mysql-backup 1
  namespace: openshift-adp
spec:
  csiSnapshotTimeout: 10m0s
  defaultVolumesToFsBackup: true
  hooks:

```

```

resources:
- name: dumpdb
  pre:
  - exec:
      command:
      - /bin/sh
      - -c
      - mysqldump -u $MYSQL_USER --password=$MYSQL_PASSWORD system --no-
tablespaces
      > /var/lib/mysqldump/data/dump.sql ❷
    container: system-mysql
    onError: Fail
    timeout: 5m
  includedNamespaces:
  - threescale
  includedResources: ❸
  - deployment
  - pods
  - replicationControllers
  - persistentvolumeclaims
  - persistentvolumes
  itemOperationTimeout: 1h0m0s
  labelSelector:
    matchLabels:
      app: 3scale-api-management
      threescale_component_element: mysql
  snapshotMoveData: false
  ttl: 720h0m0s

```

- ❶ The value of the **metadata.name** parameter in the backup is the same value used in the **metadata.backupName** parameter used when restoring the MySQL database.
- ❷ A directory where the data is backed up.
- ❸ Resources to back up.

5. Back up the MySQL database by running the following command:

```
$ oc create -f mysql.yaml
```

Example output

```
backup.velero.io/mysql-backup created
```

Verification

- Verify that the MySQL backup is completed by running the following command:

```
$ oc get backups.velero.io mysql-backup -o yaml
```

Example output

```
status:
```

```

completionTimestamp: "2025-04-17T13:25:19Z"
errors: 1
expiration: "2025-05-17T13:25:16Z"
formatVersion: 1.1.0
hookStatus: {}
phase: Completed
progress: {}
startTimestamp: "2025-04-17T13:25:16Z"
version: 1

```

5.22.2.4. Backing up the back-end Redis database

You can back up the Redis database by adding the required annotations and by listing which resources to back up using the **includedResources** parameter.

Prerequisites

- You backed up the Red Hat 3scale API Management operator.
- You backed up your MySQL database.
- The Redis queues have been drained before performing the backup.

Procedure

1. Edit the annotations on the **backend-redis** deployment by running the following command:

```
$ oc edit deployment backend-redis -n threescale
```

```

annotations:
post.hook.backup.velero.io/command: >-
  ["/bin/bash", "-c", "redis-cli CONFIG SET auto-aof-rewrite-percentage
  100"]
pre.hook.backup.velero.io/command: >-
  ["/bin/bash", "-c", "redis-cli CONFIG SET auto-aof-rewrite-percentage
  0"]

```

2. Create a YAML file with the following configuration to back up the Redis database:

Example redis-backup.yaml file

```

apiVersion: velero.io/v1
kind: Backup
metadata:
  name: redis-backup 1
  namespace: openshift-adp
spec:
  csiSnapshotTimeout: 10m0s
  defaultVolumesToFsBackup: true
  includedNamespaces:
  - threescale
  includedResources:
  - deployment
  - pods

```

```

- replicationcontrollers
- persistentvolumes
- persistentvolumeclaims
itemOperationTimeout: 1h0m0s
labelSelector:
  matchLabels:
    app: 3scale-api-management
    threescale_component: backend
    threescale_component_element: redis
snapshotMoveData: false
snapshotVolumes: false
ttl: 720h0m0s

```

- 1 The value of the **metadata.name** parameter in the backup is the same value used in the **metadata.backupName** parameter used when restoring the Redis database.

3. Back up the Redis database by running the following command:

```
$ oc create -f redis-backup.yaml
```

Example output

```
backup.velero.io/redis-backup created
```

Verification

- Verify that the Redis backup is completed by running the following command:

```
$ oc get backups.velero.io redis-backup -o yaml
```

Example output

```

status:
  completionTimestamp: "2025-04-17T13:25:19Z"
  errors: 1
  expiration: "2025-05-17T13:25:16Z"
  formatVersion: 1.1.0
  hookStatus: {}
  phase: Completed
  progress: {}
  startTimestamp: "2025-04-17T13:25:16Z"
  version: 1

```

5.22.3. Restoring 3scale API Management by using OADP

You can restore Red Hat 3scale API Management components by restoring the backed up 3scale operator resources. You can also restore databases such as MySQL and Redis.

After the data has been restored, you can scale up the 3scale operator and deployment.

Prerequisites

- You installed and configured Red Hat 3scale API Management. For more information, see [Installing 3scale API Management on OpenShift](#) and [Red Hat 3scale API Management](#).
- You backed up the 3scale operator, and databases such as MySQL and Redis.
- Ensure that you are restoring 3scale on the same cluster where it was backed up from.
- If you want to restore 3scale on a different cluster, ensure that the original backed-up cluster and the cluster you want to restore the operator on are using the same custom domain.

5.22.3.1. Restoring the 3scale API Management operator, secrets, and APIManager

You can restore the Red Hat 3scale API Management operator resources, and both the **Secret** and APIManager custom resources (CRs) by using the following procedure.

Prerequisites

- You backed up the 3scale operator.
- You backed up the MySQL and Redis databases.
- You are restoring the database on the same cluster, where it was backed up.
If you are restoring the operator to a different cluster that you backed up from, install and configure OADP with **nodeAgent** enabled on the destination cluster. Ensure that the OADP configuration is same as it was on the source cluster.

Procedure

1. Delete the 3scale operator custom resource definitions (CRDs) along with the **threescale** namespace by running the following command:

```
$ oc delete project threescale
```

Example output

```
"threescale" project deleted successfully
```

2. Create a YAML file with the following configuration to restore the 3scale operator:

Example restore.yaml file

```
apiVersion: velero.io/v1
kind: Restore
metadata:
  name: operator-installation-restore
  namespace: openshift-adp
spec:
  backupName: operator-install-backup 1
  excludedResources:
    - nodes
    - events
    - events.events.k8s.io
    - backups.velero.io
    - restores.velero.io
```



```
- resticrepositories.velero.io
- csinodes.storage.k8s.io
- volumeattachments.storage.k8s.io
- backuprepositories.velero.io
itemOperationTimeout: 4h0m0s
```

1 Restoring the 3scale operator's backup

3. Restore the 3scale operator by running the following command:

```
$ oc create -f restore.yaml
```

Example output

```
restore.velerio.io/operator-installation-restore created
```

4. Manually create the **s3-credentials Secret** object by running the following command:

```
$ oc apply -f - <<EOF
---
apiVersion: v1
kind: Secret
metadata:
  name: s3-credentials
  namespace: threescale
stringData:
  AWS_ACCESS_KEY_ID: <ID_123456> 1
  AWS_SECRET_ACCESS_KEY: <ID_98765544> 2
  AWS_BUCKET: <mybucket.example.com> 3
  AWS_REGION: <us-east-1> 4
type: Opaque
EOF
```

- 1 Replace <ID_123456> with your AWS credentials ID.
- 2 Replace <ID_98765544> with your AWS credentials KEY.
- 3 Replace <mybucket.example.com> with your target bucket name.
- 4 Replace <us-east-1> with the AWS region of your bucket.

5. Scale down the 3scale operator by running the following command:

```
$ oc scale deployment threescale-operator-controller-manager-v2 --replicas=0 -n threescale
```

Example output

```
deployment.apps/threescale-operator-controller-manager-v2 scaled
```

6. Create a YAML file with the following configuration to restore the **Secret**:

Example restore-secret.yaml file

```

apiVersion: velero.io/v1
kind: Restore
metadata:
  name: operator-resources-secrets
  namespace: openshift-adp
spec:
  backupName: operator-resources-secrets 1
  excludedResources:
    - nodes
    - events
    - events.events.k8s.io
    - backups.velero.io
    - restores.velero.io
    - resticrepositories.velero.io
    - csinodes.storage.k8s.io
    - volumeattachments.storage.k8s.io
    - backuprepositories.velero.io
  itemOperationTimeout: 4h0m0s

```

- 1** Restoring the **Secret** backup.

7. Restore the **Secret** by running the following command:

```
$ oc create -f restore-secrets.yaml
```

Example output

```
restore.velero.io/operator-resources-secrets created
```

8. Create a YAML file with the following configuration to restore APIManager:

Example restore-apimanager.yaml file

```

apiVersion: velero.io/v1
kind: Restore
metadata:
  name: operator-resources-apim
  namespace: openshift-adp
spec:
  backupName: operator-resources-apim 1
  excludedResources: 2
    - nodes
    - events
    - events.events.k8s.io
    - backups.velero.io
    - restores.velero.io
    - resticrepositories.velero.io
    - csinodes.storage.k8s.io
    - volumeattachments.storage.k8s.io
    - backuprepositories.velero.io
  itemOperationTimeout: 4h0m0s

```

■

- 1 Restoring the APIManager backup.
- 2 The resources that you do not want to restore.

9. Restore the APIManager by running the following command:

```
$ oc create -f restore-apimanager.yaml
```

Example output

```
restore.velerio.io/operator-resources-apim created
```

10. Scale up the 3scale operator by running the following command:

```
$ oc scale deployment threescale-operator-controller-manager-v2 --replicas=1 -n threescale
```

Example output

```
deployment.apps/threescale-operator-controller-manager-v2 scaled
```

5.22.3.2. Restoring a MySQL database

Restoring a MySQL database re-creates the following resources:

- The **Pod**, **ReplicationController**, and **Deployment** objects.
- The additional persistent volumes (PVs) and associated persistent volume claims (PVCs).
- The MySQL dump, which the **example-claim** PVC contains.



WARNING

Do not delete the default PV and PVC associated with the database. If you do, your backups are deleted.

Prerequisites

- You restored the **Secret** and APIManager custom resources (CRs).

Procedure

1. Scale down the Red Hat 3scale API Management operator by running the following command:

```
$ oc scale deployment threescale-operator-controller-manager-v2 --replicas=0 -n threescale
```

Example output

```
deployment.apps/threescale-operator-controller-manager-v2 scaled
```

2. Create the following script to scale down the 3scale operator:

```
$ vi ./scaledowndeployment.sh
```

Example script:

```
for deployment in apicast-production apicast-staging backend-cron backend-listener backend-redis backend-worker system-app system-memcache system-mysql system-redis system-searchd system-sidekiq zync zync-database zync-que; do
  oc scale deployment/$deployment --replicas=0 -n threescale
done
```

3. Scale down all the deployment 3scale components by running the following script:

```
$ ./scaledowndeployment.sh
```

Example output

```
deployment.apps.openshift.io/apicast-production scaled
deployment.apps.openshift.io/apicast-staging scaled
deployment.apps.openshift.io/backend-cron scaled
deployment.apps.openshift.io/backend-listener scaled
deployment.apps.openshift.io/backend-redis scaled
deployment.apps.openshift.io/backend-worker scaled
deployment.apps.openshift.io/system-app scaled
deployment.apps.openshift.io/system-memcache scaled
deployment.apps.openshift.io/system-mysql scaled
deployment.apps.openshift.io/system-redis scaled
deployment.apps.openshift.io/system-searchd scaled
deployment.apps.openshift.io/system-sidekiq scaled
deployment.apps.openshift.io/zync scaled
deployment.apps.openshift.io/zync-database scaled
deployment.apps.openshift.io/zync-que scaled
```

4. Delete the **system-mysql Deployment** object by running the following command:

```
$ oc delete deployment system-mysql -n threescale
```

Example output

```
Warning: apps.openshift.io/v1 deployment is deprecated in v4.14+, unavailable in v4.10000+
deployment.apps.openshift.io "system-mysql" deleted
```

5. Create the following YAML file to restore the MySQL database:

Example restore-mysql.yaml file

```
apiVersion: velero.io/v1
kind: Restore
metadata:
```

```

name: restore-mysql
namespace: openshift-adp
spec:
  backupName: mysql-backup ❶
  excludedResources:
    - nodes
    - events
    - events.events.k8s.io
    - backups.velero.io
    - restores.velero.io
    - csinodes.storage.k8s.io
    - volumeattachments.storage.k8s.io
    - backuprepositories.velero.io
    - resticrepositories.velero.io
  hooks:
    resources:
      - name: restoreDB
        postHooks:
          - exec:
              command:
                - /bin/sh
                - '-c'
                - >
                  sleep 30

                  mysql -h 127.0.0.1 -D system -u root
                  --password=$MYSQL_ROOT_PASSWORD <
                  /var/lib/mysqldump/data/dump.sql ❷
            container: system-mysql
            execTimeout: 80s
            onError: Fail
            waitTimeout: 5m
  itemOperationTimeout: 1h0m0s
  restorePVs: true

```

❶ Restoring the MySQL backup.

❷ A path where the data is restored from.

6. Restore the MySQL database by running the following command:

```
$ oc create -f restore-mysql.yaml
```

Example output

```
restore.velero.io/restore-mysql created
```

Verification

1. Verify that the **PodVolumeRestore** restore is completed by running the following command:

```
$ oc get podvolumerestores.velero.io -n openshift-adp
```

Example output

Example output

NAME	NAMESPACE	POD	UPLOADER TYPE	VOLUME
restore-mysql-rbzym	threescale	system-mysql-2-kjkh	kopia	mysql-storage
Completed	771879108	771879108	40m	
restore-mysql-z7x7l	threescale	system-mysql-2-kjkh	kopia	example-claim
Completed	380415	380415	40m	

2. Verify that the additional PVC has been restored by running the following command:

```
$ oc get pvc -n threescale
```

Example output

NAME	STATUS	VOLUME	CAPACITY	ACCESS MODES
backend-redis-storage	Bound	pvc-3dca410d-3b9f-49d4-aebf-75f47152e09d	1Gi	
RWO	gp3-csi	<unset>	68m	
example-claim	Bound	pvc-cbaa49b0-06cd-4b1a-9e90-0ef755c67a54	1Gi	RWO
gp3-csi	<unset>	57m		
mysql-storage	Bound	pvc-4549649f-b9ad-44f7-8f67-dd6b9dbb3896	1Gi	RWO
gp3-csi	<unset>	68m		
system-redis-storage	Bound	pvc-04dadafd-8a3e-4d00-8381-6041800a24fc	1Gi	
RWO	gp3-csi	<unset>	68m	
system-searchd	Bound	pvc-afbf606c-d4a8-4041-8ec6-54c5baf1a3b9	1Gi	RWO
gp3-csi	<unset>	68m		

5.22.3.3. Restoring the back-end Redis database

You can restore the back-end Redis database by deleting the deployment and specifying which resources you do not want to restore.

Prerequisites

- You restored the Red Hat 3scale API Management operator resources, **Secret**, and APIManager custom resources.
- You restored the MySQL database.

Procedure

1. Delete the **backend-redis** deployment by running the following command:

```
$ oc delete deployment backend-redis -n threescale
```

Example output

```
Warning: apps.openshift.io/v1 deployment is deprecated in v4.14+, unavailable in v4.10000+
deployment.apps.openshift.io "backend-redis" deleted
```

2. Create a YAML file with the following configuration to restore the Redis database:

Example restore-backend.yaml file

```

apiVersion: velero.io/v1
kind: Restore
metadata:
  name: restore-backend
  namespace: openshift-adp
spec:
  backupName: redis-backup ❶
  excludedResources:
    - nodes
    - events
    - events.events.k8s.io
    - backups.velero.io
    - restores.velero.io
    - resticrepositories.velero.io
    - csinodes.storage.k8s.io
    - volumeattachments.storage.k8s.io
    - backuprepositories.velero.io
  itemOperationTimeout: 1h0m0s
  restorePVs: true

```

❶ Restoring the Redis backup.

3. Restore the Redis database by running the following command:

```
$ oc create -f restore-backend.yaml
```

Example output

```
restore.velero.io/restore-backend created
```

Verification

- Verify that the **PodVolumeRestore** restore is completed by running the following command:

```
$ oc get podvolumerestores.velero.io -n openshift-adp
```

Example output:

NAME	NAMESPACE	POD	UPLOADER	TYPE	VOLUME
restore-backend-jmrwx	threescale	backend-redis-1-bsfmv	kopia		backend-redis-
storage	Completed	76123	76123	21m	

5.22.3.4. Scaling up the 3scale API Management operator and deployment

You can scale up the Red Hat 3scale API Management operator and any deployment that was manually scaled down. After a few minutes, 3scale installation should be fully functional, and its state should match the backed-up state.

Prerequisites

- You restored the 3scale operator resources, and both the **Secret** and APIManager custom resources (CRs).
- You restored the MySQL and back-end Redis databases.
- Ensure that there are no scaled up deployments or no extra pods running. There might be some **system-mysql** or **backend-redis** pods running detached from deployments after restoration, which can be removed after the restoration is successful.

Procedure

1. Scale up the 3scale operator by running the following command:

```
$ oc scale deployment threescale-operator-controller-manager-v2 --replicas=1 -n threescale
```

Example output

```
deployment.apps/threescale-operator-controller-manager-v2 scaled
```

2. Ensure that the 3scale pod is running to verify if the 3scale operator was deployed by running the following command:

```
$ oc get pods -n threescale
```

Example output

NAME	READY	STATUS	RESTARTS	AGE
threescale-operator-controller-manager-v2-79546bd8c-b4qbh	1/1	Running	0	2m5s

3. Create the following script to scale up the deployments:

```
$ vi ./scaledeployment.sh
```

Example script file:

```
for deployment in apicast-production apicast-staging backend-cron backend-listener backend-redis backend-worker system-app system-memcache system-mysql system-redis system-searchd system-sidekiq zync zync-database zync-que; do
  oc scale deployment/$deployment --replicas=1 -n threescale
done
```

4. Scale up the deployments by running the following script:

```
$ ./scaledeployment.sh
```

Example output

```
deployment.apps.openshift.io/apicast-production scaled
deployment.apps.openshift.io/apicast-staging scaled
```



```

deployment.apps.openshift.io/backend-cron scaled
deployment.apps.openshift.io/backend-listener scaled
deployment.apps.openshift.io/backend-redis scaled
deployment.apps.openshift.io/backend-worker scaled
deployment.apps.openshift.io/system-app scaled
deployment.apps.openshift.io/system-memcache scaled
deployment.apps.openshift.io/system-mysql scaled
deployment.apps.openshift.io/system-redis scaled
deployment.apps.openshift.io/system-searchd scaled
deployment.apps.openshift.io/system-sidekiq scaled
deployment.apps.openshift.io/zync scaled
deployment.apps.openshift.io/zync-database scaled
deployment.apps.openshift.io/zync-queue scaled

```

5. Get the **3scale-admin** route to log in to the 3scale UI by running the following command:

```
$ oc get routes -n threescale
```

Example output

NAME	HOST/PORT	PATH
SERVICES	PORT	TERMINATION WILDCARD
backend	backend-3scale.apps.custom-cluster-name.openshift.com	
backend-listener	http edge/Allow	None
zync-3scale-api-b4l4d	api-3scale-apicast-production.apps.custom-cluster-	
name.openshift.com	apicast-production gateway edge/Redirect	None
zync-3scale-api-b6sns	api-3scale-apicast-staging.apps.custom-cluster-	
name.openshift.com	apicast-staging gateway edge/Redirect	None
zync-3scale-master-7sc4j	master.apps.custom-cluster-name.openshift.com	
system-master	http edge/Redirect	None
zync-3scale-provider-7r2nm	3scale-admin.apps.custom-cluster-name.openshift.com	
system-provider	http edge/Redirect	None
zync-3scale-provider-mjxlb	3scale.apps.custom-cluster-name.openshift.com	
system-developer	http edge/Redirect	None

In this example, **3scale-admin.apps.custom-cluster-name.openshift.com** is the 3scale-admin URL.

6. Use the URL from this output to log in to the 3scale operator as an administrator. You can verify that the data, when you took backup, is available.

5.23. OADP DATA MOVER

5.23.1. About the OADP Data Mover

Use the OpenShift API for Data Protection (OADP) built-in Data Mover to move Container Storage Interface (CSI) volume snapshots to remote object storage and restore stateful applications after cluster failures. This provides disaster recovery capabilities for both containerized and virtual machine workloads.

The Data Mover uses [Kopia](#) as the uploader mechanism to read the snapshot data and write to the unified repository.

OADP supports CSI snapshots on the following:

- Red Hat OpenShift Data Foundation
- Any other cloud storage provider with the Container Storage Interface (CSI) driver that supports the Kubernetes Volume Snapshot API

5.23.1.1. Data Mover support

Review Data Mover support and compatibility across OADP versions to understand which backups can be restored. This helps you plan version upgrades and backup strategies.

The OADP built-in Data Mover, which was introduced in OADP 1.3 as a Technology Preview, is now fully supported for both containerized and virtual machine workloads.

Supported

The Data Mover backups taken with OADP 1.3 can be restored using OADP 1.3 and later.

Not supported

Backups taken with OADP 1.1 or OADP 1.2 using the Data Mover feature cannot be restored using OADP 1.3 and later.

OADP 1.1 and OADP 1.2 are no longer supported. The DataMover feature in OADP 1.1 or OADP 1.2 was a Technology Preview and was never supported. DataMover backups taken with OADP 1.1 or OADP 1.2 cannot be restored on later versions of OADP.

5.23.1.2. Enabling the built-in Data Mover

Enable the built-in Data Mover by configuring the CSI plugin and node agent in the **DataProtectionApplication** custom resource (CR). This provides volume-level backup and restore operations by using the Kopia uploader.

Procedure

- Include the CSI plugin and enable the node agent in the **DataProtectionApplication** custom resource (CR) as shown in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: dpa-sample
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
  velero:
    defaultPlugins:
      - openshift
      - aws
      - csi
    defaultSnapshotMoveData: true
    defaultVolumesToFSBackup:
    featureFlags:
      - EnableCSI
# ...
```

where:

enable

Specifies the flag to enable the node agent.

uploaderType

Specifies the type of uploader. The possible values are **restic** or **kopia**. The built-in Data Mover uses Kopia as the default uploader mechanism regardless of the value of the **uploaderType** field.

csi

Specifies the CSI plugin included in the list of default plugins.

defaultVolumesToFSBackup

Specifies the default behavior for volumes. In OADP 1.3.1 and later, set to **true** if you use Data Mover only for volumes that opt out of **fs-backup**. Set to **false** if you use Data Mover by default for volumes.

5.23.1.3. Built-in Data Mover controller and custom resource definitions (CRDs)

Review the custom resource definitions (CRDs) that the built-in Data Mover uses to manage volume snapshot backup and restore operations. This helps you understand how Data Mover handles data upload, download, and repository management.

The built-in Data Mover feature introduces three new API objects defined as CRDs for managing backup and restore:

- **DataDownload:** Represents a data download of a volume snapshot. The CSI plugin creates one **DataDownload** object per volume to be restored. The **DataDownload** CR includes information about the target volume, the specified Data Mover, the progress of the current data download, the specified backup repository, and the result of the current data download after the process is complete.
- **DataUpload:** Represents a data upload of a volume snapshot. The CSI plugin creates one **DataUpload** object per CSI snapshot. The **DataUpload** CR includes information about the specified snapshot, the specified Data Mover, the specified backup repository, the progress of the current data upload, and the result of the current data upload after the process is complete.
- **BackupRepository:** Represents and manages the lifecycle of the backup repositories. OADP creates a backup repository per namespace when the first CSI snapshot backup or restore for a namespace is requested.

5.23.1.4. About incremental back up support

OADP supports incremental backups of **block** and **Filesystem** persistent volumes for both containerized, and OpenShift Virtualization workloads. The following table summarizes the support for File System Backup (FSB), Container Storage Interface (CSI), and CSI Data Mover:

Table 5.9. OADP backup support matrix for containerized workloads

Volume mode	FSB - Restic	FSB - Kopia	CSI	CSI Data Mover
Filesystem	S ^[1] , I ^[2]	S ^[1] , I ^[2]	S ^[1]	S ^[1] , I ^[2]

Volume mode	FSB - Restic	FSB - Kopia	CSI	CSI Data Mover
Block	N ^[3]	N ^[3]	S ^[1]	S ^[1] , I ^[2]

Table 5.10. OADP backup support matrix for OpenShift Virtualization workloads

Volume mode	FSB - Restic	FSB - Kopia	CSI	CSI Data Mover
Filesystem	N ^[3]	N ^[3]	S ^[1]	S ^[1] , I ^[2]
Block	N ^[3]	N ^[3]	S ^[1]	S ^[1] , I ^[2]

1. Backup supported
2. Incremental backup supported
3. Not supported

**NOTE**

The CSI Data Mover backups use Kopia regardless of **uploaderType**.

5.23.2. Backing up and restoring CSI snapshots data movement

You can back up and restore persistent volumes by using the OADP 1.3 Data Mover.

5.23.2.1. Backing up persistent volumes with CSI snapshots

You can use the OADP Data Mover to back up Container Storage Interface (CSI) volume snapshots to a remote object store.

Prerequisites

- You have access to the cluster with the **cluster-admin** role.
- You have installed the OADP Operator.
- You have included the CSI plugin and enabled the node agent in the **DataProtectionApplication** custom resource (CR).
- You have an application with persistent volumes running in a separate namespace.
- You have added the **metadata.labels.velero.io/csi-volumesnapshot-class: "true"** key-value pair to the **VolumeSnapshotClass** CR.

Procedure

1. Create a YAML file for the **Backup** object, as in the following example:

Example Backup CR

```

kind: Backup
apiVersion: velero.io/v1
metadata:
  name: backup
  namespace: openshift-adp
spec:
  csiSnapshotTimeout: 10m0s
  defaultVolumesToFsBackup: 1
  includedNamespaces:
  - mysql-persistent
  itemOperationTimeout: 4h0m0s
  snapshotMoveData: true 2
  storageLocation: default
  ttl: 720h0m0s
  volumeSnapshotLocations:
  - dpa-sample-1
# ...

```

- 1 Set to **true** if you use Data Mover only for volumes that opt out of **fs-backup**. Set to **false** if you use Data Mover by default for volumes.
- 2 Set to **true** to enable movement of CSI snapshots to remote object storage.



NOTE

If you format the volume by using XFS filesystem and the volume is at 100% capacity, the backup fails with a **no space left on device** error. For example:

```

Error: relabel failed /var/lib/kubelet/pods/3ac..34/volumes/ \
kubernetes.io~csi/pvc-684..12c/mount: lsetxattr /var/lib/kubelet/ \
pods/3ac..34/volumes/kubernetes.io~csi/pvc-68..2c/mount/data-xfs-103: \
no space left on device

```

In this scenario, consider resizing the volume or using a different filesystem type, for example, **ext4**, so that the backup completes successfully.

2. Apply the manifest:

```
$ oc create -f backup.yaml
```

A **DataUpload** CR is created after the snapshot creation is complete.

Verification

- Verify that the snapshot data is successfully transferred to the remote object store by monitoring the **status.phase** field of the **DataUpload** CR. Possible values are **In Progress**, **Completed**, **Failed**, or **Canceled**. The object store is configured in the **backupLocations** stanza of the **DataProtectionApplication** CR.
 - Run the following command to get a list of all **DataUpload** objects:

```
$ oc get datauploads -A
```

Example output

NAMESPACE	NAME	STATUS	STARTED	BYTES DONE	TOTAL BYTES	STORAGE LOCATION	AGE	NODE
openshift-adp	backup-test-1-sw76b	Completed	9m47s	108104082	108104082			
dpa-sample-1	9m47s		ip-10-0-150-57.us-west-2.compute.internal					
openshift-adp	mongo-block-7dtpf	Completed	14m	1073741824	1073741824			
dpa-sample-1	14m		ip-10-0-150-57.us-west-2.compute.internal					

- Check the value of the **status.phase** field of the specific **DataUpload** object by running the following command:

```
$ oc get datauploads <dataupload_name> -o yaml
```

Example output

```
apiVersion: velero.io/v2alpha1
kind: DataUpload
metadata:
  name: backup-test-1-sw76b
  namespace: openshift-adp
spec:
  backupStorageLocation: dpa-sample-1
  csiSnapshot:
    snapshotClass: ""
    storageClass: gp3-csi
    volumeSnapshot: velero-mysql-fq8sl
  operationTimeout: 10m0s
  snapshotType: CSI
  sourceNamespace: mysql-persistent
  sourcePVC: mysql
status:
  completionTimestamp: "2023-11-02T16:57:02Z"
  node: ip-10-0-150-57.us-west-2.compute.internal
  path: /host_pods/15116bac-cc01-4d9b-8ee7-609c3bef6bde/volumes/kubernetes.io~csi/pvc-eead8167-556b-461a-b3ec-441749e291c4/mount
  phase: Completed 1
  progress:
    bytesDone: 108104082
    totalBytes: 108104082
  snapshotID: 8da1c5febf25225f4577ada2aeb9f899
  startTimestamp: "2023-11-02T16:56:22Z"
```

- 1 Indicates that snapshot data is successfully transferred to the remote object store.

5.23.2.2. Restoring CSI volume snapshots

You can restore a volume snapshot by creating a **Restore** CR.



NOTE

You cannot restore Volsync backups from OADP 1.2 with the OADP 1.3 built-in Data Mover. It is recommended to do a file system backup of all of your workloads with Restic before upgrading to OADP 1.3.

Prerequisites

- You have access to the cluster with the **cluster-admin** role.
- You have an OADP **Backup** CR from which to restore the data.

Procedure

1. Create a YAML file for the **Restore** CR, as in the following example:

Example Restore CR

```
apiVersion: velero.io/v1
kind: Restore
metadata:
  name: restore
  namespace: openshift-adp
spec:
  backupName: <backup>
# ...
```

2. Apply the manifest:

```
$ oc create -f restore.yaml
```

A **DataDownload** CR is created when the restore starts.

Verification

- You can monitor the status of the restore process by checking the **status.phase** field of the **DataDownload** CR. Possible values are **In Progress**, **Completed**, **Failed**, or **Canceled**.
 - To get a list of all **DataDownload** objects, run the following command:

```
$ oc get datadownloads -A
```

Example output

```
NAMESPACE   NAME                               STATUS   STARTED  BYTES DONE  TOTAL
BYTES  STORAGE LOCATION  AGE  NODE
openshift-adp  restore-test-1-sk7lg  Completed  7m11s   108104082  108104082
dpa-sample-1   7m11s  ip-10-0-150-57.us-west-2.compute.internal
```

- Enter the following command to check the value of the **status.phase** field of the specific **DataDownload** object:

```
$ oc get datadownloads <datadownload_name> -o yaml
```

Example output

```

apiVersion: velero.io/v2alpha1
kind: DataDownload
metadata:
  name: restore-test-1-sk7lg
  namespace: openshift-adp
spec:
  backupStorageLocation: dpa-sample-1
  operationTimeout: 10m0s
  snapshotID: 8da1c5febf25225f4577ada2aeb9f899
  sourceNamespace: mysql-persistent
  targetVolume:
    namespace: mysql-persistent
    pv: ""
    pvc: mysql
status:
  completionTimestamp: "2023-11-02T17:01:24Z"
  node: ip-10-0-150-57.us-west-2.compute.internal
  phase: Completed 1
  progress:
    bytesDone: 108104082
    totalBytes: 108104082
  startTimestamp: "2023-11-02T17:00:52Z"

```

1 Indicates that the CSI snapshot data is successfully restored.

5.23.2.3. Deletion policy for OADP 1.3

The deletion policy determines rules for removing data from a system, specifying when and how deletion occurs based on factors such as retention periods, data sensitivity, and compliance requirements. It manages data removal effectively while meeting regulations and preserving valuable information.

5.23.2.3.1. Deletion policy guidelines for OADP 1.3

Review the following deletion policy guidelines for the OADP 1.3:

- In OADP 1.3.x, when using any type of backup and restore methods, you can set the **deletionPolicy** field to **Retain** or **Delete** in the **VolumeSnapshotClass** custom resource (CR).

5.23.3. Configuring backup and restore PVCs for Data Mover

Configure backup and restore persistent volume claims (PVCs) to optimize Data Mover operations. For storage classes like CephFS, these intermediate PVCs allow the system to create read-only volumes from snapshots, resulting in significantly faster backups.

You create a **readonly** backup PVC by using the **nodeAgent.backupPVC** section of the **DataProtectionApplication** (DPA) and setting the **readOnly** access mode to **true**.

You can use the following fields in the **nodeAgent.backupPVC** section of the DPA to configure the backup PVC.

- **storageClass**: The name of the storage class to use for the backup PVC.

- **readOnly**: Indicates if the backup PVC should be mounted as read-only. Setting this field to **true** also requires you to set the **spcNoRelabeling** field to **true**.
- **spcNoRelabeling**: Disables automatic relabeling of the volume if set to **true**. You can set this field to **true** only when **readOnly** is **true**. When the **readOnly** flag is **true**, SELinux relabeling of the volume is not possible. This causes the Data Mover backup to fail. Therefore, when you are using the **readOnly** access mode for the CephFS storage class, you must disable relabeling.

5.23.3.1. Configuring a backup PVC for a Data Mover backup

Configure backup persistent volume claim (PVC) settings in the **DataProtectionApplication** (DPA) to optimize Data Mover backup performance for different storage classes. The feature gives you read-only access modes for faster data movement.

Prerequisites

- You have installed the OADP Operator.

Procedure

1. Configure the **nodeAgent.backupPVC** section in the DPA as shown in the following example:

Example Data Protection Application

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: ts-dpa
  namespace: openshift-adp
spec:
  backupLocations:
  - velero:
      credential:
        key: cloud
        name: cloud-credentials-gcp
      default: true
      objectStorage:
        bucket: oadp...2jw
        prefix: velero
        provider: gcp
      configuration:
        nodeAgent:
          enable: true
          uploaderType: kopia
          backupPVC:
            storage-class-1:
              readOnly: true
              spcNoRelabeling: true
              storageClass: gp3-csi
            storage-class-2:
              readOnly: false
              spcNoRelabeling: false
              storageClass: gp3-csi
        velero:
          defaultPlugins:
```

- gcp
- openshift
- csi

where:

backupPVC

Specifies the **backupPVC** section. In this example, the **backupPVC** section has configurations for two storage classes, **storage-class-1** and **storage-class-2**.

readOnly

Specifies that the **backupPVC** for **storage-class-1** is configured as **readOnly**.

spcNoRelabeling

Specifies that the **spcNoRelabeling** field is set to **true** because the **backupPVC** for **storage-class-1** is **readOnly**.

2. Create a **Backup** custom resource by using the following configuration:

Example Backup

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: test-backup
  namespace: openshift-adp
spec:
  includedNamespaces:
    - <application_namespace>
  snapshotMoveData: true
```

snapshotMoveData

Set to **true** for a Data Mover backup.

Verification

- Verify that the backup PVCs are created as read-only (**ROX**) by running the following command:

Example command

```
$ oc get pvc -n openshift-adp -w
```

```
test-backup1-l..d  Bound  pvc-1298.....22f8  2Gi    ROX        standard-csi  <unset>
37s
test-backup1-l..d  Bound  pvc-1298....022f8  2Gi    ROX        standard-csi  <unset>
37s
```

5.23.3.2. Configuring a restorePVC for a Data Mover restore

Configure restore persistent volume claim (PVC) settings in the **DataProtectionApplication** (DPA) to optimize Data Mover restore operations and enable parallel volume restores. This improves restore performance by distributing restore pods across nodes.

A **restorePVC** is an intermediate PVC that is used to write data during the Data Mover restore operation.

You can configure the **restorePVC** in the **DataProtectionApplication** (DPA) object by using the **ignoreDelayBinding** field. Setting the **ignoreDelayBinding** field to **true** allows the restore operation to ignore the **WaitForFirstConsumer** binding mode. The data movement restore operation then creates the restore pod and provisions the associated volume to an arbitrary node.

The **ignoreDelayBinding** setting is helpful in scenarios where multiple volume restores are happening in parallel. With the **ignoreDelayBinding** field set to **true**, the restore pods can be spread evenly to all nodes.

Prerequisites

- You have installed the OADP Operator.
- You have a created a Data Mover backup of an application.

Procedure

- Configure the **restorePVC** section in the DPA as shown in the following example:

Example Data Protection Application

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: ts-dpa
  namespace: openshift-adp
spec:
  # ...
  configuration:
    nodeAgent:
      enable: true
      uploaderType: kopia
    restorePVC:
      ignoreDelayBinding: true
```

where:

restorePVC

Specifies the **restorePVC** section.

ignoreDelayBinding

Set the **ignoreDelayBinding** field to **true**.

5.23.4. Overriding Kopia hashing, encryption, and splitter algorithms

You can override the default values of Kopia hashing, encryption, and splitter algorithms by using specific environment variables in the Data Protection Application (DPA).

5.23.4.1. Configuring the DPA to override Kopia hashing, encryption, and splitter algorithms

You can use an OpenShift API for Data Protection (OADP) option to override the default Kopia

algorithms for hashing, encryption, and splitter to improve Kopia performance or to compare performance metrics. You can set the following environment variables in the **spec.configuration.velero.podConfig.env** section of the DPA:

- **KOPIA_HASHING_ALGORITHM**
- **KOPIA_ENCRYPTION_ALGORITHM**
- **KOPIA_SPLITTER_ALGORITHM**

Prerequisites

- You have installed the OADP Operator.
- You have created the secret by using the credentials provided by the cloud provider.



NOTE

The configuration of the Kopia algorithms for splitting, hashing, and encryption in the Data Protection Application (DPA) apply only during the initial Kopia repository creation, and cannot be changed later.

To use different Kopia algorithms, ensure that the object storage does not contain any previous Kopia repositories of backups. Configure a new object storage in the Backup Storage Location (BSL) or specify a unique prefix for the object storage in the BSL configuration.

Procedure

- Configure the DPA with the environment variables for hashing, encryption, and splitter as shown in the following example.

Example DPA

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
#...
configuration:
  nodeAgent:
    enable: true 1
    uploaderType: kopia 2
  velero:
    defaultPlugins:
      - openshift
      - aws
      - csi 3
    defaultSnapshotMoveData: true
    podConfig:
      env:
        - name: KOPIA_HASHING_ALGORITHM
          value: <hashing_algorithm_name> 4
        - name: KOPIA_ENCRYPTION_ALGORITHM
```

```

value: <encryption_algorithm_name> 5
- name: KOPIA_SPLITTER_ALGORITHM
  value: <splitter_algorithm_name> 6

```

- 1 Enable the **nodeAgent**.
- 2 Specify the **uploaderType** as **kopia**.
- 3 Include the **csi** plugin.
- 4 Specify a hashing algorithm. For example, **BLAKE3-256**.
- 5 Specify an encryption algorithm. For example, **CHACHA20-POLY1305-HMAC-SHA256**.
- 6 Specify a splitter algorithm. For example, **DYNAMIC-8M-RABINKARP**.

5.23.4.2. Use case for overriding Kopia hashing, encryption, and splitter algorithms

The use case example demonstrates taking a backup of an application by using Kopia environment variables for hashing, encryption, and splitter. You store the backup in an AWS S3 bucket. You then verify the environment variables by connecting to the Kopia repository.

Prerequisites

- You have installed the OADP Operator.
- You have an AWS S3 bucket configured as the backup storage location.
- You have created the secret by using the credentials provided by the cloud provider.
- You have installed the Kopia client.
- You have an application with persistent volumes running in a separate namespace.

Procedure

1. Configure the Data Protection Application (DPA) as shown in the following example:

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_name> 1
  namespace: openshift-adp
spec:
  backupLocations:
  - name: aws
    velero:
      config:
        profile: default
        region: <region_name> 2
      credential:
        key: cloud
        name: cloud-credentials 3
      default: true

```

```

objectStorage:
  bucket: <bucket_name> 4
  prefix: velero
  provider: aws
configuration:
  nodeAgent:
    enable: true
    uploaderType: kopia
  velero:
    defaultPlugins:
      - openshift
      - aws
      - csi 5
    defaultSnapshotMoveData: true
  podConfig:
    env:
      - name: KOPIA_HASHING_ALGORITHM
        value: BLAKE3-256 6
      - name: KOPIA_ENCRYPTION_ALGORITHM
        value: CHACHA20-POLY1305-HMAC-SHA256 7
      - name: KOPIA_SPLITTER_ALGORITHM
        value: DYNAMIC-8M-RABINKARP 8

```

- 1 Specify a name for the DPA.
- 2 Specify the region for the backup storage location.
- 3 Specify the name of the default **Secret** object.
- 4 Specify the AWS S3 bucket name.
- 5 Include the **csi** plugin.
- 6 Specify the hashing algorithm as **BLAKE3-256**.
- 7 Specify the encryption algorithm as **CHACHA20-POLY1305-HMAC-SHA256**.
- 8 Specify the splitter algorithm as **DYNAMIC-8M-RABINKARP**.

2. Create the DPA by running the following command:

```
$ oc create -f <dpa_file_name> 1
```

- 1 Specify the file name of the DPA you configured.

3. Verify that the DPA has reconciled by running the following command:

```
$ oc get dpa -o yaml
```

4. Create a backup CR as shown in the following example:

Example backup CR

```

apiVersion: velero.io/v1
kind: Backup
metadata:
  name: test-backup
  namespace: openshift-adp
spec:
  includedNamespaces:
    - <application_namespace> ❶
  defaultVolumesToFsBackup: true

```

- ❶ Specify the namespace for the application installed in the cluster.

5. Create a backup by running the following command:

```
$ oc apply -f <backup_file_name> ❶
```

- ❶ Specify the name of the backup CR file.

6. Verify that the backup completed by running the following command:

```
$ oc get backups.velero.io <backup_name> -o yaml ❶
```

- ❶ Specify the name of the backup.

Verification

1. Connect to the Kopia repository by running the following command:

```

$ kopia repository connect s3 \
  --bucket=<bucket_name> \ ❶
  --prefix=velero/kopia/<application_namespace> \ ❷
  --password=static-passw0rd \ ❸
  --access-key="<aws_s3_access_key>" \ ❹
  --secret-access-key="<aws_s3_secret_access_key>" \ ❺

```

- ❶ Specify the AWS S3 bucket name.
- ❷ Specify the namespace for the application.
- ❸ This is the Kopia password to connect to the repository.
- ❹ Specify the AWS S3 access key.
- ❺ Specify the AWS S3 storage provider secret access key.



NOTE

If you are using a storage provider other than AWS S3, you will need to add **--endpoint**, the bucket endpoint URL parameter, to the command.

2. Verify that Kopia uses the environment variables that are configured in the DPA for the backup by running the following command:

```
$ kopia repository status
```

Example output

```
Config file:      ../../config/kopia/repository.config

Description:      Repository in S3: s3.amazonaws.com <bucket_name>
# ...

Storage type:     s3
Storage capacity: unbounded
Storage config:   {
    "bucket": <bucket_name>,
    "prefix": "velero/kopia/<application_namespace>/",
    "endpoint": "s3.amazonaws.com",
    "accessKeyID": <access_key>,
    "secretAccessKey": "*****",
    "sessionToken": ""
}

Unique ID:        58....aeb0
Hash:             BLAKE3-256
Encryption:       CHACHA20-POLY1305-HMAC-SHA256
Splitter:         DYNAMIC-8M-RABINKARP
Format version:   3
# ...
```

5.23.4.3. Benchmarking Kopia hashing, encryption, and splitter algorithms

You can run Kopia commands to benchmark the hashing, encryption, and splitter algorithms. Based on the benchmarking results, you can select the most suitable algorithm for your workload. In this procedure, you run the Kopia benchmarking commands from a pod on the cluster. The benchmarking results can vary depending on CPU speed, available RAM, disk speed, current I/O load, and so on.

Prerequisites

- You have installed the OADP Operator.
- You have an application with persistent volumes running in a separate namespace.
- You have run a backup of the application with Container Storage Interface (CSI) snapshots.



NOTE

The configuration of the Kopia algorithms for splitting, hashing, and encryption in the Data Protection Application (DPA) apply only during the initial Kopia repository creation, and cannot be changed later.

To use different Kopia algorithms, ensure that the object storage does not contain any previous Kopia repositories of backups. Configure a new object storage in the Backup Storage Location (BSL) or specify a unique prefix for the object storage in the BSL configuration.

Procedure

1. Configure the **must-gather** pod as shown in the following example. Make sure you are using the **oadp-mustgather** image for OADP version 1.3 and later.

Example pod configuration

```
apiVersion: v1
kind: Pod
metadata:
  name: oadp-mustgather-pod
  labels:
    purpose: user-interaction
spec:
  containers:
  - name: oadp-mustgather-container
    image: registry.redhat.io/oadp/oadp-mustgather-rhel9:v1.3
    command: ["sleep"]
    args: ["infinity"]
```



NOTE

The Kopia client is available in the **oadp-mustgather** image.

2. Create the pod by running the following command:

```
$ oc apply -f <pod_config_file_name> ❶
```

- ❶ Specify the name of the YAML file for the pod configuration.

3. Verify that the Security Context Constraints (SCC) on the pod is **anyuid**, so that Kopia can connect to the repository.

```
$ oc describe pod/oadp-mustgather-pod | grep scc
```

Example output

```
openshift.io/scc: anyuid
```

4. Connect to the pod via SSH by running the following command:

```
$ oc -n openshift-adp rsh pod/oadp-mustgather-pod
```

5. Connect to the Kopia repository by running the following command:

```
sh-5.1# kopia repository connect s3 \
  --bucket=<bucket_name> \ 1
  --prefix=velero/kopia/<application_namespace> \ 2
  --password=static-passw0rd \ 3
  --access-key="<access_key>" \ 4
  --secret-access-key="<secret_access_key>" \ 5
  --endpoint=<bucket_endpoint> \ 6
```

- 1 Specify the object storage provider bucket name.
- 2 Specify the namespace for the application.
- 3 This is the Kopia password to connect to the repository.
- 4 Specify the object storage provider access key.
- 5 Specify the object storage provider secret access key.
- 6 Specify the bucket endpoint. You do not need to specify the bucket endpoint, if you are using AWS S3 as the storage provider.



NOTE

This is an example command. The command can vary based on the object storage provider.

6. To benchmark the hashing algorithm, run the following command:

```
sh-5.1# kopia benchmark hashing
```

Example output

```
Benchmarking hash 'BLAKE2B-256' (100 x 1048576 bytes, parallelism 1)
Benchmarking hash 'BLAKE2B-256-128' (100 x 1048576 bytes, parallelism 1)
Benchmarking hash 'BLAKE2S-128' (100 x 1048576 bytes, parallelism 1)
Benchmarking hash 'BLAKE2S-256' (100 x 1048576 bytes, parallelism 1)
Benchmarking hash 'BLAKE3-256' (100 x 1048576 bytes, parallelism 1)
Benchmarking hash 'BLAKE3-256-128' (100 x 1048576 bytes, parallelism 1)
Benchmarking hash 'HMAC-SHA224' (100 x 1048576 bytes, parallelism 1)
Benchmarking hash 'HMAC-SHA256' (100 x 1048576 bytes, parallelism 1)
Benchmarking hash 'HMAC-SHA256-128' (100 x 1048576 bytes, parallelism 1)
Benchmarking hash 'HMAC-SHA3-224' (100 x 1048576 bytes, parallelism 1)
Benchmarking hash 'HMAC-SHA3-256' (100 x 1048576 bytes, parallelism 1)
  Hash                Throughput
-----
```

```
0. BLAKE3-256          15.3 GB / second
1. BLAKE3-256-128      15.2 GB / second
2. HMAC-SHA256-128     6.4 GB / second
```

```

3. HMAC-SHA256      6.4 GB / second
4. HMAC-SHA224      6.4 GB / second
5. BLAKE2B-256-128  4.2 GB / second
6. BLAKE2B-256      4.1 GB / second
7. BLAKE2S-256      2.9 GB / second
8. BLAKE2S-128      2.9 GB / second
9. HMAC-SHA3-224    1.6 GB / second
10. HMAC-SHA3-256   1.5 GB / second

```

Fastest option for this machine is: --block-hash=BLAKE3-256

7. To benchmark the encryption algorithm, run the following command:

```
sh-5.1# kopia benchmark encryption
```

Example output

```

Benchmarking encryption 'AES256-GCM-HMAC-SHA256'... (1000 x 1048576 bytes,
parallelism 1)
Benchmarking encryption 'CHACHA20-POLY1305-HMAC-SHA256'... (1000 x 1048576
bytes, parallelism 1)
  Encryption              Throughput
-----
0. AES256-GCM-HMAC-SHA256    2.2 GB / second
1. CHACHA20-POLY1305-HMAC-SHA256 1.8 GB / second
-----
Fastest option for this machine is: --encryption=AES256-GCM-HMAC-SHA256

```

8. To benchmark the splitter algorithm, run the following command:

```
sh-5.1# kopia benchmark splitter
```

Example output

```

splitting 16 blocks of 32MiB each, parallelism 1
DYNAMIC              747.6 MB/s count:107 min:9467 10th:2277562 25th:2971794
50th:4747177 75th:7603998 90th:8388608 max:8388608
DYNAMIC-128K-BUZHASH  718.5 MB/s count:3183 min:3076 10th:80896 25th:104312
50th:157621 75th:249115 90th:262144 max:262144
DYNAMIC-128K-RABINKARP 164.4 MB/s count:3160 min:9667 10th:80098 25th:106626
50th:162269 75th:250655 90th:262144 max:262144
# ...
FIXED-512K           102.9 TB/s count:1024 min:524288 10th:524288 25th:524288
50th:524288 75th:524288 90th:524288 max:524288
FIXED-8M             566.3 TB/s count:64 min:8388608 10th:8388608 25th:8388608
50th:8388608 75th:8388608 90th:8388608 max:8388608
-----
0. FIXED-8M           566.3 TB/s count:64 min:8388608 10th:8388608 25th:8388608
50th:8388608 75th:8388608 90th:8388608 max:8388608
1. FIXED-4M           425.8 TB/s count:128 min:4194304 10th:4194304 25th:4194304
50th:4194304 75th:4194304 90th:4194304 max:4194304
# ...
22. DYNAMIC-128K-RABINKARP 164.4 MB/s count:3160 min:9667 10th:80098
25th:106626 50th:162269 75th:250655 90th:262144 max:262144

```

5.24. APIS USED WITH OADP

You can use the following APIs with OADP:

Velero API

Velero API documentation is maintained by Velero and is not maintained by Red Hat. For more information, see [API types](#) (Velero documentation).

OADP API

The following are the OADP APIs:

- **DataProtectionApplicationSpec**
- **BackupLocation**
- **SnapshotLocation**
- **ApplicationConfig**
- **VeleroConfig**
- **CustomPlugin**
- **ResticConfig**
- **PodConfig**
- **Features**
- **DataMover**

For more information, see in [OADP Operator](#)(Go documentation).

5.24.1. DataProtectionApplicationSpec type

The following are **DataProtectionApplicationSpec** OADP APIs:

Table 5.11. DataProtectionApplicationSpec

Property	Type	Description
backupLocations	[] BackupLocation	Defines the list of configurations to use for BackupStorageLocations .
snapshotLocations	[] SnapshotLocation	Defines the list of configurations to use for VolumeSnapshotLocations .

Property	Type	Description
unsupportedOverrides	map [UnsupportedImageKey] string	Can be used to override the deployed dependent images for development. Options are veleroImageFqin , awsPluginImageFqin , hypershiftPluginImageFqin , openshiftPluginImageFqin , azurePluginImageFqin , gcpPluginImageFqin , csiPluginImageFqin , dataMoverImageFqin , resticRestoreImageFqin , kubevirtPluginImageFqin , and operator-type .
podAnnotations	map [string] string	Used to add annotations to pods deployed by Operators.
podDnsPolicy	DNSPolicy	Defines the configuration of the DNS of a pod.
podDnsConfig	PodDNSConfig	Defines the DNS parameters of a pod in addition to those generated from DNSPolicy .
backupImages	* bool	Used to specify whether or not you want to deploy a registry for enabling backup and restore of images.
configuration	* ApplicationConfig	Used to define the data protection application's server configuration.
features	* Features	Defines the configuration for the DPA to enable the Technology Preview features.

Additional resources

- [Complete schema definitions for the OADP API](#)

5.24.2. BackupLocation type

The following are **BackupLocation** OADP APIs:

Table 5.12. BackupLocation

Property	Type	Description
velero	* velero.BackupStorageLocationSpec	Location to store volume snapshots, as described in Backup Storage Location .
bucket	* CloudStorageLocation	Automates creation of a bucket at some cloud storage providers for use as a backup storage location.



IMPORTANT

The **bucket** parameter 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](#).

Additional resources

- [Complete schema definitions for the type **BackupLocation**](#)

5.24.3. SnapshotLocation type

The following are **SnapshotLocation** OADP APIs:

Table 5.13. SnapshotLocation

Property	Type	Description
velero	* VolumeSnapshotLocationSpec	Location to store volume snapshots, as described in Volume Snapshot Location .

Additional resources

- [Complete schema definitions for the type **SnapshotLocation**](#)

5.24.4. ApplicationConfig type

The following are **ApplicationConfig** OADP APIs:

Table 5.14. ApplicationConfig

Property	Type	Description

Property	Type	Description
velero	* VeleroConfig	Defines the configuration for the Velero server.
restic	* ResticConfig	Defines the configuration for the Restic server.

Additional resources

- [Complete schema definitions for the type **ApplicationConfig**](#)

5.24.5. VeleroConfig type

The following are **VeleroConfig** OADP APIs:

Table 5.15. VeleroConfig

Property	Type	Description
featureFlags	[] string	Defines the list of features to enable for the Velero instance.
defaultPlugins	[] string	The following types of default Velero plugins can be installed: aws , azure , csi , gcp , kubevirt , and openshift .
customPlugins	[] CustomPlugin	Used for installation of custom Velero plugins.
restoreResourcesVersionPriority	string	Represents a config map that is created if defined for use in conjunction with the EnableAPIGroupVersions feature flag. Defining this field automatically adds EnableAPIGroupVersions to the Velero server feature flag.
noDefaultBackupLocation	bool	To install Velero without a default backup storage location, you must set the noDefaultBackupLocation flag in order to confirm installation.
podConfig	* PodConfig	Defines the configuration of the Velero pod.

Property	Type	Description
logLevel	string	Velero server's log level (use debug for the most granular logging, leave unset for Velero default). Valid options are trace , debug , info , warning , error , fatal , and panic .

Additional resources

- [Complete schema definitions for the type **VeleroConfig**](#)

5.24.6. CustomPlugin type

The following are **CustomPlugin** OADP APIs:

Table 5.16. CustomPlugin

Property	Type	Description
name	string	Name of custom plugin.
image	string	Image of custom plugin.

Additional resources

- [Complete schema definitions for the type **CustomPlugin**](#)

5.24.7. ResticConfig type

The following are **ResticConfig** OADP APIs:

Table 5.17. ResticConfig

Property	Type	Description
enable	*bool	If set to true , enables backup and restore using Restic. If set to false , snapshots are needed.
supplementalGroups	[]int64	Defines the Linux groups to be applied to the Restic pod.

Property	Type	Description
timeout	string	A user-supplied duration string that defines the Restic timeout. Default value is 1hr (1 hour). A duration string is a possibly signed sequence of decimal numbers, each with optional fraction and a unit suffix, such as 300ms , -1.5h , or 2h45m . Valid time units are ns , us (or µs), ms , s , m , and h .
podConfig	* PodConfig	Defines the configuration of the Restic pod.

Additional resources

- [Complete schema definitions for the type **ResticConfig**](#)

5.24.8. PodConfig type

The following are **PodConfig** OADP APIs:

Table 5.18. PodConfig

Property	Type	Description
nodeSelector	map [string] string	Defines the nodeSelector to be supplied to a Velero podSpec or a Restic podSpec .
tolerations	[] Toleration	Defines the list of tolerations to be applied to a Velero deployment or a Restic daemonset .
resourceAllocations	ResourceRequirements	Set specific resource limits and requests for a Velero pod or a Restic pod as described in the Setting Velero CPU and memory resource allocations section.
labels	map [string] string	Labels to add to pods.

Additional resources

- [OADP plugins](#)
- [Complete schema definitions for the type **PodConfig**](#)(Go documentation)

5.24.9. Features type

The following are **Features** OADP APIs:

Table 5.19. Features

Property	Type	Description
dataMover	* DataMover	Defines the configuration of the Data Mover.

Additional resources

- [Complete schema definitions for the type **Features**](#)

5.24.10. DataMover type

The following are **DataMover** OADP APIs:

Table 5.20. DataMover

Property	Type	Description
enable	bool	If set to true , deploys the volume snapshot mover controller and a modified CSI Data Mover plugin. If set to false , these are not deployed.
credentialName	string	User-supplied Restic Secret name for Data Mover.
timeout	string	A user-supplied duration string for VolumeSnapshotBackup and VolumeSnapshotRestore to complete. Default is 10m (10 minutes). A duration string is a possibly signed sequence of decimal numbers, each with optional fraction and a unit suffix, such as 300ms , -1.5h , or 2h45m . Valid time units are ns , us (or µs), ms , s , m , and h .

5.25. ADVANCED OADP FEATURES AND FUNCTIONALITIES

This document provides information about advanced features and functionalities of OpenShift API for Data Protection (OADP).

5.25.1. Working with different Kubernetes API versions on the same cluster

5.25.1.1. Listing the Kubernetes API group versions on a cluster

A source cluster might offer multiple versions of an API, where one of these versions is the preferred API version. For example, a source cluster with an API named **Example** might be available in the **example.com/v1** and **example.com/v1beta2** API groups.

If you use Velero to back up and restore such a source cluster, Velero backs up only the version of that resource that uses the preferred version of its Kubernetes API.

To return to the above example, if **example.com/v1** is the preferred API, then Velero only backs up the version of a resource that uses **example.com/v1**. Moreover, the target cluster needs to have **example.com/v1** registered in its set of available API resources in order for Velero to restore the resource on the target cluster.

Therefore, you need to generate a list of the Kubernetes API group versions on your target cluster to be sure the preferred API version is registered in its set of available API resources.

Procedure

- Enter the following command:

```
$ oc api-resources
```

5.25.1.2. About Enable API Group Versions

By default, Velero only backs up resources that use the preferred version of the Kubernetes API. However, Velero also includes a feature, [Enable API Group Versions](#), that overcomes this limitation. When enabled on the source cluster, this feature causes Velero to back up *all* Kubernetes API group versions that are supported on the cluster, not only the preferred one. After the versions are stored in the backup .tar file, they are available to be restored on the destination cluster.

For example, a source cluster with an API named **Example** might be available in the **example.com/v1** and **example.com/v1beta2** API groups, with **example.com/v1** being the preferred API.

Without the Enable API Group Versions feature enabled, Velero backs up only the preferred API group version for **Example**, which is **example.com/v1**. With the feature enabled, Velero also backs up **example.com/v1beta2**.

When the Enable API Group Versions feature is enabled on the destination cluster, Velero selects the version to restore on the basis of the order of priority of API group versions.



NOTE

Enable API Group Versions is still in beta.

Velero uses the following algorithm to assign priorities to API versions, with **1** as the top priority:

1. Preferred version of the *destination* cluster
2. Preferred version of the *source_* cluster
3. Common non-preferred supported version with the highest Kubernetes version priority

Additional resources

- [Enable API Group Versions Feature](#)

5.25.1.3. Using Enable API Group Versions

You can use Velero's Enable API Group Versions feature to back up *all* Kubernetes API group versions that are supported on a cluster, not only the preferred one.



NOTE

Enable API Group Versions is still in beta.

Procedure

- Configure the **EnableAPIGroupVersions** feature flag:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  configuration:
    velero:
      featureFlags:
        - EnableAPIGroupVersions
```

Additional resources

- [Enable API Group Versions Feature](#)

5.25.2. Backing up data from one cluster and restoring it to another cluster

5.25.2.1. About backing up data from one cluster and restoring it on another cluster

OpenShift API for Data Protection (OADP) is designed to back up and restore application data in the same OpenShift Container Platform cluster. Migration Toolkit for Containers (MTC) is designed to migrate containers, including application data, from one OpenShift Container Platform cluster to another cluster.

You can use OADP to back up application data from one OpenShift Container Platform cluster and restore it on another cluster. However, doing so is more complicated than using MTC or using OADP to back up and restore on the same cluster.

To successfully use OADP to back up data from one cluster and restore it to another cluster, you must take into account the following factors, in addition to the prerequisites and procedures that apply to using OADP to back up and restore data on the same cluster:

- Operators
- Use of Velero
- UID and GID ranges

5.25.2.1.1. Operators

You must exclude Operators from the backup of an application for backup and restore to succeed.

5.25.2.1.2. Use of Velero

Velero, which OADP is built upon, does not natively support migrating persistent volume snapshots across cloud providers. To migrate volume snapshot data between cloud platforms, you must *either* enable the Velero Restic file system backup option, which backs up volume contents at the file system level, *or* use the OADP Data Mover for CSI snapshots.



NOTE

In OADP 1.1 and earlier, the Velero Restic file system backup option is called **restic**. In OADP 1.2 and later, the Velero Restic file system backup option is called **file-system-backup**.

- You must also use Velero's [File System Backup](#) to migrate data between AWS regions or between Microsoft Azure regions.
- Velero does not support restoring data to a cluster with an *earlier* Kubernetes version than the source cluster.
- It is theoretically possible to migrate workloads to a destination with a *later* Kubernetes version than the source, but you must consider the compatibility of API groups between clusters for each custom resource. If a Kubernetes version upgrade breaks the compatibility of core or native API groups, you must first update the impacted custom resources.

5.25.2.2. About determining which pod volumes to back up

Before you start a backup operation by using File System Backup (FSB), you must specify which pods contain a volume that you want to back up. Velero refers to this process as "discovering" the appropriate pod volumes.

Velero supports two approaches for determining pod volumes. Use the opt-in or the opt-out approach to allow Velero to decide between an FSB, a volume snapshot, or a Data Mover backup.

- **Opt-in approach:** With the opt-in approach, volumes are backed up using snapshot or Data Mover by default. FSB is used on specific volumes that are opted-in by annotations.
- **Opt-out approach:** With the opt-out approach, volumes are backed up using FSB by default. Snapshots or Data Mover is used on specific volumes that are opted-out by annotations.

5.25.2.2.1. Limitations

- FSB does not support backing up and restoring **hostpath** volumes. However, FSB does support backing up and restoring local volumes.
- Velero uses a static, common encryption key for all backup repositories it creates. **This static key means that anyone who can access your backup storage can also decrypt your backup data.** It is essential that you limit access to backup storage.
- For PVCs, every incremental backup chain is maintained across pod reschedules. For pod volumes that are *not* PVCs, such as **emptyDir** volumes, if a pod is deleted or recreated, for example, by a **ReplicaSet** or a deployment, the next backup of those volumes will be a full backup and not an incremental backup. It is assumed that the lifecycle of a pod volume is defined by its pod.

- Even though backup data can be kept incrementally, backing up large files, such as a database, can take a long time. This is because FSB uses deduplication to find the difference that needs to be backed up.
- FSB reads and writes data from volumes by accessing the file system of the node on which the pod is running. For this reason, FSB can only back up volumes that are mounted from a pod and not directly from a PVC. Some Velero users have overcome this limitation by running a staging pod, such as a BusyBox or Alpine container with an infinite sleep, to mount these PVC and PV pairs before performing a Velero backup..
- FSB expects volumes to be mounted under **<hostPath>/<pod UID>**, with **<hostPath>** being configurable. Some Kubernetes systems, for example, vCluster, do not mount volumes under the **<pod UID>** subdirectory, and VFSB does not work with them as expected.

5.25.2.2.2. Backing up pod volumes by using the opt-in method

You can use the opt-in method to specify which volumes need to be backed up by File System Backup (FSB). You can do this by using the **backup.velero.io/backup-volumes** command.

Procedure

- On each pod that contains one or more volumes that you want to back up, enter the following command:

```
$ oc -n <your_pod_namespace> annotate pod/<your_pod_name> \
  backup.velero.io/backup-volumes=<your_volume_name_1>, \ <your_volume_name_2>>,..., \
  <your_volume_name_n>
```

where:

<your_volume_name_x>

specifies the name of the xth volume in the pod specification.

5.25.2.2.3. Backing up pod volumes by using the opt-out method

When using the opt-out approach, all pod volumes are backed up by using File System Backup (FSB), although there are some exceptions:

- Volumes that mount the default service account token, secrets, and configuration maps.
- **hostPath** volumes

You can use the opt-out method to specify which volumes **not** to back up. You can do this by using the **backup.velero.io/backup-volumes-excludes** command.

Procedure

- On each pod that contains one or more volumes that you do not want to back up, run the following command:

```
$ oc -n <your_pod_namespace> annotate pod/<your_pod_name> \
  backup.velero.io/backup-volumes-excludes=<your_volume_name_1>, \
  <your_volume_name_2>>,...,<your_volume_name_n>
```

where:

`<your_volume_name_x>`

specifies the name of the xth volume in the pod specification.



NOTE

You can enable this behavior for all Velero backups by running the **velero install** command with the **--default-volumes-to-fs-backup** flag.

5.25.2.3. UID and GID ranges

If you back up data from one cluster and restore it to another cluster, problems might occur with UID (User ID) and GID (Group ID) ranges. The following section explains these potential issues and mitigations:

Summary of the issues

The namespace UID and GID ranges might change depending on the destination cluster. OADP does not back up and restore OpenShift UID range metadata. If the backed up application requires a specific UID, ensure the range is available upon restore. For more information about OpenShift's UID and GID ranges, see [A Guide to OpenShift and UIDs](#).

Detailed description of the issues

When you create a namespace in OpenShift Container Platform by using the shell command **oc create namespace**, OpenShift Container Platform assigns the namespace a unique User ID (UID) range from its available pool of UIDs, a Supplemental Group (GID) range, and unique SELinux MCS labels. This information is stored in the **metadata.annotations** field of the cluster. This information is part of the Security Context Constraints (SCC) annotations, which comprise of the following components:

- **openshift.io/sa.scc.mcs**
- **openshift.io/sa.scc.supplemental-groups**
- **openshift.io/sa.scc.uid-range**

When you use OADP to restore the namespace, it automatically uses the information in **metadata.annotations** without resetting it for the destination cluster. As a result, the workload might not have access to the backed up data if any of the following is true:

- There is an existing namespace with other SCC annotations, for example, on another cluster. In this case, OADP uses the existing namespace during the backup instead of the namespace you want to restore.
- A label selector was used during the backup, but the namespace in which the workloads are executed does not have the label. In this case, OADP does not back up the namespace, but creates a new namespace during the restore that does not contain the annotations of the backed up namespace. This results in a new UID range being assigned to the namespace. This can be an issue for customer workloads if OpenShift Container Platform assigns a pod a **securityContext** UID to a pod based on namespace annotations that have changed since the persistent volume data was backed up.
- The UID of the container no longer matches the UID of the file owner.
- An error occurs because OpenShift Container Platform has not changed the UID range of the destination cluster to match the backup cluster data. As a result, the backup cluster has a

different UID than the destination cluster, which means that the application cannot read or write data on the destination cluster.

Mitigations

You can use one or more of the following mitigations to resolve the UID and GID range issues:

- Simple mitigations:
 - If you use a label selector in the **Backup** CR to filter the objects to include in the backup, be sure to add this label selector to the namespace that contains the workspace.
 - Remove any pre-existing version of a namespace on the destination cluster before attempting to restore a namespace with the same name.
- Advanced mitigations:
 - Fix UID ranges after migration by [Resolving overlapping UID ranges in OpenShift namespaces after migration](#). Step 1 is optional.

For an in-depth discussion of UID and GID ranges in OpenShift Container Platform with an emphasis on overcoming issues in backing up data on one cluster and restoring it on another, see [A Guide to OpenShift and UIDs](#).

5.25.2.4. Backing up data from one cluster and restoring it to another cluster

In general, you back up data from one OpenShift Container Platform cluster and restore it on another OpenShift Container Platform cluster in the same way that you back up and restore data to the same cluster. However, there are some additional prerequisites and differences in the procedure when backing up data from one OpenShift Container Platform cluster and restoring it on another.

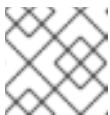
Prerequisites

- All relevant prerequisites for backing up and restoring on your platform (for example, AWS, Microsoft Azure, Google Cloud, and so on), especially the prerequisites for the Data Protection Application (DPA), are described in the relevant sections of this guide.

Procedure

1. Make the following additions to the procedures given for your platform:
 - Ensure that the backup store location (BSL) and volume snapshot location have the same names and paths to restore resources to another cluster.
 - Share the same object storage location credentials across the clusters.
 - For best results, use OADP to create the namespace on the destination cluster.
 - If you use the Velero **file-system-backup** option, enable the **--default-volumes-to-fs-backup** flag for use during backup by running the following command:

```
$ velero backup create <backup_name> --default-volumes-to-fs-backup  
<any_other_options>
```


**NOTE**

In OADP 1.2 and later, the Velero Restic option is called **file-system-backup**.

**IMPORTANT**

Before restoring a CSI back up, edit the **VolumeSnapshotClass** custom resource (CR), and set the **snapshot.storage.kubernetes.io/is-default-class** parameter to false. Otherwise, the restore will partially fail due to the same value in the **VolumeSnapshotClass** in the target cluster for the same drive.

5.25.3. OADP storage class mapping

5.25.3.1. Storage class mapping

Storage class mapping allows you to define rules or policies specifying which storage class should be applied to different types of data. This feature automates the process of determining storage classes based on access frequency, data importance, and cost considerations. It optimizes storage efficiency and cost-effectiveness by ensuring that data is stored in the most suitable storage class for its characteristics and usage patterns.

You can use the **change-storage-class-config** field to change the storage class of your data objects, which lets you optimize costs and performance by moving data between different storage tiers, such as from standard to archival storage, based on your needs and access patterns.

5.25.3.1.1. Storage class mapping with Migration Toolkit for Containers

You can use the Migration Toolkit for Containers (MTC) to migrate containers, including application data, from one OpenShift Container Platform cluster to another cluster and for storage class mapping and conversion. You can convert the storage class of a persistent volume (PV) by migrating it within the same cluster. To do so, you must create and run a migration plan in the MTC web console.

5.25.3.1.2. Mapping storage classes with OADP

You can use OpenShift API for Data Protection (OADP) with the Velero plugin v1.1.0 and later to change the storage class of a persistent volume (PV) during restores, by configuring a storage class mapping in the config map in the Velero namespace.

To deploy ConfigMap with OADP, use the **change-storage-class-config** field. You must change the storage class mapping based on your cloud provider.

Procedure

1. Change the storage class mapping by running the following command:

```
$ cat change-storageclass.yaml
```

2. Create a config map in the Velero namespace as shown in the following example:

Example

```
apiVersion: v1
kind: ConfigMap
metadata:
```

```
name: change-storage-class-config
namespace: openshift-adp
labels:
  velero.io/plugin-config: ""
  velero.io/change-storage-class: RestoreItemAction
data:
  standard-csi: ssd-csi
```

3. Save your storage class mapping preferences by running the following command:

```
$ oc create -f change-storage-class-config
```

5.25.4. Additional resources

- [Working with different Kubernetes API versions on the same cluster](#) .
- [Using Data Mover for CSI snapshots](#) .
- [Backing up applications with File System Backup: Kopia or Restic](#) .
- [Migration converting storage classes](#) .

5.26. OADP TROUBLESHOOTING

5.26.1. Troubleshooting

You can troubleshoot OADP issues by using the following methods:

- Debug Velero custom resources (CRs) by using the [OpenShift CLI tool](#) or the [Velero CLI tool](#). The Velero CLI tool provides more detailed logs and information.
- Debug Velero or Restic pod crashes, which are caused due to a lack of memory or CPU by using [Pods crash or restart due to lack of memory or CPU](#) .
- Debug issues with Velero and admission webhooks by using [Restoring workarounds for Velero backups that use admission webhooks](#).
- Check [OADP installation issues](#), [OADP Operator issues](#), [backup and restore CR issues](#) , and [Restic issues](#).
- Use the available [OADP timeouts](#) to reduce errors, retries, or failures.
- Run the [DataProtectionTest \(DPT\)](#) custom resource to verify your backup storage bucket configuration and check the CSI snapshot readiness for persistent volume claims.
- Collect logs and CR information by using the [must-gather tool](#).
- Monitor and analyze the workload performance with the help of [OADP monitoring](#).

5.26.2. Velero CLI tool

You can obtain the **velero** CLI tool by using the following options:

- Downloading the **velero** CLI tool

- Accessing the **velero** binary in the Velero deployment in the cluster

5.26.2.1. Downloading the Velero CLI tool

You can download and install the Velero CLI tool by following the instructions on the Velero documentation page. The page includes instructions for the following options:

- macOS by using Homebrew
- GitHub
- Windows by using Chocolatey

Prerequisites

- You have access to a Kubernetes cluster, v1.16 or later, with DNS and container networking enabled.
- You have installed **kubectrl** locally.

Procedure

1. Open a browser and navigate to ["Install the CLI" on the Velero website](#).
2. Follow the appropriate procedure for macOS, GitHub, or Windows.
3. Download the Velero version appropriate for your version of OADP and OpenShift Container Platform.

5.26.2.1.1. OADP-Velero-OpenShift Container Platform version relationship

OADP version	Velero version	OpenShift Container Platform version
1.3.0	{velero-1.12}	4.12-4.15
1.3.1	{velero-1.12}	4.12-4.15
1.3.2	{velero-1.12}	4.12-4.15
1.3.3	{velero-1.12}	4.12-4.15
1.3.4	{velero-1.12}	4.12-4.15
1.3.5	{velero-1.12}	4.12-4.15
1.4.0	{velero-1.14}	4.14-4.18
1.4.1	{velero-1.14}	4.14-4.18
1.4.2	{velero-1.14}	4.14-4.18

OADP version	Velero version	OpenShift Container Platform version
1.4.3	{velero-1.14}	4.14-4.18
1.5.0	{velero-1.16}	4.19

5.26.2.2. Accessing the Velero binary in the Velero deployment in the cluster

You can use a shell command to access the Velero binary in the Velero deployment in the cluster.

Prerequisites

- Your **DataProtectionApplication** custom resource has a status of **Reconcile complete**.

Procedure

- Set the needed alias by using the following command:

```
$ alias velero='oc -n openshift-adp exec deployment/velero -c velero -it -- ./velero'
```

5.26.2.3. Debugging Velero resources with the OpenShift CLI tool

You can debug a failed backup or restore by checking Velero custom resources (CRs) and the **Velero** pod log with the OpenShift CLI tool.

Procedure

- Retrieve a summary of warnings and errors associated with a **Backup** or **Restore** CR by using the following **oc describe** command:

```
$ oc describe <velero_cr> <cr_name>
```

- Retrieve the **Velero** pod logs by using the following **oc logs** command:

```
$ oc logs pod/<velero>
```

- Specify the Velero log level in the **DataProtectionApplication** resource as shown in the following example.



NOTE

This option is available starting from OADP 1.0.3.

Example Velero log level file

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
```

```

name: velero-sample
spec:
  configuration:
    velero:
      logLevel: warning

```

The following **logLevel** values are available:

- **trace**
- **debug**
- **info**
- **warning**
- **error**
- **fatal**
- **panic**

Use the **info logLevel** value for most logs.

5.26.2.4. Debugging Velero resources with the Velero CLI tool

You can debug **Backup** and **Restore** custom resources (CRs) and retrieve logs with the Velero CLI tool. The Velero CLI tool provides more detailed information than the OpenShift CLI tool.

Procedure

- Use the **oc exec** command to run a Velero CLI command:

```

$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
  <backup_restore_cr> <command> <cr_name>

```

Example oc exec command

```

$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
  backup describe 0e44ae00-5dc3-11eb-9ca8-df7e5254778b-2d8ql

```

- List all Velero CLI commands by using the following **velero --help** option:

```

$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
  --help

```

- Retrieve the logs of a **Backup** or **Restore** CR by using the following **velero logs** command:

```

$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
  <backup_restore_cr> logs <cr_name>

```

Example velero logs command

```

$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \

```

```
restore logs ccc7c2d0-6017-11eb-afab-85d0007f5a19-x4lbf
```

- Retrieve a summary of warnings and errors associated with a **Backup** or **Restore** CR by using the following **velero describe** command:

```
$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
  <backup_restore_cr> describe <cr_name>
```

Example **velero describe** command

```
$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
  backup describe 0e44ae00-5dc3-11eb-9ca8-df7e5254778b-2d8ql
```

The following types of restore errors and warnings are shown in the output of a **velero describe** request:

- **Velero**: A list of messages related to the operation of Velero itself, for example, messages related to connecting to the cloud, reading a backup file, and so on
- **Cluster**: A list of messages related to backing up or restoring cluster-scoped resources
- **Namespaces**: A list of list of messages related to backing up or restoring resources stored in namespaces
One or more errors in one of these categories results in a **Restore** operation receiving the status of **PartiallyFailed** and not **Completed**. Warnings do not lead to a change in the completion status.

Consider the following points for these restore errors:

- For resource-specific errors, that is, **Cluster** and **Namespaces** errors, the **restore describe --details** output includes a resource list that includes all resources that Velero restored. For any resource that has such an error, check if the resource is actually in the cluster.
- If there are **Velero** errors but no resource-specific errors in the output of a **describe** command, it is possible that the restore completed without any actual problems in restoring workloads. In this case, carefully validate post-restore applications.
For example, if the output contains **PodVolumeRestore** or node agent-related errors, check the status of **PodVolumeRestores** and **DataDownloads**. If none of these are failed or still running, then volume data might have been fully restored.

5.26.3. Pods crash or restart due to lack of memory or CPU

If a Velero or Restic pod crashes due to a lack of memory or CPU, you can set specific resource requests for either of those resources. The values for the resource request fields must follow the same format as Kubernetes resource requirements.

If you do not specify **configuration.velero.podConfig.resourceAllocations** or **configuration.restic.podConfig.resourceAllocations**, see the following default **resources** specification configuration for a Velero or Restic pod:

```
requests:
  cpu: 500m
  memory: 128Mi
```

Additional resources

- [Velero CPU and memory requirements based on collected data](#)

5.26.3.1. Setting resource requests for a Velero pod

You can use the **configuration.velero.podConfig.resourceAllocations** specification field in the **oadp_v1alpha1_dpa.yaml** file to set specific resource requests for a **Velero** pod.

Procedure

- Set the **cpu** and **memory** resource requests in the YAML file:

Example Velero file

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
configuration:
  velero:
    podConfig:
      resourceAllocations: 1
      requests:
        cpu: 200m
        memory: 256Mi
```

- 1 The **resourceAllocations** listed are for average usage.

5.26.3.2. Setting resource requests for a Restic pod

You can use the **configuration.restic.podConfig.resourceAllocations** specification field to set specific resource requests for a **Restic** pod.



NOTE

With OADP 1.5.0, the **configuration.restic.podConfig.resourceAllocations** specification field is removed from Data Protection Application (DPA). Use the **nodeAgent** section with the **uploaderType** field set to **Kopia** instead of **Restic**.

Procedure

- Set the **cpu** and **memory** resource requests in the YAML file:

Example Restic file

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
configuration:
  restic:
    podConfig:
      resourceAllocations: 1
```

```
requests:
  cpu: 1000m
  memory: 16Gi
```

- 1 The **resourceAllocations** listed are for average usage.

5.26.3.3. Setting resource requests for a nodeAgent pod

You can use the **configuration.nodeAgent.podConfig.resourceAllocations** specification field to set specific resource requests for a **nodeAgent** pod.



NOTE

With OADP 1.5.0, the **configuration.restic.podConfig.resourceAllocations** specification field is removed from Data Protection Application (DPA). Use the **nodeAgent** section with the **uploaderType** field set to **Kopia** instead of **Restic**.

Procedure

1. Set the **cpu** and **memory** resource requests in the YAML file:

Example nodeAgent.yaml file

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: ts-dpa
spec:
  backupLocations:
  - velero:
      default: true
      objectStorage:
        bucket: oadp.....njph
        prefix: velero
      credential:
        key: cloud
        name: cloud-credentials-gcp
      provider: gcp
  configuration:
    velero:
      defaultPlugins:
      - gcp
      - openshift
      - csi
  nodeAgent:
    enable: true
    uploaderType: kopia
    podConfig:
      resourceAllocations: 1
      requests:
        cpu: 1000m
        memory: 16Gi 2
```


- 1 The resource allocation examples shown are for average usage.
- 2 You can modify this parameter depending on your infrastructure and usage.

2. Create the DPA CR by running the following command:

```
$ oc create -f nodeAgent.yaml
```

Verification

1. Verify that the **nodeAgent** pods are running by using the following command:

```
$ oc get pods
```

Example output

NAME	READY	STATUS	RESTARTS	AGE
node-agent-hbj9l	1/1	Running	0	97s
node-agent-wmwgz	1/1	Running	0	95s
node-agent-zvc7k	1/1	Running	0	98s
openshift-adp-controller-manager-7f9db86d96-4lhgq	1/1	Running	0	137m
velero-7b6c7fb8d7-ppc8m	1/1	Running	0	4m2s

2. Check the resource requests by describing one of the **nodeAgent** pod:

```
$ oc describe pod node-agent-hbj9l | grep -C 5 Requests
```

Example output

```
--log-format=text
State:      Running
  Started:   Mon, 09 Jun 2025 16:22:15 +0530
Ready:      True
Restart Count: 0
Requests:
  cpu:      1
  memory:   1Gi
Environment:
  NODE_NAME:      (v1:spec.nodeName)
  VELERO_NAMESPACE: openshift-adp (v1:metadata.namespace)
```

5.26.4. Restoring workarounds for Velero backups that use admission webhooks

Velero has limited abilities to resolve admission webhook issues during a restore. If you have workloads with admission webhooks, you might need to use an additional Velero plugin or make changes to how you restore the workload.

Typically, workloads with admission webhooks require you to create a resource of a specific kind first. This is especially true if your workload has child resources because admission webhooks typically block child resources.

For example, creating or restoring a top-level object such as **service.serving.knative.dev** typically

creates child resources automatically. If you do this first, you will not need to use Velero to create and restore these resources. This avoids the problem of child resources being blocked by an admission webhook that Velero might use.



NOTE

Velero plugins are started as separate processes. After a Velero operation has completed, either successfully or not, it exits. Receiving a **received EOF, stopping recv loop** message in the debug logs indicates that a plugin operation has completed. It does not mean that an error has occurred.

5.26.4.1. Restoring Knative resources

You might encounter problems by using Velero to back up Knative resources that use admission webhooks.

You can avoid such problems by restoring the top level **Service** resource whenever you back up and restore Knative resources that use admission webhooks.

Procedure

- Restore the top level **service.serving.knative.dev Service** resource by using the following command:

```
$ velero restore <restore_name> \
  --from-backup=<backup_name> --include-resources \
  service.serving.knative.dev
```

5.26.4.2. Restoring IBM AppConnect resources

If you experience issues when you use Velero to restore an IBM® AppConnect resource that has an admission webhook, you can run the checks in this procedure.

Procedure

- Check if you have any mutating admission plugins of **kind: MutatingWebhookConfiguration** in the cluster by entering/running the following command:

```
$ oc get mutatingwebhookconfigurations
```

- Examine the YAML file of each **kind: MutatingWebhookConfiguration** to ensure that none of its rules block creation of the objects that are experiencing issues. For more information, see [the official Kubernetes documentation](#).
- Check that any **spec.version** in **type: Configuration.appconnect.ibm.com/v1beta1** used at backup time is supported by the installed Operator.

5.26.4.3. Avoiding the Velero plugin panic error

A missing secret can cause a panic error for the Velero plugin during image stream backups.

When the backup and the Backup Storage Location (BSL) are managed outside the scope of the Data Protection Application (DPA), the OADP controller does not create the relevant **oadp-<bsl_name>-<bsl_provider>-registry-secret** parameter.

During the backup operation, the OpenShift Velero plugin panics on the **imagestream** backup, with the following panic error:

```
024-02-27T10:46:50.028951744Z time="2024-02-27T10:46:50Z" level=error msg="Error backing up item"
backup=openshift-adp/<backup name> error="error executing custom action
(groupResource=imagestreams.image.openshift.io,
namespace=<BSL Name>, name=postgres): rpc error: code = Aborted desc = plugin panicked:
runtime error: index out of range with length 1, stack trace: goroutine 94...
```

Use the following workaround to avoid the Velero plugin panic error.

Procedure

1. Label the custom BSL with the relevant label by using the following command:

```
$ oc label backupstoragelocations.velero.io <bsl_name> app.kubernetes.io/component=bsl
```

2. After the BSL is labeled, wait until the DPA reconciles.



NOTE

You can force the reconciliation by making any minor change to the DPA itself.

Verification

- After the DPA is reconciled, confirm that the parameter has been created and that the correct registry data has been populated into it by entering the following command:

```
$ oc -n openshift-adp get secret/oadp-<bsl_name>-<bsl_provider>-registry-secret -o json | jq
-r '.data'
```

5.26.4.4. Workaround for OpenShift ADP Controller segmentation fault

If you configure a Data Protection Application (DPA) with both **cloudstorage** and **restic** enabled, the **openshift-adp-controller-manager** pod crashes and restarts indefinitely until the pod fails with a crash loop segmentation fault.

Define either **velero** or **cloudstorage** when you configure a DPA. Otherwise, the **openshift-adp-controller-manager** pod fails with a crash loop segmentation fault due to the following settings:

- If you define both **velero** and **cloudstorage**, the **openshift-adp-controller-manager** fails.
- If you do not define both **velero** and **cloudstorage**, the **openshift-adp-controller-manager** fails.

For more information about this issue, see [OADP-1054](#).

Additional resources

- [Admission plugins](#)
- [Webhook admission plugins](#)
- [Types of webhook admission plugins](#)

5.26.5. OADP installation issues

You might encounter issues caused by using invalid directories or incorrect credentials when you install the Data Protection Application (DPA).

5.26.5.1. Resolving invalid directories in backup storage

The object storage contains top-level directories that are not Velero directories. The **Velero** pod log displays the following error message:

```
Backup storage contains invalid top-level directories.
```

Procedure

- If the object storage is not dedicated to Velero, you must specify a prefix for the bucket by setting the **spec.backupLocations.velero.objectStorage.prefix** parameter in the **DataProtectionApplication** manifest.

5.26.5.2. Resolving incorrect AWS credentials

If the **credentials-velero** file that is used to create the **Secret** object is incorrectly formatted, multiple errors might occur, including the following examples:

- The **oadp-aws-registry** pod log displays the following error message:

```
`InvalidAccessKeyId: The AWS Access Key Id you provided does not exist in our records.`
```

- The **Velero** pod log displays the following error message:

```
NoCredentialProviders: no valid providers in chain.
```

Procedure

- Ensure that the **credentials-velero** file is correctly formatted, as shown in the following example:

Example credentials-velero file

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

- ❶ AWS default profile.
- ❷ Do not enclose the values with quotation marks (" , ").

5.26.6. OADP Operator issues

The OpenShift API for Data Protection (OADP) Operator might encounter issues caused by problems it is not able to resolve.

5.26.6.1. Resolving silent failure of the OADP Operator

The S3 buckets of an OADP Operator might be empty, but when you run the command **oc get po -n <oadp_operator_namespace>**, you see that the Operator has a status of **Running**.

In such a case, the Operator is said to have *failed silently* because it incorrectly reports that it is running. The problem is caused when cloud credentials provide insufficient permissions.

To fix this issue, retrieve a list of backup storage locations (BSLs) and check the manifest of each BSL for credential issues.

Procedure

1. Retrieve a list of BSLs by using either the OpenShift or Velero command-line interface (CLI):

- a. Retrieve a list of BSLs by using the OpenShift CLI (**oc**):

```
$ oc get backupstoragelocations.velero.io -A
```

- b. Retrieve a list of BSLs by using the **velero** CLI:

```
$ velero backup-location get -n <oadp_operator_namespace>
```

2. Use the list of BSLs from the previous step and run the following command to examine the manifest of each BSL for an error:

```
$ oc get backupstoragelocations.velero.io -n <namespace> -o yaml
```

Example result

```
apiVersion: v1
items:
- apiVersion: velero.io/v1
  kind: BackupStorageLocation
  metadata:
    creationTimestamp: "2023-11-03T19:49:04Z"
    generation: 9703
    name: example-dpa-1
    namespace: openshift-adp-operator
    ownerReferences:
    - apiVersion: oadp.openshift.io/v1alpha1
      blockOwnerDeletion: true
      controller: true
      kind: DataProtectionApplication
      name: example-dpa
      uid: 0beeeaff-0287-4f32-bcb1-2e3c921b6e82
    resourceVersion: "24273698"
    uid: ba37cd15-cf17-4f7d-bf03-8af8655cea83
  spec:
```

```

config:
  enableSharedConfig: "true"
  region: us-west-2
credential:
  key: credentials
  name: cloud-credentials
default: true
objectStorage:
  bucket: example-oadp-operator
  prefix: example
  provider: aws
status:
  lastValidationTime: "2023-11-10T22:06:46Z"
  message: "BackupStorageLocation \"example-dpa-1\" is unavailable: rpc
    error: code = Unknown desc = WebIdentityErr: failed to retrieve credentials\ncaused
    by: AccessDenied: Not authorized to perform sts:AssumeRoleWithWebIdentity\n\tstatus
    code: 403, request id: d3f2e099-70a0-467b-997e-ff62345e3b54"
  phase: Unavailable
kind: List
metadata:
  resourceVersion: ""

```

5.26.7. OADP timeouts

Extending a timeout allows complex or resource-intensive processes to complete successfully without premature termination. This configuration can reduce errors, retries, or failures.

Ensure that you balance timeout extensions in a logical manner so that you do not configure excessively long timeouts that might hide underlying issues in the process. Consider and monitor an appropriate timeout value that meets the needs of the process and the overall system performance.

The following OADP timeouts show instructions of how and when to implement these parameters:

- [Restic timeout](#)
- [Velero resource timeout](#)
- [Data Mover timeout](#)
- [CSI snapshot timeout](#)
- [Item operation timeout - backup](#)
- [Item operation timeout - restore](#)

5.26.7.1. Implementing restic timeout

The **`spec.configuration.nodeAgent.timeout`** parameter defines the Restic timeout. The default value is **1h**.

Use the Restic **`timeout`** parameter in the **`nodeAgent`** section for the following scenarios:

- For Restic backups with total PV data usage that is greater than 500GB.
- If backups are timing out with the following error:

```
level=error msg="Error backing up item" backup=velero/monitoring error="timed out waiting
for all PodVolumeBackups to complete"
```

Procedure

- Edit the values in the **spec.configuration.nodeAgent.timeout** block of the **DataProtectionApplication** custom resource (CR) manifest, as shown in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_name>
spec:
  configuration:
    nodeAgent:
      enable: true
      uploaderType: restic
      timeout: 1h
# ...
```

5.26.7.2. Implementing velero resource timeout

resourceTimeout defines how long to wait for several Velero resources before timeout occurs, such as Velero custom resource definition (CRD) availability, **volumeSnapshot** deletion, and repository availability. The default is **10m**.

Use the **resourceTimeout** for the following scenarios:

- For backups with total PV data usage that is greater than 1 TB. This parameter is used as a timeout value when Velero tries to clean up or delete the Container Storage Interface (CSI) snapshots, before marking the backup as complete.
 - A sub-task of this cleanup tries to patch VSC, and this timeout can be used for that task.
- To create or ensure a backup repository is ready for filesystem based backups for Restic or Kopia.
- To check if the Velero CRD is available in the cluster before restoring the custom resource (CR) or resource from the backup.

Procedure

- Edit the values in the **spec.configuration.velero.resourceTimeout** block of the **DataProtectionApplication** CR manifest, as shown in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_name>
spec:
  configuration:
    velero:
      resourceTimeout: 10m
# ...
```

5.26.7.2.1. Implementing velero default item operation timeout

The **defaultItemOperationTimeout** setting defines how long to wait on asynchronous **BackupItemActions** and **RestoreItemActions** to complete before timing out. The default value is **1h**.

Use the **defaultItemOperationTimeout** for the following scenarios:

- Only with Data Mover 1.2.x.
- To specify the amount of time a particular backup or restore should wait for the Asynchronous actions to complete. In the context of OADP features, this value is used for the Asynchronous actions involved in the Container Storage Interface (CSI) Data Mover feature.
- When **defaultItemOperationTimeout** is defined in the Data Protection Application (DPA) using the **defaultItemOperationTimeout**, it applies to both backup and restore operations. You can use **itemOperationTimeout** to define only the backup or only the restore of those CRs, as described in the following "Item operation timeout - restore", and "Item operation timeout - backup" sections.

Procedure

- Edit the values in the **spec.configuration.velero.defaultItemOperationTimeout** block of the **DataProtectionApplication** CR manifest, as shown in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_name>
spec:
  configuration:
    velero:
      defaultItemOperationTimeout: 1h
# ...
```

5.26.7.3. Implementing Data Mover timeout

timeout is a user-supplied timeout to complete **VolumeSnapshotBackup** and **VolumeSnapshotRestore**. The default value is **10m**.

Use the Data Mover **timeout** for the following scenarios:

- If creation of **VolumeSnapshotBackups** (VSBs) and **VolumeSnapshotRestores** (VSRs), times out after 10 minutes.
- For large scale environments with total PV data usage that is greater than 500GB. Set the timeout for **1h**.
- With the **VolumeSnapshotMover** (VSM) plugin.
- Only with OADP 1.1.x.

Procedure

- Edit the values in the **spec.features.dataMover.timeout** block of the **DataProtectionApplication** CR manifest, as shown in the following example:


```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_name>
spec:
  features:
    dataMover:
      timeout: 10m
# ...

```

5.26.7.4. Implementing CSI snapshot timeout

CSISnapshotTimeout specifies the time during creation to wait until the **CSI VolumeSnapshot** status becomes **ReadyToUse**, before returning error as timeout. The default value is **10m**.

Use the **CSISnapshotTimeout** for the following scenarios:

- With the CSI plugin.
- For very large storage volumes that may take longer than 10 minutes to snapshot. Adjust this timeout if timeouts are found in the logs.



NOTE

Typically, the default value for **CSISnapshotTimeout** does not require adjustment, because the default setting can accommodate large storage volumes.

Procedure

- Edit the values in the **spec.csiSnapshotTimeout** block of the **Backup** CR manifest, as shown in the following example:

```

apiVersion: velero.io/v1
kind: Backup
metadata:
  name: <backup_name>
spec:
  csiSnapshotTimeout: 10m
# ...

```

5.26.7.5. Implementing item operation timeout - restore

The **ItemOperationTimeout** setting specifies the time that is used to wait for **RestoreItemAction** operations. The default value is **1h**.

Use the restore **ItemOperationTimeout** for the following scenarios:

- Only with Data Mover 1.2.x.
- For Data Mover uploads and downloads to or from the **BackupStorageLocation**. If the restore action is not completed when the timeout is reached, it will be marked as failed. If Data Mover operations are failing due to timeout issues, because of large storage volume sizes, then this timeout setting may need to be increased.

Procedure

- Edit the values in the **Restore.spec.itemOperationTimeout** block of the **Restore** CR manifest, as shown in the following example:

```
apiVersion: velero.io/v1
kind: Restore
metadata:
  name: <restore_name>
spec:
  itemOperationTimeout: 1h
# ...
```

5.26.7.6. Implementing item operation timeout - backup

The **ItemOperationTimeout** setting specifies the time used to wait for asynchronous **BackupItemAction** operations. The default value is **1h**.

Use the backup **ItemOperationTimeout** for the following scenarios:

- Only with Data Mover 1.2.x.
- For Data Mover uploads and downloads to or from the **BackupStorageLocation**. If the backup action is not completed when the timeout is reached, it will be marked as failed. If Data Mover operations are failing due to timeout issues, because of large storage volume sizes, then this timeout setting may need to be increased.

Procedure

- Edit the values in the **Backup.spec.itemOperationTimeout** block of the **Backup** CR manifest, as shown in the following example:

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: <backup_name>
spec:
  itemOperationTimeout: 1h
# ...
```

5.26.8. Backup and Restore CR issues

You might encounter the following common issues with **Backup** and **Restore** custom resources (CRs):

- Backup CR cannot retrieve volume
- Backup CR status remains in progress
- Backup CR status remains in the **PartiallyFailed** phase/state/etc

5.26.8.1. Troubleshooting issue where backup CR cannot retrieve volume

If the persistent volume (PV) and the snapshot locations are in different regions, the **Backup** custom resource (CR) displays the following error message:

Sample error

InvalidVolume.NotFound: The volume 'vol-xxxx' does not exist.

Procedure

1. Edit the value of the **spec.snapshotLocations.velero.config.region** key in the **DataProtectionApplication** manifest so that the snapshot location is in the same region as the PV.
2. Create a new **Backup** CR.

5.26.8.2. Troubleshooting issue where backup CR status remains in progress

If a backup is interrupted, it cannot be resumed, and the status of a **Backup** customer resource (CR) remains in the **InProgress** phase and does not complete.

Procedure

1. Retrieve the details of the **Backup** CR by running the following command:

```
$ oc -n {namespace} exec deployment/velero -c velero -- ./velero \
  backup describe <backup>
```

2. Delete the **Backup** CR by running the following command:

```
$ oc delete backups.velero.io <backup> -n openshift-adp
```

You do not need to clean up the backup location because an in progress **Backup** CR has not uploaded files to object storage.

3. Create a new **Backup** CR.
4. View the Velero backup details by running the following command:

```
$ velero backup describe <backup_name> --details
```

5.26.8.3. Troubleshooting issue where backup CR status remains partially failed

The status of a **Backup** CR without Restic in use remains in the **PartiallyFailed** phase and is not completed. A snapshot of the affiliated PVC is not created.

If the backup created based on the CSI snapshot class is missing a label, the CSI snapshot plugin fails to create a snapshot. As a result, the **Velero** pod logs an error similar to the following message:

```
time="2023-02-17T16:33:13Z" level=error msg="Error backing up item" backup=openshift-adp/user1-
backup-check5 error="error executing custom action (groupResource=persistentvolumeclaims,
namespace=busy1, name=pvc1-user1): rpc error: code = Unknown desc = failed to get
volumesnapshotclass for storageclass ocs-storagecluster-ceph-rbd: failed to get
volumesnapshotclass for provisioner openshift-storage.rbd.csi.ceph.com, ensure that the desired
volumesnapshot class has the velero.io/csi-volumesnapshot-class label" logSource="/remote-
source/velero/app/pkg/backup/backup.go:417" name=busybox-79799557b5-vprq
```

Procedure

1. Delete the **Backup** CR by running the following command::

```
$ oc delete backups.velero.io <backup> -n openshift-adp
```

2. If required, clean up the stored data on the **BackupStorageLocation** resource to free up space.
3. Apply the **velero.io/csi-volumesnapshot-class=true** label to the **VolumeSnapshotClass** object by running the following command:

```
$ oc label volumesnapshotclass/<snapclass_name> velero.io/csi-volumesnapshot-class=true
```

4. Create a new **Backup** CR.

5.26.9. Restic issues

You might encounter the following issues when you back up applications with Restic:

- Restic permission error for NFS data volumes with the **root_squash** resource/parameter enabled
- Restic **Backup** CR cannot be recreated after bucket is emptied
- Restic restore partially failing on OpenShift Container Platform 4.14 due to changed pod security admission (PSA) policy

5.26.9.1. Troubleshooting Restic permission errors for NFS data volumes

If your NFS data volumes have the **root_squash** parameter enabled, **Restic** maps set to the **nfsnobody** value, and do not have permission to create backups, the Restic` pod log displays the following error message:

Sample error

```
controller=pod-volume-backup error="fork/exec/usr/bin/restic: permission denied".
```

You can resolve this issue by creating a supplemental group for **Restic** and adding the group ID to the **DataProtectionApplication** manifest.

Procedure

1. Create a supplemental group for **Restic** on the NFS data volume.
2. Set the **setgid** bit on the NFS directories so that group ownership is inherited.
3. Add the **spec.configuration.nodeAgent.supplementalGroups** parameter and the group ID to the **DataProtectionApplication** manifest, as shown in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
# ...
spec:
  configuration:
```

```
nodeAgent:
  enable: true
  uploaderType: restic
  supplementalGroups:
    - <group_id> 1
# ...
```

- 1** Specify the supplemental group ID.

4. Wait for the **Restic** pods to restart so that the changes are applied.

5.26.9.2. Troubleshooting Restic Backup CR issue that cannot be re-created after bucket is emptied

Velero does not re-create or update the Restic repository from the **ResticRepository** manifest if the Restic directories are deleted from object storage. For more information, see [Velero issue 4421](#).

If you create a Restic **Backup** CR for a namespace, empty the object storage bucket, and then re-create the **Backup** CR for the same namespace, the re-created **Backup** CR fails. In this case, the **velero** pod log displays the following error message:

+ .Sample error

```
stderr=Fatal: unable to open config file: Stat: The specified key does not exist.\nls there a repository
at the following location?
```

Procedure

- Remove the related Restic repository from the namespace by running the following command:

```
$ oc delete resticrepository openshift-adp <name_of_the_reestic_repository>
```

In the following error log, **mysql-persistent** is the problematic Restic repository. The name of the repository appears in italics for clarity.

```
time="2021-12-29T18:29:14Z" level=info msg="1 errors
encountered backup up item" backup=velero/backup65
logSource="pkg/backup/backup.go:431" name=mysql-7d99fc949-qbkds
time="2021-12-29T18:29:14Z" level=error msg="Error backing up item"
backup=velero/backup65 error="pod volume backup failed: error running
restic backup, stderr=Fatal: unable to open config file: Stat: The
specified key does not exist.\nls there a repository at the following
location?\n3:http://minio-minio.apps.mayap-oadp-
veleo-1234.qe.devcluster.openshift.com/mayapvelerooadp2/velero1/
restic/mysql-persistent\n: exit status 1" error.file="/remote-source/
src/github.com/vmware-tanzu/velero/pkg/restic/backupper.go:184"
error.function="github.com/vmware-tanzu/velero/
pkg/restic.(*backupper).BackupPodVolumes"
logSource="pkg/backup/backup.go:435" name=mysql-7d99fc949-qbkds
```

5.26.9.3. Troubleshooting restic restore partially failed issue on OpenShift Container Platform 4.14 due to changed PSA policy

OpenShift Container Platform 4.14 enforces a Pod Security Admission (PSA) policy that can hinder the readiness of pods during a Restic restore process.

If a **SecurityContextConstraints** (SCC) resource is not found when a pod is created, and the PSA policy on the pod is not set up to meet the required standards, pod admission is denied.

This issue arises due to the resource restore order of Velero.

Sample error

```
"level=error" in line#2273: time="2023-06-12T06:50:04Z"
level=error msg="error restoring mysql-869f9f44f6-tp5lv: pods\\
"mysql-869f9f44f6-tp5lv\\" is forbidden: violates PodSecurity\\
"restricted:v1.24\\": privil eged (container \\\"mysql\\
" must not set securityContext.privileged=true),
allowPrivilegeEscalation != false (containers \\
"restic-wait\\", \\\"mysql\\\" must set securityContext.allowPrivilegeEscalation=false), unrestricted
capabilities (containers \\
"restic-wait\\", \\\"mysql\\\" must set securityContext.capabilities.drop=[\\\"ALL\\\"]), seccompProfile
(pod or containers \\
"restic-wait\\", \\\"mysql\\\" must set securityContext.seccompProfile.type to \\
"RuntimeDefault\\\" or \\\"Localhost\\\")" logSource="/remote-
source/velero/app/pkg/restore/restore.go:1388" restore=openshift-adp/todolist-backup-0780518c-
08ed-11ee-805c-0a580a80e92c\n
velero container contains "level=error" in line#2447: time="2023-06-12T06:50:05Z"
level=error msg="Namespace todolist-mariadb,
resource restore error: error restoring pods/todolist-mariadb/mysql-869f9f44f6-tp5lv: pods \\
"mysql-869f9f44f6-tp5lv\\" is forbidden: violates PodSecurity \\\"restricted:v1.24\\\": privileged
(container \\
"mysql\\\" must not set securityContext.privileged=true),
allowPrivilegeEscalation != false (containers \\
"restic-wait\\",\\\"mysql\\\" must set securityContext.allowPrivilegeEscalation=false), unrestricted
capabilities (containers \\
"restic-wait\\", \\\"mysql\\\" must set securityContext.capabilities.drop=[\\\"ALL\\\"]), seccompProfile
(pod or containers \\
"restic-wait\\", \\\"mysql\\\" must set securityContext.seccompProfile.type to \\
"RuntimeDefault\\\" or \\\"Localhost\\\")"
logSource="/remote-source/velero/app/pkg/controller/restore_controller.go:510"
restore=openshift-adp/todolist-backup-0780518c-08ed-11ee-805c-0a580a80e92c\n"]
```

Procedure

1. In your DPA custom resource (CR), check or set the **restore-resource-priorities** field on the Velero server to ensure that **securitycontextconstraints** is listed in order before **pods** in the list of resources:

```
$ oc get dpa -o yaml
```

Example DPA CR

```
# ...
configuration:
  restic:
    enable: true
  velero:
```

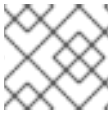
```

args:
  restore-resource-priorities:
'securitycontextconstraints,customresourcedefinitions,namespaces,storageclasses,volumesnap:
hotclass.snapshot.storage.k8s.io,volumesnapshotcontents.snapshot.storage.k8s.io,volumesnap
hots.snapshot.storage.k8s.io,datauploads.velero.io,persistentvolumes,persistentvolumeclaims,s
erviceaccounts,secrets,configmaps,limitranges,pods,replicasets.apps,clusterclasses.cluster.x-
k8s.io,endpoints,services,-,clusterbootstraps.run.tanzu.vmware.com,clusters.cluster.x-
k8s.io,clusterresourcesets.addons.cluster.x-k8s.io' 1
  defaultPlugins:
    - gcp
    - openshift

```

1 If you have an existing restore resource priority list, ensure you combine that existing list with the complete list.

2. Ensure that the security standards for the application pods are aligned, as provided in [Fixing PodSecurity Admission warnings for deployments](#), to prevent deployment warnings. If the application is not aligned with security standards, an error can occur regardless of the SCC.



NOTE

This solution is temporary, and ongoing discussions are in progress to address it.

Additional resources

- [Fixing PodSecurity Admission warnings for deployments](#)

5.26.10. OADP Data protection test

The **DataProtectionTest** (DPT) is a custom resource (CR) that provides a framework to validate your OADP configuration. The DPT CR checks and reports information for the following parameters:

- The upload performance of the backups to the object storage.
- The CSI snapshot readiness for persistent volume claims.
- The storage bucket configuration, such as encryption and versioning.

Using this information in the DPT CR, you can ensure that your data protection environment is properly configured and performing according to the set configuration.

5.26.10.1. OADP DataProtectionTest CR specification fields

You can configure the following specification fields in the **DataProtectionTest** (DPT) custom resource (CR).

Table 5.21. DPT CR spec fields

Field	Type	Description

Field	Type	Description
backupLocationName	string	Name of the BackupStorageLocation CR configured in the DataProtectionApplication (DPA) CR.
backupLocationSpec	object	Inline specification of the BackupStorageLocation CR.
uploadSpeedTestConfig	object	Configuration to run an upload speed test to the object storage.
csiVolumeSnapshotTestConfigs	list	List of persistent volume claims to take a snapshot of and to verify the snapshot readiness.
forceRun	boolean	Re-run the DPT CR even if status is Complete or Failed .
skipTLSVerify	boolean	Bypasses the TLS certificate validation if set to true .

5.26.10.2. OADP DataProtectionTest CR status fields

You can review the status of the **DataProtectionTest** (DPT) custom resource (CR) by using the following status fields:

Table 5.22. DPT CR status fields

Field	Type	Description
phase	string	Current phase of the DPT CR. Values are InProgress , Complete , or Failed .
lastTested	timestamp	The timestamp when the DPT CR was last run.
uploadTest	object	Results of the upload speed test.
bucketMetadata	object	Information about the storage bucket encryption and versioning.
snapshotTests	list	Snapshot test results for each persistent volume claim.

Field	Type	Description
snapshotSummary	string	Aggregated pass/fail summary for snapshots. For example, 2/2 passed .
s3Vendor	string	AWS S3-compatible storage bucket vendors. For example, AWS, MinIO, Ceph.
errorMessage	string	Error message if the DPT CR fails.

5.26.10.3. Using the DataProtectionTest custom resource

You can configure the **DataProtectionTest** (DPT) custom resource (CR) and then run the DPT CR to verify the Container Storage Initiative (CSI) snapshot readiness and the data upload performance to the storage bucket.

Prerequisites

- You have logged in to the OpenShift Container Platform cluster as a user with the **cluster-admin** role.
- You have installed the OpenShift CLI (**oc**).
- You have installed the OADP Operator.
- You have created the **DataProtectionApplication** (DPA) CR.
- You have configured a backup storage location (BSL) to store the backups.
- You have an application with persistent volume claims (PVCs) running in a separate namespace.

Procedure

1. Create a manifest file for the DPT CR as shown in the example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionTest
metadata:
  name: dpt-sample
  namespace: openshift-adp
spec:
  backupLocationName: <bsl_name> ❶
  csiVolumeSnapshotTestConfigs: ❷
  - snapshotClassName: csi-gce-pd-vsc
    timeout: 90s
    volumeSnapshotSource:
      persistentVolumeClaimName: <pvc1_name> ❸
      persistentVolumeClaimNamespace: <pvc_namespace> ❹
  - snapshotClassName: csi-gce-pd-vsc
    timeout: 120s
```

```

volumeSnapshotSource:
  persistentVolumeClaimName: <pvc2_name> 5
  persistentVolumeClaimNamespace: <pvc_namespace>
forceRun: false 6
uploadSpeedTestConfig: 7
  fileSize: 200MB
  timeout: 120s

```

- 1** Specify the name of the BSL.
- 2** Specify a list for **csiVolumeSnapshotTestConfigs**. In this example, two PVCs are being tested.
- 3** Specify the name of the first PVC.
- 4** Specify the namespace of the PVC.
- 5** Specify the name of the second PVC.
- 6** Set the **forceRun** flag to **false** if you want to make the OADP controller skip re-running tests.
- 7** Configure the **uploadSpeedTestConfig** object by setting the **fileSize** and **timeout** fields.

2. Create the DPT CR by running the following command:

```
$ oc create -f <dpt_file_name> 1
```

- 1** Specify the file name of the DPT manifest.

Verification

1. Verify that the phase of the DPT CR is **Complete** by running the following command:

```
$ oc get dpt dpt-sample
```

The example output is as following:

```

NAME      PHASE    LASTTESTED  UPLOADSPEED(MBPS)  ENCRYPTION
VERSIONING SNAPSHOT  AGE
dpt-sample Complete  17m        546             AES256    Enabled    2/2 passed  17m

```

2. Verify that the CSI snapshots are ready and the data upload tests are successful by running the following command:

```
$ oc get dpt dpt-sample -o yaml
```

The example output is as following:

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionTest
....

```

```

status:
  bucketMetadata: ❶
    encryptionAlgorithm: AES256
    versioningStatus: Enabled
    lastTested: "202...:47:51Z"
    phase: Complete
    s3Vendor: AWS ❷
    snapshotSummary: 2/2 passed ❸
    snapshotTests:
      - persistentVolumeClaimName: mysql-data
        persistentVolumeClaimNamespace: ocp-mysql
        readyDuration: 24s
        status: Ready
      - persistentVolumeClaimName: mysql-data1
        persistentVolumeClaimNamespace: ocp-mysql
        readyDuration: 40s
        status: Ready
    uploadTest: ❹
      duration: 3.071s
      speedMbps: 546
      success: true

```

- ❶ The bucket metadata information.
- ❷ The S3 bucket vendor.
- ❸ Summary of the CSI snapshot tests.
- ❹ The upload test details.

5.26.10.4. Running a data protection test by configuring a backup storage location specification

You can configure the **DataProtectionTest** (DPT) custom resource (CR) by specifying the backup storage location (BSL) specification instead of a BSL name. You then run the DPT CR to verify the Container Storage Initiative (CSI) snapshot readiness and the data upload performance to the storage bucket.

Prerequisites

- You have logged in to the OpenShift Container Platform cluster as a user with the **cluster-admin** role.
- You have installed the OpenShift CLI (**oc**).
- You have installed the OADP Operator.
- You have created the **DataProtectionApplication** (DPA) CR.
- You have configured a bucket to store the backups.
- You have created the **Secret** object to access the bucket storage.
- You have an application with persistent volume claims (PVCs) running in a separate namespace.

Procedure

1. Create a manifest file for the DPT CR as shown in the example:

```

apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionTest
metadata:
  name: dpt-sample
  namespace: openshift-adp
spec:
  backupLocationSpec: ❶
    provider: aws
    default: true
    objectStorage:
      bucket: sample-bucket ❷
      prefix: velero
    config:
      region: us-east-1 ❸
      profile: "default"
      insecureSkipTLSVerify: "true"
      s3Url: "https://s3.amazonaws.com/sample-bucket"
    credential: ❹
      name: cloud-credentials
      key: cloud
  uploadSpeedTestConfig: ❺
    fileSize: 50MB
    timeout: 120s
  csiVolumeSnapshotTestConfigs: ❻
    - volumeSnapshotSource:
        persistentVolumeClaimName: mongo
        persistentVolumeClaimNamespace: mongo-persistent
        snapshotClassName: csi-snapclass
        timeout: 2m
  forceRun: true
  skipTLSVerify: true ❼

```

- ❶ Configure the BSL spec by specifying details such as the cloud provider.
- ❷ Specify the bucket name. In this example, the bucket name is **sample-bucket**.
- ❸ Specify the cloud provider region.
- ❹ Specify the cloud credentials for the storage bucket.
- ❺ (Optional) Configure the **uploadSpeedTestConfig** object by setting the **fileSize** and **timeout** fields.
- ❻ Configure the **csiVolumeSnapshotTestConfigs** object.
- ❼ Set to **true** to skip the TLS certificate validation during the DPT CR run.

2. Create the DPT CR by running the following command:

```
$ oc create -f <dpt_file_name> ❶
```

- 1 Specify the file name of the DPT manifest.

Verification

1. Verify that the phase of the DPT CR is **Complete** by running the following command:

```
$ oc get dpt dpt-sample
```

The example output is as following:

```
NAME      PHASE    LASTTESTED  UPLOADSPEED(MBPS)  ENCRYPTION
VERSIONING SNAPSHOTS  AGE
dpt-sample Complete  17m         546                AES256    Enabled    2/2 passed  17m
```

5.26.10.5. Running a data protection test on an Azure object storage

If you are using OADP on an Azure object storage, you need to specify the Azure **STORAGE_ACCOUNT_ID** as part of the secret object. Use the following procedure to run a **DataProtectionTest** (DPT) custom resource (CR) on an Azure cluster.

Prerequisites

- You have logged in to the Azure cluster as a user with the **cluster-admin** role.
- You have installed the OpenShift CLI (**oc**).
- You have installed the OADP Operator.
- You have configured a bucket to store the backups.
- You have an application with persistent volume claims (PVCs) running in a separate namespace.

Procedure

1. Add the **Storage Blob Data Contributor** role to Azure **storageAccount** object to avoid DPT run failure. Run the following command:

```
$ az role assignment create \
--assignee "$AZURE_CLIENT_ID" \
--role "Storage Blob Data Contributor" \
--scope
"/subscriptions/$AZURE_SUBSCRIPTION_ID/resourceGroups/$AZURE_RESOURCE_GRO
UP/providers/Microsoft.Storage/storageAccounts/$AZURE_STORAGE_ACCOUNT_ID"
```

2. In your terminal, export the Azure parameters and create a secret credentials file with the parameters as shown in the following example.
To run the DPT CR on Azure, you need to specify the **STORAGE_ACCOUNT_ID** parameter in the secret credentials file.

```
AZURE_SUBSCRIPTION_ID=<subscription-id>
AZURE_TENANT_ID=<tenant-id>
AZURE_CLIENT_ID=<client-id>
```

```
AZURE_CLIENT_SECRET=<client-secret>
AZURE_RESOURCE_GROUP=<resource-group>
AZURE_STORAGE_ACCOUNT_ID=<storage-account>
```

3. Create the **Secret** CR as shown in the following example:

```
$ oc create secret generic cloud-credentials-azure -n openshift-adp --from-file cloud=
<credentials_file_path>
```

4. Create the **DataProtectionApplication** (DPA) CR by using the configuration shown in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: ts-dpa
  namespace: openshift-adp
spec:
  configuration:
    velero:
      defaultPlugins:
        - azure
        - openshift
  backupLocations:
    - velero:
        config:
          resourceGroup: oadp-....-b7q4-rg
          storageAccount: oadp...kb7q4
          subscriptionId: 53b8f5...fd54c8a
        credential:
          key: cloud
          name: cloud-credentials-azure ❶
        provider: azure
        default: true
        objectStorage:
          bucket: <bucket_name>
          prefix: velero
```

- ❶ Specify the name of the **Secret** object. In this example, the name is **cloud-credentials-azure**.

5. Create the DPT CR by specifying the name of backup storage location (BSL), **VolumeSnapshotClass** object, and the persistent volume claim details as shown in the following example:

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionTest
metadata:
  name: dpt-sample
  namespace: openshift-adp
spec:
  backupLocationName: <bsl_name> ❶
  uploadSpeedTestConfig:
    fileSize: 40MB
```

```

timeout: 120s
csiVolumeSnapshotTestConfigs:
- snapshotClassName: csi-azuredisk-vsc ❷
  timeout: 90s
  volumeSnapshotSource:
    persistentVolumeClaimName: mysql-data ❸
    persistentVolumeClaimNamespace: ocp-mysql ❹
- snapshotClassName: csi-azuredisk-vsc
  timeout: 120s
  volumeSnapshotSource:
    persistentVolumeClaimName: mysql-data1
    persistentVolumeClaimNamespace: ocp-mysql

```

- ❶ Specify the name of the BSL.
- ❷ The Azure snapshot class name.
- ❸ The name of the persistent volume claim.
- ❹ The name of the persistent volume claim namespace.

6. Run the DPT CR to verify the snapshot readiness.

5.26.10.6. Troubleshooting the DataProtectionTest custom resource

Use the following table to troubleshoot common issues when running the **DataProtectionTest** (DPT) custom resource (CR).

Table 5.23. DPT CR troubleshooting

Error	Reason	Solution
DPT stuck in InProgress state	Bucket credentials or bucket access failure	Check Secret object, bucket permissions, and logs.
Upload test failed	Incorrect Secret object or S3 endpoint	Check the BackupStorageLocation object config and the access keys.
Snapshot tests fail	Incorrect configuration of CSI snapshot controller	Check the VolumeSnapshotClass object availability and the CSI driver logs.
Bucket encryption or versioning not populated	Cloud provider limitations	Not all object storage providers expose these fields consistently.

5.26.11. Using the must-gather tool

You can collect logs and information about OADP custom resources by using the **must-gather** tool. The **must-gather** data must be attached to all customer cases.

The **must-gather** tool is a container and does not run all the time. The tool runs for a few minutes only after you invoke the tool by running the **must-gather** command.

5.26.11.1. Using the must-gather tool

You can run the **must-gather** tool with the following options. To use an option, add a flag corresponding to that option in the **must-gather** command.

Default configuration

This configuration collects pod logs, OADP, and **Velero** custom resource (CR) information for all namespaces where the OADP Operator is installed.

Timeout

Data collection can take a long time if there are many failed **Backup** CRs. You can improve performance by setting a timeout value.

Insecure TLS connections

If a custom CA certificate is used, use the **must-gather** tool with insecure TLS connections.

The **must-gather** tool generates a Markdown output file with the collected information. The Markdown file is located in a cluster directory.

For more information about the supported flags, use the help flag with the **must-gather** tool as shown in the following example:

```
$ oc adm must-gather --image=registry.redhat.io/oadp/oadp-mustgather-rhel9:v1.5 -- /usr/bin/gather -h
```

Prerequisites

- You have logged in to the OpenShift Container Platform cluster as a user with the **cluster-admin** role.
- You have installed the OpenShift CLI (**oc**).

Procedure

1. Navigate to the directory where you want to store the **must-gather** data.
2. Run the **oc adm must-gather** command for one of the following data collection options:
 - To use the default configuration of the **must-gather** tool, run the following command:

```
$ oc adm must-gather --image=registry.redhat.io/oadp/oadp-mustgather-rhel9:v1.5
```

- To use the timeout flag with the **must-gather** tool, run the following command:

```
$ oc adm must-gather --image=registry.redhat.io/oadp/oadp-mustgather-rhel9:v1.5 -- /usr/bin/gather --request-timeout 1m 1
```

1 In this example, the timeout is 1 minute.

- To use the insecure TLS connection flag with the **must-gather** tool, run the following command:

```
$ oc adm must-gather --image=registry.redhat.io/oadp/oadp-mustgather-rhel9:v1.5 --
/usr/bin/gather --skip-tls
```

- To use a combination of the insecure TLS connection and the timeout flags with the **must-gather** tool, run the following command:

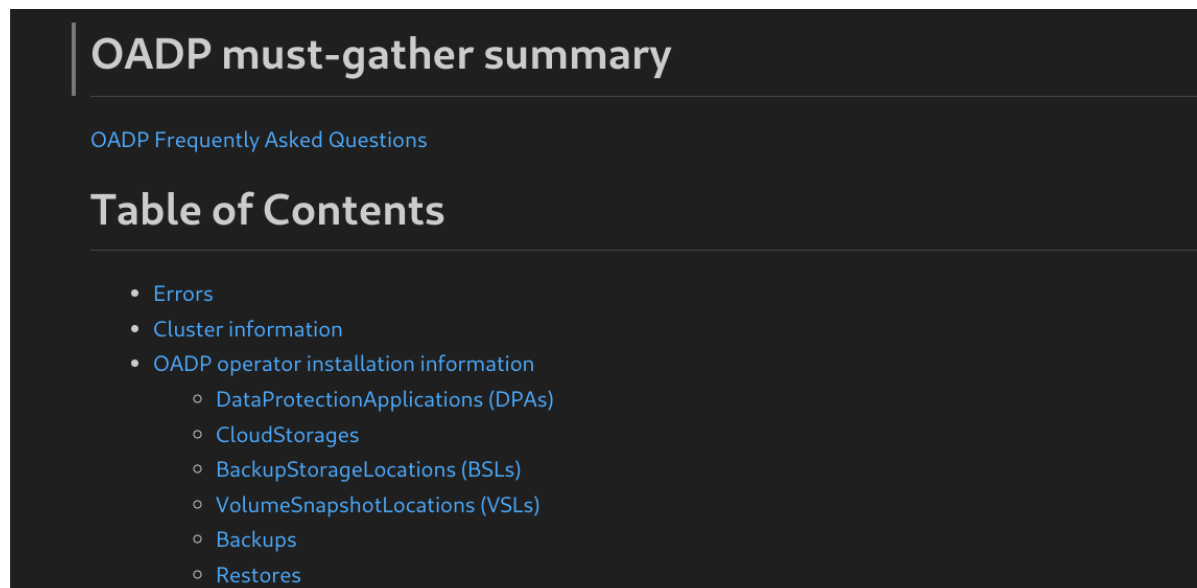
```
$ oc adm must-gather --image=registry.redhat.io/oadp/oadp-mustgather-rhel9:v1.5 --
/usr/bin/gather --request-timeout 15s --skip-tls ❶
```

- ❶ In this example, the timeout is 15 seconds. By default, the **--skip-tls** flag value is **false**. Set the value to **true** to allow insecure TLS connections.

Verification

1. Verify that the Markdown output file is generated at the following location: **must-gather.local.89...054550/registry.redhat.io/oadp/oadp-mustgather-rhel9:v1.5-sha256-0...84/clusters/a4...86/oadp-must-gather-summary.md**
2. Review the **must-gather** data in the Markdown file by opening the file in a Markdown previewer. For an example output, refer to the following image. You can upload this output file to a support case on the [Red Hat Customer Portal](#).

Figure 5.2. Example markdown output of must-gather tool



Additional resources

- [Gathering cluster data](#)

5.26.12. OADP monitoring

By using the OpenShift Container Platform monitoring stack, users and administrators can effectively perform the following tasks:

- Monitor and manage clusters
- Analyze the workload performance of user applications
- Monitor services running on the clusters
- Receive alerts if an event occurs

Additional resources

- [About OpenShift Container Platform monitoring](#)

5.26.12.1. OADP monitoring setup

The OADP Operator leverages an OpenShift User Workload Monitoring provided by the OpenShift Monitoring Stack for retrieving metrics from the Velero service endpoint. The monitoring stack allows creating user-defined Alerting Rules or querying metrics by using the OpenShift Metrics query front end.

With enabled User Workload Monitoring, it is possible to configure and use any Prometheus-compatible third-party UI, such as Grafana, to visualize Velero metrics.

Monitoring metrics requires enabling monitoring for the user-defined projects and creating a **ServiceMonitor** resource to scrape those metrics from the already enabled OADP service endpoint that resides in the **openshift-adp** namespace.



NOTE

The OADP support for Prometheus metrics is offered on a best-effort basis and is not fully supported.

For more information about setting up the monitoring stack, see [Configuring user workload monitoring](#).

Prerequisites

- You have access to an OpenShift Container Platform cluster using an account with **cluster-admin** permissions.
- You have created a cluster monitoring config map.

Procedure

1. Edit the **cluster-monitoring-config ConfigMap** object in the **openshift-monitoring** namespace by using the following command:

```
$ oc edit configmap cluster-monitoring-config -n openshift-monitoring
```

2. Add or enable the **enableUserWorkload** option in the **data** section's **config.yaml** field by using the following command:

```
apiVersion: v1
kind: ConfigMap
data:
  config.yaml: |
```

```
enableUserWorkload: true 1
metadata:
# ...
```

- 1** Add this option or set to **true**

3. Wait a short period to verify the User Workload Monitoring Setup by checking that the following components are up and running in the **openshift-user-workload-monitoring** namespace:

```
$ oc get pods -n openshift-user-workload-monitoring
```

Example output

NAME	READY	STATUS	RESTARTS	AGE
prometheus-operator-6844b4b99c-b57j9	2/2	Running	0	43s
prometheus-user-workload-0	5/5	Running	0	32s
prometheus-user-workload-1	5/5	Running	0	32s
thanos-ruler-user-workload-0	3/3	Running	0	32s
thanos-ruler-user-workload-1	3/3	Running	0	32s

4. Verify the existence of the **user-workload-monitoring-config** ConfigMap in the **openshift-user-workload-monitoring**. If it exists, skip the remaining steps in this procedure.

```
$ oc get configmap user-workload-monitoring-config -n openshift-user-workload-monitoring
```

Example output

```
Error from server (NotFound): configmaps "user-workload-monitoring-config" not found
```

5. Create a **user-workload-monitoring-config ConfigMap** object for the User Workload Monitoring, and save it under the **2_configure_user_workload_monitoring.yaml** file name:

Example output

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
```

6. Apply the **2_configure_user_workload_monitoring.yaml** file by using the following command:

```
$ oc apply -f 2_configure_user_workload_monitoring.yaml
configmap/user-workload-monitoring-config created
```

5.26.12.2. Creating OADP service monitor

OADP provides an **openshift-adp-velero-metrics-svc** service, which is created when the Data Protection Application (DPA) is configured. The user workload monitoring service monitor must point to the defined service. To get details about the service, complete the following steps.

Procedure

1. Ensure that the **openshift-adp-velero-metrics-svc** service exists. It should contain **app.kubernetes.io/name=velero** label, which is used as selector for the **ServiceMonitor** object.

```
$ oc get svc -n openshift-adp -l app.kubernetes.io/name=velero
```

Example output

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
openshift-adp-velero-metrics-svc	ClusterIP	172.30.38.244	<none>	8085/TCP	1h

2. Create a **ServiceMonitor** YAML file that matches the existing service label, and save the file as **3_create_oadp_service_monitor.yaml**. The service monitor is created in the **openshift-adp** namespace where the **openshift-adp-velero-metrics-svc** service resides.

Example ServiceMonitor object

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  labels:
    app: oadp-service-monitor
    name: oadp-service-monitor
    namespace: openshift-adp
spec:
  endpoints:
    - interval: 30s
      path: /metrics
      targetPort: 8085
      scheme: http
  selector:
    matchLabels:
      app.kubernetes.io/name: "velero"
```

3. Apply the **3_create_oadp_service_monitor.yaml** file:

```
$ oc apply -f 3_create_oadp_service_monitor.yaml
```

Example output

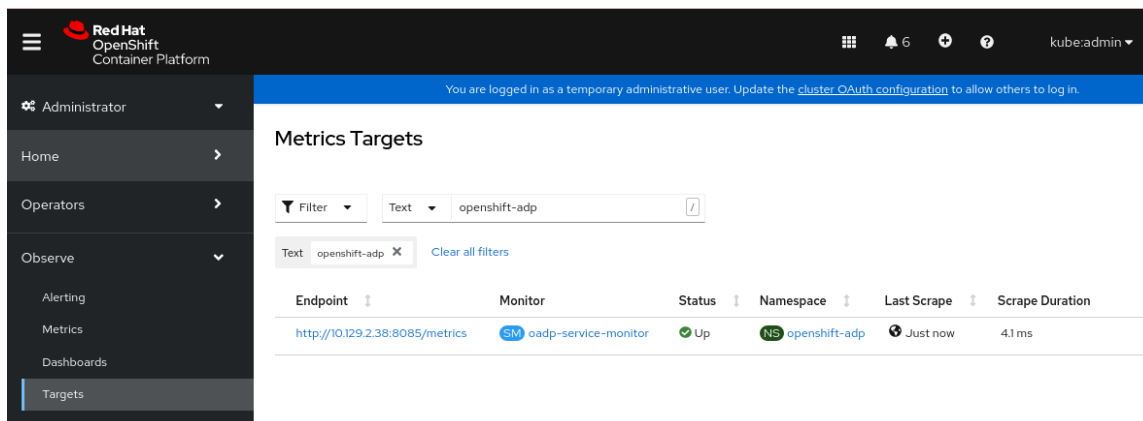
```
servicemonitor.monitoring.coreos.com/oadp-service-monitor created
```

Verification

- Confirm that the new service monitor is in an **Up** state by using the **Administrator** perspective of the OpenShift Container Platform web console. Wait a few minutes for the service monitor to reach the **Up** state.

- Navigate to the **Observe → Targets** page.
- Ensure the **Filter** is unselected or that the **User** source is selected and type **openshift-adp** in the **Text** search field.
- Verify that the status for the **Status** for the service monitor is **Up**.

Figure 5.3. OADP metrics targets



5.26.12.3. Creating an alerting rule

The OpenShift Container Platform monitoring stack receives Alerts configured by using Alerting Rules. To create an Alerting rule for the OADP project, use one of the Metrics scraped with the user workload monitoring.

Procedure

- Create a **PrometheusRule** YAML file with the sample **OADPBackupFailing** alert and save it as **4_create_oadp_alert_rule.yaml**:

Sample OADPBackupFailing alert

```
apiVersion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
  name: sample-oadp-alert
  namespace: openshift-adp
spec:
  groups:
  - name: sample-oadp-backup-alert
    rules:
    - alert: OADPBackupFailing
      annotations:
        description: 'OADP had {{$value | humanize}} backup failures over the last 2 hours.'
        summary: OADP has issues creating backups
      expr: |
        increase(velero_backup_failure_total{job="openshift-adp-velero-metrics-svc"}[2h]) > 0
      for: 5m
      labels:
        severity: warning
```

In this sample, the Alert displays under the following conditions:

- During the last 2 hours, the number of new failing backups was greater than 0 and the state persisted for at least 5 minutes.
 - If the time of the first increase is less than 5 minutes, the Alert is in a **Pending** state, after which it turns into a **Firing** state.
2. Apply the **4_create_oadp_alert_rule.yaml** file, which creates the **PrometheusRule** object in the **openshift-adp** namespace:

```
$ oc apply -f 4_create_oadp_alert_rule.yaml
```

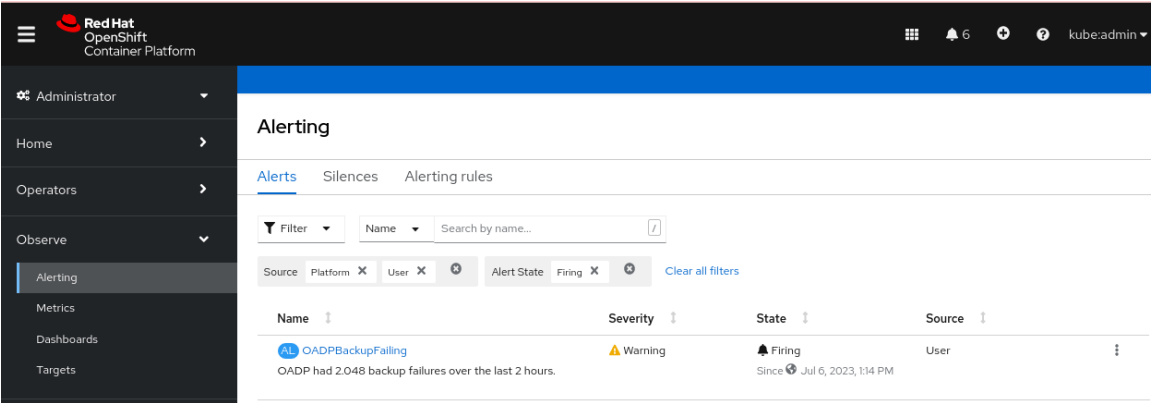
Example output

```
prometheusrule.monitoring.coreos.com/sample-oadp-alert created
```

Verification

- After the Alert is triggered, you can view it in the following ways:
 - In the **Developer** perspective, select the **Observe** menu.
 - In the **Administrator** perspective under the **Observe → Alerting** menu, select **User** in the **Filter** box. Otherwise, by default only the **Platform** Alerts are displayed.

Figure 5.4. OADP backup failing alert



Additional resources

- [Managing alerts as an Administrator](#)

5.26.12.4. List of available metrics

Refer to the following table for a list of **Velero** metrics provided by OADP together with their [Types](#):

Table 5.24. Velero metrics

Metric name	Description	Type
velero_backup_tarball_size_bytes	Size, in bytes, of a backup	Gauge

Metric name	Description	Type
velero_backup_total	Current number of existent backups	Gauge
velero_backup_attempt_total	Total number of attempted backups	Counter
velero_backup_success_total	Total number of successful backups	Counter
velero_backup_partial_failure_total	Total number of partially failed backups	Counter
velero_backup_failure_total	Total number of failed backups	Counter
velero_backup_validation_failure_total	Total number of validation failed backups	Counter
velero_backup_duration_seconds	Time taken to complete backup, in seconds	Histogram
velero_backup_duration_seconds_bucket	Total count of observations for a bucket in the histogram for the metric velero_backup_duration_seconds	Counter
velero_backup_duration_seconds_count	Total count of observations for the metric velero_backup_duration_seconds	Counter
velero_backup_duration_seconds_sum	Total sum of observations for the metric velero_backup_duration_seconds	Counter
velero_backup_deletion_attempt_total	Total number of attempted backup deletions	Counter
velero_backup_deletion_success_total	Total number of successful backup deletions	Counter
velero_backup_deletion_failure_total	Total number of failed backup deletions	Counter

Metric name	Description	Type
velero_backup_last_successful_timestamp	Last time a backup ran successfully, Unix timestamp in seconds	Gauge
velero_backup_items_total	Total number of items backed up	Gauge
velero_backup_items_errors	Total number of errors encountered during backup	Gauge
velero_backup_warning_total	Total number of warned backups	Counter
velero_backup_last_status	Last status of the backup. A value of 1 is success, 0 is failure	Gauge
velero_restore_total	Current number of existent restores	Gauge
velero_restore_attempt_total	Total number of attempted restores	Counter
velero_restore_validation_failed_total	Total number of failed restores failing validations	Counter
velero_restore_success_total	Total number of successful restores	Counter
velero_restore_partial_failure_total	Total number of partially failed restores	Counter
velero_restore_failed_total	Total number of failed restores	Counter
velero_volume_snapshot_attempt_total	Total number of attempted volume snapshots	Counter
velero_volume_snapshot_success_total	Total number of successful volume snapshots	Counter
velero_volume_snapshot_failure_total	Total number of failed volume snapshots	Counter
velero_csi_snapshot_attempt_total	Total number of CSI attempted volume snapshots	Counter
velero_csi_snapshot_success_total	Total number of CSI successful volume snapshots	Counter

Metric name	Description	Type
velero_csi_snapshot_failure_total	Total number of CSI failed volume snapshots	Counter

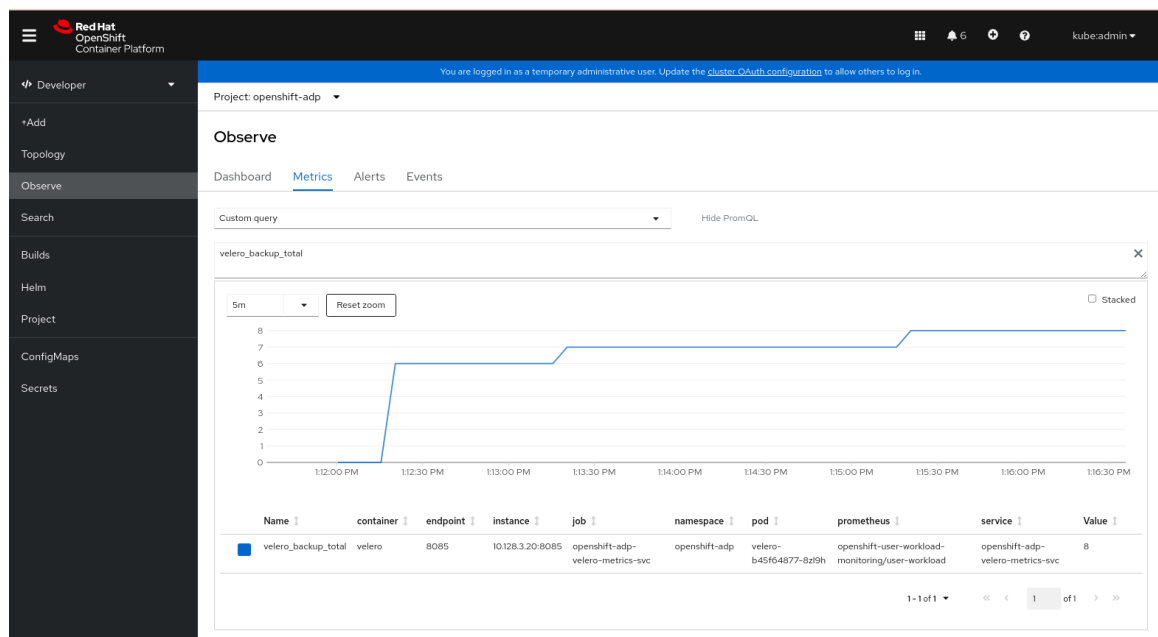
5.26.12.5. Viewing metrics using the Observe UI

You can view metrics in the OpenShift Container Platform web console from the **Administrator** or **Developer** perspective, which must have access to the **openshift-adp** project.

Procedure

- Navigate to the **Observe → Metrics** page:
 - If you are using the **Developer** perspective, follow these steps:
 - a. Select **Custom query**, or click on the **Show PromQL** link.
 - b. Type the query and click **Enter**.
 - If you are using the **Administrator** perspective, type the expression in the text field and select **Run Queries**.

Figure 5.5. OADP metrics query



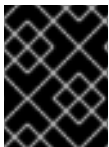
CHAPTER 6. CONTROL PLANE BACKUP AND RESTORE

6.1. BACKING UP ETCD

etcd is the key-value store for OpenShift Container Platform, which persists the state of all resource objects.

Back up your cluster's etcd data regularly and store in a secure location ideally outside the OpenShift Container Platform environment. Do not take an etcd backup before the first certificate rotation completes, which occurs 24 hours after installation, otherwise the backup will contain expired certificates. It is also recommended to take etcd backups during non-peak usage hours because the etcd snapshot has a high I/O cost.

Be sure to take an etcd backup before you update your cluster. Taking a backup before you update is important because when you restore your cluster, you must use an etcd backup that was taken from the same z-stream release. For example, an OpenShift Container Platform 4.17.5 cluster must use an etcd backup that was taken from 4.17.5.



IMPORTANT

Back up your cluster's etcd data by performing a single invocation of the backup script on a control plane host. Do not take a backup for each control plane host.

After you have an etcd backup, you can [restore to a previous cluster state](#).

6.1.1. Backing up etcd data

Follow these steps to back up etcd data by creating an etcd snapshot and backing up the resources for the static pods. This backup can be saved and used at a later time if you need to restore etcd.



IMPORTANT

Only save a backup from a single control plane host. Do not take a backup from each control plane host in the cluster.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have checked whether the cluster-wide proxy is enabled.

TIP

You can check whether the proxy is enabled by reviewing the output of **oc get proxy cluster -o yaml**. The proxy is enabled if the **httpProxy**, **httpsProxy**, and **noProxy** fields have values set.

Procedure

1. Start a debug session as root for a control plane node:

```
$ oc debug --as-root node/<node_name>
```

2. Change your root directory to **/host** in the debug shell:

```
sh-4.4# chroot /host
```

3. If the cluster-wide proxy is enabled, export the **NO_PROXY**, **HTTP_PROXY**, and **HTTPS_PROXY** environment variables by running the following commands:

```
$ export HTTP_PROXY=http://<your_proxy.example.com>:8080
```

```
$ export HTTPS_PROXY=https://<your_proxy.example.com>:8080
```

```
$ export NO_PROXY=<example.com>
```

4. Run the **cluster-backup.sh** script in the debug shell and pass in the location to save the backup to.

TIP

The **cluster-backup.sh** script is maintained as a component of the etcd Cluster Operator and is a wrapper around the **etcdctl snapshot save** command.

```
sh-4.4# /usr/local/bin/cluster-backup.sh /home/core/assets/backup
```

Example script output

```
found latest kube-apiserver: /etc/kubernetes/static-pod-resources/kube-apiserver-pod-6
found latest kube-controller-manager: /etc/kubernetes/static-pod-resources/kube-controller-
manager-pod-7
found latest kube-scheduler: /etc/kubernetes/static-pod-resources/kube-scheduler-pod-6
found latest etcd: /etc/kubernetes/static-pod-resources/etcd-pod-3
ede95fe6b88b87ba86a03c15e669fb4aa5bf0991c180d3c6895ce72eaade54a1
etcdctl version: 3.4.14
API version: 3.4
{"level":"info","ts":1624647639.0188997,"caller":"snapshot/v3_snapshot.go:119","msg":"created
temporary db file","path":"/home/core/assets/backup/snapshot_2021-06-25_190035.db.part"}
{"level":"info","ts":"2021-06-
25T19:00:39.030Z","caller":"clientv3/maintenance.go:200","msg":"opened snapshot stream;
downloading"}
{"level":"info","ts":1624647639.0301006,"caller":"snapshot/v3_snapshot.go:127","msg":"fetching
snapshot","endpoint":"https://10.0.0.5:2379"}
{"level":"info","ts":"2021-06-
25T19:00:40.215Z","caller":"clientv3/maintenance.go:208","msg":"completed snapshot read;
closing"}
{"level":"info","ts":1624647640.6032252,"caller":"snapshot/v3_snapshot.go:142","msg":"fetched
snapshot","endpoint":"https://10.0.0.5:2379","size":"114 MB","took":1.584090459}
{"level":"info","ts":1624647640.6047094,"caller":"snapshot/v3_snapshot.go:152","msg":"saved",
"path":"/home/core/assets/backup/snapshot_2021-06-25_190035.db"}
Snapshot saved at /home/core/assets/backup/snapshot_2021-06-25_190035.db
{"hash":3866667823,"revision":31407,"totalKey":12828,"totalSize":114446336}
snapshot db and kube resources are successfully saved to /home/core/assets/backup
```

In this example, two files are created in the `/home/core/assets/backup/` directory on the control plane host:

- **snapshot_<timestamp>.db**: This file is the etcd snapshot. The **cluster-backup.sh** script confirms its validity.
- **static_kuberresources_<timestamp>.tar.gz**: This file contains the resources for the static pods. If etcd encryption is enabled, it also contains the encryption keys for the etcd snapshot.



NOTE

If etcd encryption is enabled, it is recommended to store this second file separately from the etcd snapshot for security reasons. However, this file is required to restore from the etcd snapshot.

Keep in mind that etcd encryption only encrypts values, not keys. This means that resource types, namespaces, and object names are unencrypted.

6.1.2. Additional resources

- [Recovering an unhealthy etcd cluster](#)

6.1.3. Creating automated etcd backups

The automated backup feature for etcd supports both recurring and single backups. Recurring backups create a cron job that starts a single backup each time the job triggers.



IMPORTANT

Automating etcd backups 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](#).

Follow these steps to enable automated backups for etcd.



WARNING

Enabling the **TechPreviewNoUpgrade** feature set on your cluster prevents minor version updates. The **TechPreviewNoUpgrade** feature set cannot be disabled. Do not enable this feature set on production clusters.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have access to the OpenShift CLI (**oc**).

Procedure

1. Create a **FeatureGate** custom resource (CR) file named **enable-tech-preview-no-upgrade.yaml** with the following contents:

```
apiVersion: config.openshift.io/v1
kind: FeatureGate
metadata:
  name: cluster
spec:
  featureSet: TechPreviewNoUpgrade
```

2. Apply the CR and enable automated backups:

```
$ oc apply -f enable-tech-preview-no-upgrade.yaml
```

3. It takes time to enable the related APIs. Verify the creation of the custom resource definition (CRD) by running the following command:

```
$ oc get crd | grep backup
```

Example output

```
backups.config.openshift.io 2023-10-25T13:32:43Z
etcdbackups.operator.openshift.io 2023-10-25T13:32:04Z
```

6.1.3.1. Creating a single automated etcd backup

Follow these steps to create a single etcd backup by creating and applying a custom resource (CR).

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have access to the OpenShift CLI (**oc**).

Procedure

- If dynamically-provisioned storage is available, complete the following steps to create a single automated etcd backup:
 - a. Create a persistent volume claim (PVC) named **etcd-backup-pvc.yaml** with contents such as the following example:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: etcd-backup-pvc
  namespace: openshift-etcd
```

```
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 200Gi 1
  volumeMode: Filesystem
```

- 1** The amount of storage available to the PVC. Adjust this value for your requirements.

- b. Apply the PVC by running the following command:

```
$ oc apply -f etcd-backup-pvc.yaml
```

- c. Verify the creation of the PVC by running the following command:

```
$ oc get pvc
```

Example output

NAME	STATUS	VOLUME	CAPACITY	ACCESS MODES
STORAGECLASS	AGE			
etcd-backup-pvc	Bound			51s



NOTE

Dynamic PVCs stay in the **Pending** state until they are mounted.

- d. Create a CR file named **etcd-single-backup.yaml** with contents such as the following example:

```
apiVersion: operator.openshift.io/v1alpha1
kind: EtcdBackup
metadata:
  name: etcd-single-backup
  namespace: openshift-etcd
spec:
  pvcName: etcd-backup-pvc 1
```

- 1** The name of the PVC to save the backup to. Adjust this value according to your environment.

- e. Apply the CR to start a single backup:

```
$ oc apply -f etcd-single-backup.yaml
```

- If dynamically-provisioned storage is not available, complete the following steps to create a single automated etcd backup:
 - a. Create a **StorageClass** CR file named **etcd-backup-local-storage.yaml** with the following contents:

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: etcd-backup-local-storage
provisioner: kubernetes.io/no-provisioner
volumeBindingMode: Immediate

```

- b. Apply the **StorageClass** CR by running the following command:

```
$ oc apply -f etcd-backup-local-storage.yaml
```

- c. Create a PV named **etcd-backup-pv-fs.yaml** with contents such as the following example:

```

apiVersion: v1
kind: PersistentVolume
metadata:
  name: etcd-backup-pv-fs
spec:
  capacity:
    storage: 100Gi ❶
  volumeMode: Filesystem
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Retain
  storageClassName: etcd-backup-local-storage
  local:
    path: /mnt
  nodeAffinity:
    required:
      nodeSelectorTerms:
        - matchExpressions:
            - key: kubernetes.io/hostname
              operator: In
              values:
                - <example_master_node> ❷

```

❶ The amount of storage available to the PV. Adjust this value for your requirements.

❷ Replace this value with the node to attach this PV to.

- d. Verify the creation of the PV by running the following command:

```
$ oc get pv
```

Example output

NAME	CAPACITY	ACCESS MODES	RECLAIM POLICY	STATUS
CLAIM	STORAGECLASS	REASON	AGE	
etcd-backup-pv-fs	100Gi	RWO	Retain	Available
local-storage	10s			etcd-backup-

- e. Create a PVC named **etcd-backup-pvc.yaml** with contents such as the following example:

■

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: etcd-backup-pvc
  namespace: openshift-etcd
spec:
  accessModes:
    - ReadWriteOnce
  volumeMode: Filesystem
  resources:
    requests:
      storage: 10Gi 1
```

- 1** The amount of storage available to the PVC. Adjust this value for your requirements.

f. Apply the PVC by running the following command:

```
$ oc apply -f etcd-backup-pvc.yaml
```

g. Create a CR file named **etcd-single-backup.yaml** with contents such as the following example:

```
apiVersion: operator.openshift.io/v1alpha1
kind: EtcdBackup
metadata:
  name: etcd-single-backup
  namespace: openshift-etcd
spec:
  pvcName: etcd-backup-pvc 1
```

- 1** The name of the persistent volume claim (PVC) to save the backup to. Adjust this value according to your environment.

h. Apply the CR to start a single backup:

```
$ oc apply -f etcd-single-backup.yaml
```

6.1.3.2. Creating recurring automated etcd backups

Follow these steps to create automated recurring backups of etcd.

Use dynamically-provisioned storage to keep the created etcd backup data in a safe, external location if possible. If dynamically-provisioned storage is not available, consider storing the backup data on an NFS share to make backup recovery more accessible.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have access to the OpenShift CLI (**oc**).

Procedure

1. If dynamically-provisioned storage is available, complete the following steps to create automated recurring backups:
 - a. Create a persistent volume claim (PVC) named **etcd-backup-pvc.yaml** with contents such as the following example:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: etcd-backup-pvc
  namespace: openshift-etcd
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 200Gi 1
  volumeMode: Filesystem
  storageClassName: etcd-backup-local-storage
```

- 1** The amount of storage available to the PVC. Adjust this value for your requirements.

NOTE

Each of the following providers require changes to the **accessModes** and **storageClassName** keys:

Provider	accessModes value	storageClassName value
AWS with the versioned-installer-efc_operator-ci profile	- ReadWriteMany	efs-sc
Google Cloud	- ReadWriteMany	filestore-csi
Microsoft Azure	- ReadWriteMany	azurefile-csi

- b. Apply the PVC by running the following command:

```
$ oc apply -f etcd-backup-pvc.yaml
```

- c. Verify the creation of the PVC by running the following command:

```
$ oc get pvc
```

Example output

NAME	STATUS	VOLUME	CAPACITY	ACCESS MODES
STORAGECLASS	AGE			
etcd-backup-pvc	Bound			51s

**NOTE**

Dynamic PVCs stay in the **Pending** state until they are mounted.

2. If dynamically-provisioned storage is unavailable, create a local storage PVC by completing the following steps:

**WARNING**

If you delete or otherwise lose access to the node that contains the stored backup data, you can lose data.

- a. Create a **StorageClass** CR file named **etcd-backup-local-storage.yaml** with the following contents:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: etcd-backup-local-storage
provisioner: kubernetes.io/no-provisioner
volumeBindingMode: Immediate
```

- b. Apply the **StorageClass** CR by running the following command:

```
$ oc apply -f etcd-backup-local-storage.yaml
```

- c. Create a PV named **etcd-backup-pv-fs.yaml** from the applied **StorageClass** with contents such as the following example:

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: etcd-backup-pv-fs
spec:
  capacity:
    storage: 100Gi 1
  volumeMode: Filesystem
  accessModes:
    - ReadWriteMany
  persistentVolumeReclaimPolicy: Delete
  storageClassName: etcd-backup-local-storage
  local:
    path: /mnt/
  nodeAffinity:
```

```

required:
  nodeSelectorTerms:
  - matchExpressions:
    - key: kubernetes.io/hostname
      operator: In
      values:
      - <example_master_node> 2

```

- 1 The amount of storage available to the PV. Adjust this value for your requirements.
- 2 Replace this value with the master node to attach this PV to.

TIP

Run the following command to list the available nodes:

```
$ oc get nodes
```

- d. Verify the creation of the PV by running the following command:

```
$ oc get pv
```

Example output

NAME	CAPACITY	ACCESS MODES	RECLAIM POLICY	STATUS
CLAIM STORAGECLASS	REASON	AGE		
etcd-backup-pv-fs	100Gi	RWX	Delete	Available
local-storage	10s			etcd-backup-

- e. Create a PVC named **etcd-backup-pvc.yaml** with contents such as the following example:

```

kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: etcd-backup-pvc
spec:
  accessModes:
  - ReadWriteMany
  volumeMode: Filesystem
  resources:
    requests:
      storage: 10Gi 1
  storageClassName: etcd-backup-local-storage

```

- 1 The amount of storage available to the PVC. Adjust this value for your requirements.

- f. Apply the PVC by running the following command:

```
$ oc apply -f etcd-backup-pvc.yaml
```

3. Create a custom resource definition (CRD) file named **etcd-recurring-backups.yaml**. The contents of the created CRD define the schedule and retention type of automated backups.

- For the default retention type of **RetentionNumber** with 15 retained backups, use contents such as the following example:

```
apiVersion: config.openshift.io/v1alpha1
kind: Backup
metadata:
  name: etcd-recurring-backup
spec:
  etcd:
    schedule: "20 4 * * *" 1
    timeZone: "UTC"
    pvcName: etcd-backup-pvc
```

- 1 The **CronTab** schedule for recurring backups. Adjust this value for your needs.

- To use retention based on the maximum number of backups, add the following key-value pairs to the **etcd** key:

```
spec:
  etcd:
    retentionPolicy:
      retentionType: RetentionNumber 1
      retentionNumber:
        maxNumberOfBackups: 5 2
```

- 1 The retention type. Defaults to **RetentionNumber** if unspecified.

- 2 The maximum number of backups to retain. Adjust this value for your needs. Defaults to 15 backups if unspecified.



WARNING

A known issue causes the number of retained backups to be one greater than the configured value.

- For retention based on the file size of backups, use the following:

```
spec:
  etcd:
    retentionPolicy:
      retentionType: RetentionSize
      retentionSize:
        maxSizeOfBackupsGb: 20 1
```

- 1 The maximum file size of the retained backups in gigabytes. Adjust this value for your needs. Defaults to 10 GB if unspecified.



WARNING

A known issue causes the maximum size of retained backups to be up to 10 GB greater than the configured value.

4. Create the cron job defined by the CRD by running the following command:

```
$ oc create -f etcd-recurring-backup.yaml
```

5. To find the created cron job, run the following command:

```
$ oc get cronjob -n openshift-etcd
```

6.2. REPLACING AN UNHEALTHY ETCD MEMBER

This document describes the process to replace a single unhealthy etcd member.

This process depends on whether the etcd member is unhealthy because the machine is not running or the node is not ready, or whether it is unhealthy because the etcd pod is crashlooping.



NOTE

If you have lost the majority of your control plane hosts, follow the disaster recovery procedure to [restore to a previous cluster state](#) instead of this procedure.

If the control plane certificates are not valid on the member being replaced, then you must follow the procedure to [recover from expired control plane certificates](#) instead of this procedure.

If a control plane node is lost and a new one is created, the etcd cluster Operator handles generating the new TLS certificates and adding the node as an etcd member.

6.2.1. Prerequisites

- Take an [etcd backup](#) prior to replacing an unhealthy etcd member.

6.2.2. Identifying an unhealthy etcd member

You can identify if your cluster has an unhealthy etcd member.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.

- You have taken an etcd backup. For more information, see "Backing up etcd data".

Procedure

1. Check the status of the **EtcdMembersAvailable** status condition using the following command:

```
$ oc get etcd -o=jsonpath='{range .items[0].status.conditions[?(@.type=="EtcdMembersAvailable")]}{.message}{"\n"}{end}'
```

2. Review the output:

```
2 of 3 members are available, ip-10-0-131-183.ec2.internal is unhealthy
```

This example output shows that the **ip-10-0-131-183.ec2.internal** etcd member is unhealthy.

6.2.3. Determining the state of the unhealthy etcd member

The steps to replace an unhealthy etcd member depend on which of the following states your etcd member is in:

- The machine is not running or the node is not ready
- The etcd pod is crashlooping

This procedure determines which state your etcd member is in. This enables you to know which procedure to follow to replace the unhealthy etcd member.



NOTE

If you are aware that the machine is not running or the node is not ready, but you expect it to return to a healthy state soon, then you do not need to perform a procedure to replace the etcd member. The etcd cluster Operator will automatically sync when the machine or node returns to a healthy state.

Prerequisites

- You have access to the cluster as a user with the **cluster-admin** role.
- You have identified an unhealthy etcd member.

Procedure

1. Determine if the **machine is not running**

```
$ oc get machines -A -o=jsonpath='{range .items[*]}{@.status.nodeRef.name}{"\t"}{@.status.providerStatus.instanceState}{"\n"}' | grep -v running
```

Example output

```
ip-10-0-131-183.ec2.internal stopped 1
```

1

This output lists the node and the status of the node's machine. If the status is anything other than **running**, then the **machine is not running**

If the **machine is not running** then follow the *Replacing an unhealthy etcd member whose machine is not running or whose node is not ready* procedure.

2. Determine if the **node is not ready**.

If either of the following scenarios are true, then the **node is not ready**.

- If the machine is running, then check whether the node is unreachable:

```
$ oc get nodes -o jsonpath='{range .items[*]}{"\n"}{.metadata.name}{"\t"}{range .spec.taints[*]}{.key}{" "}" | grep unreachable
```

Example output

```
ip-10-0-131-183.ec2.internal node-role.kubernetes.io/master
node.kubernetes.io/unreachable node.kubernetes.io/unreachable 1
```

- 1 If the node is listed with an **unreachable** taint, then the **node is not ready**.

- If the node is still reachable, then check whether the node is listed as **NotReady**:

```
$ oc get nodes -l node-role.kubernetes.io/master | grep "NotReady"
```

Example output

```
ip-10-0-131-183.ec2.internal NotReady master 122m v1.33.4 1
```

- 1 If the node is listed as **NotReady**, then the **node is not ready**.

If the **node is not ready**, then follow the *Replacing an unhealthy etcd member whose machine is not running or whose node is not ready* procedure.

3. Determine if the **etcd pod is crashlooping**

If the machine is running and the node is ready, then check whether the etcd pod is crashlooping.

- Verify that all control plane nodes are listed as **Ready**:

```
$ oc get nodes -l node-role.kubernetes.io/master
```

Example output

NAME	STATUS	ROLES	AGE	VERSION
ip-10-0-131-183.ec2.internal	Ready	master	6h13m	v1.33.4
ip-10-0-164-97.ec2.internal	Ready	master	6h13m	v1.33.4
ip-10-0-154-204.ec2.internal	Ready	master	6h13m	v1.33.4

- Check whether the status of an etcd pod is either **Error** or **CrashloopBackoff**:

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd
```

Example output

etcd-ip-10-0-131-183.ec2.internal	2/3	Error	7	6h9m	1
etcd-ip-10-0-164-97.ec2.internal	3/3	Running	0	6h6m	
etcd-ip-10-0-154-204.ec2.internal	3/3	Running	0	6h6m	

1 Since this status of this pod is **Error**, then the **etcd pod is crashlooping**

If the **etcd pod is crashlooping** then follow the *Replacing an unhealthy etcd member whose etcd pod is crashlooping* procedure.

6.2.4. Replacing the unhealthy etcd member

Depending on the state of your unhealthy etcd member, use one of the following procedures:

- [Replacing an unhealthy etcd member whose machine is not running or whose node is not ready](#)
- [Installing a primary control plane node on an unhealthy cluster](#)
- [Replacing an unhealthy etcd member whose etcd pod is crashlooping](#)
- [Replacing an unhealthy stopped baremetal etcd member](#)

6.2.4.1. Replacing an unhealthy etcd member whose machine is not running or whose node is not ready

This procedure details the steps to replace an etcd member that is unhealthy either because the machine is not running or because the node is not ready.



NOTE

If your cluster uses a control plane machine set, see "Recovering a degraded etcd Operator" in "Troubleshooting the control plane machine set" for an etcd recovery procedure.

Prerequisites

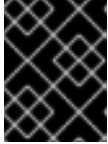
- You have identified the unhealthy etcd member.
- You have verified that either the machine is not running or the node is not ready.



IMPORTANT

You must wait if you power off other control plane nodes. The control plane nodes must remain powered off until the replacement of an unhealthy etcd member is complete.

- You have access to the cluster as a user with the **cluster-admin** role.
- You have taken an etcd backup.



IMPORTANT

Before you perform this procedure, take an etcd backup so that you can restore your cluster if you experience any issues.

Procedure

1. Remove the unhealthy member.
 - a. Choose a pod that is not on the affected node:
In a terminal that has access to the cluster as a **cluster-admin** user, run the following command:

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd
```

Example output

```
etcd-ip-10-0-131-183.ec2.internal    3/3    Running    0        123m
etcd-ip-10-0-164-97.ec2.internal    3/3    Running    0        123m
etcd-ip-10-0-154-204.ec2.internal    3/3    Running    0        124m
```

- b. Connect to the running etcd container, passing in the name of a pod that is not on the affected node:
In a terminal that has access to the cluster as a **cluster-admin** user, run the following command:

```
$ oc rsh -n openshift-etcd etcd-ip-10-0-154-204.ec2.internal
```

- c. View the member list:

```
sh-4.2# etcdctl member list -w table
```

Example output

```
+-----+-----+-----+-----+-----+
+-----+
| ID      | STATUS | NAME                | PEER ADDRS      | CLIENT
ADDRS    |
+-----+-----+-----+-----+-----+
+-----+
| 6fc1e7c9db35841d | started | ip-10-0-131-183.ec2.internal | https://10.0.131.183:2380 |
https://10.0.131.183:2379 |
| 757b6793e2408b6c | started | ip-10-0-164-97.ec2.internal | https://10.0.164.97:2380 |
https://10.0.164.97:2379 |
| ca8c2990a0aa29d1 | started | ip-10-0-154-204.ec2.internal | https://10.0.154.204:2380 |
https://10.0.154.204:2379 |
+-----+-----+-----+-----+-----+
+-----+
```

Take note of the ID and the name of the unhealthy etcd member because these values are needed later in the procedure. The **\$ etcdctl endpoint health** command will list the removed member until the procedure of replacement is finished and a new member is added.

- d. Remove the unhealthy etcd member by providing the ID to the **etcdctl member remove** command:

```
sh-4.2# etcdctl member remove 6fc1e7c9db35841d
```

Example output

```
Member 6fc1e7c9db35841d removed from cluster ead669ce1fbfb346
```

- e. View the member list again and verify that the member was removed:

```
sh-4.2# etcdctl member list -w table
```

Example output

```
+-----+-----+-----+-----+-----+
+-----+
| ID      | STATUS | NAME          | PEER ADDRS      | CLIENT
+-----+-----+-----+-----+-----+
| 757b6793e2408b6c | started | ip-10-0-164-97.ec2.internal | https://10.0.164.97:2380 |
| https://10.0.164.97:2379 |
| ca8c2990a0aa29d1 | started | ip-10-0-154-204.ec2.internal | https://10.0.154.204:2380 |
| https://10.0.154.204:2379 |
+-----+-----+-----+-----+-----+
+-----+
```

You can now exit the node shell.

2. Turn off the quorum guard by entering the following command:

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides": {"useUnsupportedUnsafeNonHANonProductionUnstableEtcd": true}}}'
```

This command ensures that you can successfully re-create secrets and roll out the static pods.



IMPORTANT

After you turn off the quorum guard, the cluster might be unreachable for a short time while the remaining etcd instances reboot to reflect the configuration change.



NOTE

etcd cannot tolerate any additional member failure when running with two members. Restarting either remaining member breaks the quorum and causes downtime in your cluster. The quorum guard protects etcd from restarts due to configuration changes that could cause downtime, so it must be disabled to complete this procedure.

3. Delete the affected node by running the following command:

■

```
$ oc delete node <node_name>
```

Example command

```
$ oc delete node ip-10-0-131-183.ec2.internal
```

4. Remove the old secrets for the unhealthy etcd member that was removed.

- a. List the secrets for the unhealthy etcd member that was removed.

```
$ oc get secrets -n openshift-etcd | grep ip-10-0-131-183.ec2.internal 1
```

- 1** Pass in the name of the unhealthy etcd member that you took note of earlier in this procedure.

There is a peer, serving, and metrics secret as shown in the following output:

Example output

```
etcd-peer-ip-10-0-131-183.ec2.internal      kubernetes.io/tls      2      47m
etcd-serving-ip-10-0-131-183.ec2.internal  kubernetes.io/tls      2      47m
etcd-serving-metrics-ip-10-0-131-183.ec2.internal kubernetes.io/tls      2
47m
```

- b. Delete the secrets for the unhealthy etcd member that was removed.

- i. Delete the peer secret:

```
$ oc delete secret -n openshift-etcd etcd-peer-ip-10-0-131-183.ec2.internal
```

- ii. Delete the serving secret:

```
$ oc delete secret -n openshift-etcd etcd-serving-ip-10-0-131-183.ec2.internal
```

- iii. Delete the metrics secret:

```
$ oc delete secret -n openshift-etcd etcd-serving-metrics-ip-10-0-131-183.ec2.internal
```

5. Check whether a control plane machine set exists by entering the following command:

```
$ oc -n openshift-machine-api get controlplanemachineset
```

- If the control plane machine set exists, delete and re-create the control plane machine. After this machine is re-created, a new revision is forced and etcd scales up automatically. For more information, see "Replacing an unhealthy etcd member whose machine is not running or whose node is not ready".
If you are running installer-provisioned infrastructure, or you used the Machine API to create your machines, follow these steps. Otherwise, you must create the new control plane by using the same method that was used to originally create it.

- a. Obtain the machine for the unhealthy member.

In a terminal that has access to the cluster as a **cluster-admin** user, run the following command:

```
$ oc get machines -n openshift-machine-api -o wide
```

Example output

```
NAME                                PHASE  TYPE    REGION  ZONE    AGE
NODE                                PROVIDERID  STATE
clustername-8qw5l-master-0          Running m4.xlarge us-east-1 us-east-1a
3h37m ip-10-0-131-183.ec2.internal  aws:///us-east-1a/i-0ec2782f8287dfb7e
stopped 1
clustername-8qw5l-master-1          Running m4.xlarge us-east-1 us-east-1b
3h37m ip-10-0-154-204.ec2.internal  aws:///us-east-1b/i-096c349b700a19631
running
clustername-8qw5l-master-2          Running m4.xlarge us-east-1 us-east-1c
3h37m ip-10-0-164-97.ec2.internal  aws:///us-east-1c/i-02626f1dba9ed5bba
running
clustername-8qw5l-worker-us-east-1a-wbtgd Running m4.large  us-east-1 us-
east-1a 3h28m ip-10-0-129-226.ec2.internal  aws:///us-east-1a/i-
010ef6279b4662ced running
clustername-8qw5l-worker-us-east-1b-lrdxb Running m4.large  us-east-1 us-
east-1b 3h28m ip-10-0-144-248.ec2.internal  aws:///us-east-1b/i-
0cb45ac45a166173b running
clustername-8qw5l-worker-us-east-1c-pkg26 Running m4.large  us-east-1 us-
east-1c 3h28m ip-10-0-170-181.ec2.internal  aws:///us-east-1c/i-
06861c00007751b0a running
```

- 1** This is the control plane machine for the unhealthy node, **ip-10-0-131-183.ec2.internal**.

b. Delete the machine of the unhealthy member:

```
$ oc delete machine -n openshift-machine-api clustername-8qw5l-master-0 1
```

- 1** Specify the name of the control plane machine for the unhealthy node.

A new machine is automatically provisioned after deleting the machine of the unhealthy member.

c. Verify that a new machine was created:

```
$ oc get machines -n openshift-machine-api -o wide
```

Example output

```
NAME                                PHASE  TYPE    REGION  ZONE
AGE  NODE                                PROVIDERID  STATE
clustername-8qw5l-master-1          Running m4.xlarge us-east-1 us-east-
1b 3h37m ip-10-0-154-204.ec2.internal  aws:///us-east-1b/i-096c349b700a19631
running
clustername-8qw5l-master-2          Running m4.xlarge us-east-1 us-east-
```

```

1c 3h37m ip-10-0-164-97.ec2.internal aws:///us-east-1c/i-02626f1dba9ed5bba
running
clustername-8qw5l-master-3      Provisioning m4.xlarge us-east-1 us-east-
1a 85s ip-10-0-133-53.ec2.internal aws:///us-east-1a/i-015b0888fe17bc2c8
running 1
clustername-8qw5l-worker-us-east-1a-wbtgd Running m4.large us-east-1
us-east-1a 3h28m ip-10-0-129-226.ec2.internal aws:///us-east-1a/i-
010ef6279b4662ced running
clustername-8qw5l-worker-us-east-1b-lrdxb Running m4.large us-east-1 us-
east-1b 3h28m ip-10-0-144-248.ec2.internal aws:///us-east-1b/i-
0cb45ac45a166173b running
clustername-8qw5l-worker-us-east-1c-pkg26 Running m4.large us-east-1
us-east-1c 3h28m ip-10-0-170-181.ec2.internal aws:///us-east-1c/i-
06861c00007751b0a running

```

- 1** The new machine, **clustername-8qw5l-master-3** is being created and is ready once the phase changes from **Provisioning** to **Running**.

It might take a few minutes for the new machine to be created. The etcd cluster Operator automatically syncs when the machine or node returns to a healthy state.



NOTE

Verify the subnet IDs that you are using for your machine sets to ensure that they end up in the correct availability zone.

- If the control plane machine set does not exist, delete and re-create the control plane machine. After this machine is re-created, a new revision is forced and etcd scales up automatically.
If you are running installer-provisioned infrastructure, or you used the Machine API to create your machines, follow these steps. Otherwise, you must create the new control plane by using the same method that was used to originally create it.

- Obtain the machine for the unhealthy member.

In a terminal that has access to the cluster as a **cluster-admin** user, run the following command:

```
$ oc get machines -n openshift-machine-api -o wide
```

Example output

```

NAME                                PHASE  TYPE    REGION  ZONE    AGE
NODE                                PROVIDERID  STATE
clustername-8qw5l-master-0          Running m4.xlarge us-east-1 us-east-1a
3h37m ip-10-0-131-183.ec2.internal aws:///us-east-1a/i-0ec2782f8287dfb7e
stopped 1
clustername-8qw5l-master-1          Running m4.xlarge us-east-1 us-east-1b
3h37m ip-10-0-154-204.ec2.internal aws:///us-east-1b/i-096c349b700a19631
running
clustername-8qw5l-master-2          Running m4.xlarge us-east-1 us-east-1c
3h37m ip-10-0-164-97.ec2.internal aws:///us-east-1c/i-02626f1dba9ed5bba
running
clustername-8qw5l-worker-us-east-1a-wbtgd Running m4.large us-east-1 us-

```

```

east-1a 3h28m ip-10-0-129-226.ec2.internal aws:///us-east-1a/i-
010ef6279b4662ced running
clustername-8qw5l-worker-us-east-1b-lrdxb Running m4.large us-east-1 us-
east-1b 3h28m ip-10-0-144-248.ec2.internal aws:///us-east-1b/i-
0cb45ac45a166173b running
clustername-8qw5l-worker-us-east-1c-pkg26 Running m4.large us-east-1 us-
east-1c 3h28m ip-10-0-170-181.ec2.internal aws:///us-east-1c/i-
06861c00007751b0a running

```

- 1 This is the control plane machine for the unhealthy node, **ip-10-0-131-183.ec2.internal**.

- b. Save the machine configuration to a file on your file system:

```

$ oc get machine clustername-8qw5l-master-0 \ 1
-n openshift-machine-api \
-o yaml \
> new-master-machine.yaml

```

- 1 Specify the name of the control plane machine for the unhealthy node.

- c. Edit the **new-master-machine.yaml** file that was created in the previous step to assign a new name and remove unnecessary fields.

- i. Remove the entire **status** section:

```

status:
  addresses:
    - address: 10.0.131.183
      type: InternalIP
    - address: ip-10-0-131-183.ec2.internal
      type: InternalDNS
    - address: ip-10-0-131-183.ec2.internal
      type: Hostname
  lastUpdated: "2020-04-20T17:44:29Z"
  nodeRef:
    kind: Node
    name: ip-10-0-131-183.ec2.internal
    uid: acca4411-af0d-4387-b73e-52b2484295ad
  phase: Running
  providerStatus:
    apiVersion: awsproviderconfig.openshift.io/v1beta1
    conditions:
      - lastProbeTime: "2020-04-20T16:53:50Z"
        lastTransitionTime: "2020-04-20T16:53:50Z"
        message: machine successfully created
        reason: MachineCreationSucceeded
        status: "True"
        type: MachineCreation
    instanceId: i-0fdb85790d76d0c3f
    instanceState: stopped
    kind: AWSMachineProviderStatus

```

- ii. Change the **metadata.name** field to a new name.
Keep the same base name as the old machine and change the ending number to the next available number. In this example, **clustername-8qw5l-master-0** is changed to **clustername-8qw5l-master-3**.

For example:

```
apiVersion: machine.openshift.io/v1beta1
kind: Machine
metadata:
  ...
  name: clustername-8qw5l-master-3
  ...
```

- iii. Remove the **spec.providerID** field:

```
providerID: aws:///us-east-1 a/i-0fdb85790d76d0c3f
```

- d. Delete the machine of the unhealthy member:

```
$ oc delete machine -n openshift-machine-api clustername-8qw5l-master-0 1
```

- 1** Specify the name of the control plane machine for the unhealthy node.

- e. Verify that the machine was deleted:

```
$ oc get machines -n openshift-machine-api -o wide
```

Example output

```
NAME                                PHASE  TYPE    REGION  ZONE    AGE
NODE                                PROVIDERID      STATE
clustername-8qw5l-master-1          Running m4.xlarge us-east-1 us-east-1b
3h37m ip-10-0-154-204.ec2.internal  aws:///us-east-1b/i-096c349b700a19631
running
clustername-8qw5l-master-2          Running m4.xlarge us-east-1 us-east-1c
3h37m ip-10-0-164-97.ec2.internal  aws:///us-east-1c/i-02626f1dba9ed5bba
running
clustername-8qw5l-worker-us-east-1a-wbtgd Running m4.large  us-east-1 us-
east-1a 3h28m ip-10-0-129-226.ec2.internal  aws:///us-east-1a/i-
010ef6279b4662ced running
clustername-8qw5l-worker-us-east-1b-lrdxb Running m4.large  us-east-1 us-
east-1b 3h28m ip-10-0-144-248.ec2.internal  aws:///us-east-1b/i-
0cb45ac45a166173b running
clustername-8qw5l-worker-us-east-1c-pkg26 Running m4.large  us-east-1 us-
east-1c 3h28m ip-10-0-170-181.ec2.internal  aws:///us-east-1c/i-
06861c00007751b0a running
```

- f. Create the new machine by using the **new-master-machine.yaml** file:

```
$ oc apply -f new-master-machine.yaml
```

- g. Verify that the new machine was created:

```
$ oc get machines -n openshift-machine-api -o wide
```

Example output

```
NAME                                PHASE    TYPE    REGION    ZONE
AGE  NODE                                PROVIDERID  STATE
clustername-8qw5l-master-1          Running   m4.xlarge us-east-1 us-east-
1b 3h37m ip-10-0-154-204.ec2.internal  aws:///us-east-1b/i-096c349b700a19631
running
clustername-8qw5l-master-2          Running   m4.xlarge us-east-1 us-east-
1c 3h37m ip-10-0-164-97.ec2.internal  aws:///us-east-1c/i-02626f1dba9ed5bba
running
clustername-8qw5l-master-3          Provisioning m4.xlarge us-east-1 us-east-
1a 85s ip-10-0-133-53.ec2.internal  aws:///us-east-1a/i-015b0888fe17bc2c8
running 1
clustername-8qw5l-worker-us-east-1a-wbtgd Running   m4.large  us-east-1
us-east-1a 3h28m ip-10-0-129-226.ec2.internal  aws:///us-east-1a/i-
010ef6279b4662ced running
clustername-8qw5l-worker-us-east-1b-lrdxb Running   m4.large  us-east-1 us-
east-1b 3h28m ip-10-0-144-248.ec2.internal  aws:///us-east-1b/i-
0cb45ac45a166173b running
clustername-8qw5l-worker-us-east-1c-pkg26 Running   m4.large  us-east-1
us-east-1c 3h28m ip-10-0-170-181.ec2.internal  aws:///us-east-1c/i-
06861c00007751b0a running
```

- 1** The new machine, **clustername-8qw5l-master-3** is being created and is ready once the phase changes from **Provisioning** to **Running**.

It might take a few minutes for the new machine to be created. The etcd cluster Operator automatically syncs when the machine or node returns to a healthy state.

6. Turn the quorum guard back on by entering the following command:

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides": null}}'
```

7. You can verify that the **unsupportedConfigOverrides** section is removed from the object by entering this command:

```
$ oc get etcd/cluster -oyaml
```

8. If you are using single-node OpenShift, restart the node. Otherwise, you might experience the following error in the etcd cluster Operator:

Example output

```
EtcdCertSignerControllerDegraded: [Operation cannot be fulfilled on secrets "etcd-peer-sno-0": the object has been modified; please apply your changes to the latest version and try again, Operation cannot be fulfilled on secrets "etcd-serving-sno-0": the object has been modified; please apply your changes to the latest version and try again, Operation cannot be fulfilled on secrets "etcd-serving-metrics-sno-0": the object has been modified; please apply your changes to the latest version and try again]
```


Verification

1. Verify that all etcd pods are running properly.

In a terminal that has access to the cluster as a **cluster-admin** user, run the following command:

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd
```

Example output

```
etcd-ip-10-0-133-53.ec2.internal    3/3    Running    0      7m49s
etcd-ip-10-0-164-97.ec2.internal    3/3    Running    0      123m
etcd-ip-10-0-154-204.ec2.internal    3/3    Running    0      124m
```

If the output from the previous command only lists two pods, you can manually force an etcd redeployment. In a terminal that has access to the cluster as a **cluster-admin** user, run the following command:

```
$ oc patch etcd cluster -p='{ "spec": { "forceRedeploymentReason": "recovery-"$( date --rfc-3339=ns )"' }' --type=merge ❶
```

- ❶ The **forceRedeploymentReason** value must be unique, which is why a timestamp is appended.

2. Verify that there are exactly three etcd members.

- a. Connect to the running etcd container, passing in the name of a pod that was not on the affected node:

In a terminal that has access to the cluster as a **cluster-admin** user, run the following command:

```
$ oc rsh -n openshift-etcd etcd-ip-10-0-154-204.ec2.internal
```

- b. View the member list:

```
sh-4.2# etcdctl member list -w table
```

Example output

```
+-----+-----+-----+-----+-----+
+-----+
| ID      | STATUS | NAME                | PEER ADDRS      | CLIENT
```

ID	STATUS	NAME	PEER ADDRS	CLIENT
5eb0d6b8ca24730c	started	ip-10-0-133-53.ec2.internal	https://10.0.133.53:2380 https://10.0.133.53:2379	
757b6793e2408b6c	started	ip-10-0-164-97.ec2.internal	https://10.0.164.97:2380 https://10.0.164.97:2379	
ca8c2990a0aa29d1	started	ip-10-0-154-204.ec2.internal	https://10.0.154.204:2380 https://10.0.154.204:2379	

```
+-----+-----+-----+-----+-----+
+-----+
```

If the output from the previous command lists more than three etcd members, you must carefully remove the unwanted member.



WARNING

Be sure to remove the correct etcd member; removing a good etcd member might lead to quorum loss.

Additional resources

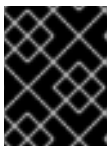
- [Recovering a degraded etcd Operator](#)
- [Installing a primary control plane node on an unhealthy cluster](#)

6.2.4.2. Replacing an unhealthy etcd member whose etcd pod is crashlooping

This procedure details the steps to replace an etcd member that is unhealthy because the etcd pod is crashlooping.

Prerequisites

- You have identified the unhealthy etcd member.
- You have verified that the etcd pod is crashlooping.
- You have access to the cluster as a user with the **cluster-admin** role.
- You have taken an etcd backup.



IMPORTANT

It is important to take an etcd backup before performing this procedure so that your cluster can be restored if you encounter any issues.

Procedure

1. Stop the crashlooping etcd pod.
 - a. Debug the node that is crashlooping.
In a terminal that has access to the cluster as a **cluster-admin** user, run the following command:

```
$ oc debug node/ip-10-0-131-183.ec2.internal 1
```

1. Replace this with the name of the unhealthy node.

- b. Change your root directory to **/host**:

```
sh-4.2# chroot /host
```

- c. Move the existing etcd pod file out of the kubelet manifest directory:

```
sh-4.2# mkdir /var/lib/etcd-backup
```

```
sh-4.2# mv /etc/kubernetes/manifests/etcd-pod.yaml /var/lib/etcd-backup/
```

- d. Move the etcd data directory to a different location:

```
sh-4.2# mv /var/lib/etcd/ /tmp
```

You can now exit the node shell.

2. Remove the unhealthy member.

- a. Choose a pod that is *not* on the affected node.

In a terminal that has access to the cluster as a **cluster-admin** user, run the following command:

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd
```

Example output

```
etcd-ip-10-0-131-183.ec2.internal    2/3    Error    7        6h9m
etcd-ip-10-0-164-97.ec2.internal    3/3    Running  0        6h6m
etcd-ip-10-0-154-204.ec2.internal    3/3    Running  0        6h6m
```

- b. Connect to the running etcd container, passing in the name of a pod that is not on the affected node.

In a terminal that has access to the cluster as a **cluster-admin** user, run the following command:

```
$ oc rsh -n openshift-etcd etcd-ip-10-0-154-204.ec2.internal
```

- c. View the member list:

```
sh-4.2# etcdctl member list -w table
```

Example output

```
+-----+-----+-----+-----+-----+
+-----+
| ID      | STATUS | NAME                | PEER ADDRS      | CLIENT
ADDRS    |
+-----+-----+-----+-----+-----+
+-----+
| 62bcf33650a7170a | started | ip-10-0-131-183.ec2.internal | https://10.0.131.183:2380 |
https://10.0.131.183:2379 |
| b78e2856655bc2eb | started | ip-10-0-164-97.ec2.internal | https://10.0.164.97:2380 |
https://10.0.164.97:2379 |
| d022e10b498760d5 | started | ip-10-0-154-204.ec2.internal | https://10.0.154.204:2380
```

```
| https://10.0.154.204:2379 |
```

```
+-----+-----+-----+-----+-----+
-----+
```

Take note of the ID and the name of the unhealthy etcd member, because these values are needed later in the procedure.

- d. Remove the unhealthy etcd member by providing the ID to the **etcdctl member remove** command:

```
sh-4.2# etcdctl member remove 62bcf33650a7170a
```

Example output

```
Member 62bcf33650a7170a removed from cluster ead669ce1fbfb346
```

- e. View the member list again and verify that the member was removed:

```
sh-4.2# etcdctl member list -w table
```

Example output

```
+-----+-----+-----+-----+-----+
-----+
| ID      | STATUS | NAME                | PEER ADDRS      | CLIENT
ADDRS    |
+-----+-----+-----+-----+-----+
-----+
| b78e2856655bc2eb | started | ip-10-0-164-97.ec2.internal | https://10.0.164.97:2380 |
https://10.0.164.97:2379 |
| d022e10b498760d5 | started | ip-10-0-154-204.ec2.internal | https://10.0.154.204:2380 |
https://10.0.154.204:2379 |
+-----+-----+-----+-----+-----+
-----+
```

You can now exit the node shell.

3. Turn off the quorum guard by entering the following command:

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides": {"useUnsupportedUnsafeNonHANonProductionUnstableEtcd": true}}}'
```

This command ensures that you can successfully re-create secrets and roll out the static pods.

4. Remove the old secrets for the unhealthy etcd member that was removed.

- a. List the secrets for the unhealthy etcd member that was removed.

```
$ oc get secrets -n openshift-etcd | grep ip-10-0-131-183.ec2.internal 1
```

- 1** Pass in the name of the unhealthy etcd member that you took note of earlier in this procedure.

There is a peer, serving, and metrics secret as shown in the following output:

Example output

```
etcd-peer-ip-10-0-131-183.ec2.internal    kubernetes.io/tls    2    47m
etcd-serving-ip-10-0-131-183.ec2.internal kubernetes.io/tls    2    47m
etcd-serving-metrics-ip-10-0-131-183.ec2.internal kubernetes.io/tls    2
47m
```

b. Delete the secrets for the unhealthy etcd member that was removed.

i. Delete the peer secret:

```
$ oc delete secret -n openshift-etcd etcd-peer-ip-10-0-131-183.ec2.internal
```

ii. Delete the serving secret:

```
$ oc delete secret -n openshift-etcd etcd-serving-ip-10-0-131-183.ec2.internal
```

iii. Delete the metrics secret:

```
$ oc delete secret -n openshift-etcd etcd-serving-metrics-ip-10-0-131-183.ec2.internal
```

5. Force etcd redeployment.

In a terminal that has access to the cluster as a **cluster-admin** user, run the following command:

```
$ oc patch etcd cluster -p='{ "spec": { "forceRedeploymentReason": "single-master-recovery-$( date --rfc-3339=ns )"' --type=merge 1
```

1 The **forceRedeploymentReason** value must be unique, which is why a timestamp is appended.

When the etcd cluster Operator performs a redeployment, it ensures that all control plane nodes have a functioning etcd pod.

6. Turn the quorum guard back on by entering the following command:

```
$ oc patch etcd/cluster --type=merge -p '{ "spec": { "unsupportedConfigOverrides": null } }
```

7. You can verify that the **unsupportedConfigOverrides** section is removed from the object by entering this command:

```
$ oc get etcd/cluster -oyaml
```

8. If you are using single-node OpenShift, restart the node. Otherwise, you might encounter the following error in the etcd cluster Operator:

Example output

```
EtdcCertSignerControllerDegraded: [Operation cannot be fulfilled on secrets "etcd-peer-sno-0": the object has been modified; please apply your changes to the latest version and try
```

again, Operation cannot be fulfilled on secrets "etcd-serving-sno-0": the object has been modified; please apply your changes to the latest version and try again, Operation cannot be fulfilled on secrets "etcd-serving-metrics-sno-0": the object has been modified; please apply your changes to the latest version and try again]

Verification

- Verify that the new member is available and healthy.
 - a. Connect to the running etcd container again.
In a terminal that has access to the cluster as a cluster-admin user, run the following command:

```
$ oc rsh -n openshift-etcd etcd-ip-10-0-154-204.ec2.internal
```

- b. Verify that all members are healthy:

```
sh-4.2# etcdctl endpoint health
```

Example output

```
https://10.0.131.183:2379 is healthy: successfully committed proposal: took = 16.671434ms
https://10.0.154.204:2379 is healthy: successfully committed proposal: took = 16.698331ms
https://10.0.164.97:2379 is healthy: successfully committed proposal: took = 16.621645ms
```

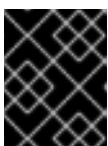
6.2.4.3. Replacing an unhealthy bare metal etcd member whose machine is not running or whose node is not ready

This procedure details the steps to replace a bare metal etcd member that is unhealthy either because the machine is not running or because the node is not ready.

If you are running installer-provisioned infrastructure or you used the Machine API to create your machines, follow these steps. Otherwise you must create the new control plane node using the same method that was used to originally create it.

Prerequisites

- You have identified the unhealthy bare metal etcd member.
- You have verified that either the machine is not running or the node is not ready.
- You have access to the cluster as a user with the **cluster-admin** role.
- You have taken an etcd backup.



IMPORTANT

You must take an etcd backup before performing this procedure so that your cluster can be restored if you encounter any issues.

Procedure

1. Verify and remove the unhealthy member.
 - a. Choose a pod that is *not* on the affected node:
In a terminal that has access to the cluster as a **cluster-admin** user, run the following command:

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd -o wide
```

Example output

```
etcd-openshift-control-plane-0 5/5 Running 11 3h56m 192.168.10.9 openshift-
control-plane-0 <none> <none>
etcd-openshift-control-plane-1 5/5 Running 0 3h54m 192.168.10.10 openshift-
control-plane-1 <none> <none>
etcd-openshift-control-plane-2 5/5 Running 0 3h58m 192.168.10.11 openshift-
control-plane-2 <none> <none>
```

- b. Connect to the running etcd container, passing in the name of a pod that is not on the affected node:
In a terminal that has access to the cluster as a **cluster-admin** user, run the following command:

```
$ oc rsh -n openshift-etcd etcd-openshift-control-plane-0
```

- c. View the member list:

```
sh-4.2# etcdctl member list -w table
```

Example output

```
+-----+-----+-----+-----+-----+
+-----+
| ID          | STATUS | NAME                | PEER ADDRS          | CLIENT
ADDRS        | IS LEARNER |                      |                      |
+-----+-----+-----+-----+-----+
+-----+
| 7a8197040a5126c8 | started | openshift-control-plane-2 | https://192.168.10.11:2380/ |
https://192.168.10.11:2379/ | false |
| 8d5abe9669a39192 | started | openshift-control-plane-1 | https://192.168.10.10:2380/ |
https://192.168.10.10:2379/ | false |
| cc3830a72fc357f9 | started | openshift-control-plane-0 | https://192.168.10.9:2380/ |
https://192.168.10.9:2379/ | false |
+-----+-----+-----+-----+-----+
+-----+
```

Take note of the ID and the name of the unhealthy etcd member, because these values are required later in the procedure. The **etcdctl endpoint health** command will list the removed member until the replacement procedure is completed and the new member is added.

- d. Remove the unhealthy etcd member by providing the ID to the **etcdctl member remove** command:

**WARNING**

Be sure to remove the correct etcd member; removing a good etcd member might lead to quorum loss.

```
sh-4.2# etcdctl member remove 7a8197040a5126c8
```

Example output

```
Member 7a8197040a5126c8 removed from cluster b23536c33f2cdd1b
```

- e. View the member list again and verify that the member was removed:

```
sh-4.2# etcdctl member list -w table
```

Example output

```
+-----+-----+-----+-----+-----+
+-----+
| ID          | STATUS | NAME                | PEER ADDRS          | CLIENT
ADDRS        | IS LEARNER |                      |                      |
+-----+-----+-----+-----+-----+
+-----+
| cc3830a72fc357f9 | started | openshift-control-plane-2 | https://192.168.10.11:2380/ |
https://192.168.10.11:2379/ | false |
| 8d5abe9669a39192 | started | openshift-control-plane-1 | https://192.168.10.10:2380/ |
https://192.168.10.10:2379/ | false |
+-----+-----+-----+-----+-----+
+-----+
```

You can now exit the node shell.

**IMPORTANT**

After you remove the member, the cluster might be unreachable for a short time while the remaining etcd instances reboot.

2. Turn off the quorum guard by entering the following command:

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides": {"useUnsupportedUnsafeNonHANonProductionUnstableEtcd": true}}}'
```

This command ensures that you can successfully re-create secrets and roll out the static pods.

3. Remove the old secrets for the unhealthy etcd member that was removed by running the following commands.
- a. List the secrets for the unhealthy etcd member that was removed.


```
$ oc get secrets -n openshift-etcd | grep openshift-control-plane-2
```

Pass in the name of the unhealthy etcd member that you took note of earlier in this procedure.

There is a peer, serving, and metrics secret as shown in the following output:

```
etcd-peer-openshift-control-plane-2      kubernetes.io/tls  2  134m
etcd-serving-metrics-openshift-control-plane-2 kubernetes.io/tls  2  134m
etcd-serving-openshift-control-plane-2    kubernetes.io/tls  2  134m
```

b. Delete the secrets for the unhealthy etcd member that was removed.

i. Delete the peer secret:

```
$ oc delete secret etcd-peer-openshift-control-plane-2 -n openshift-etcd

secret "etcd-peer-openshift-control-plane-2" deleted
```

ii. Delete the serving secret:

```
$ oc delete secret etcd-serving-metrics-openshift-control-plane-2 -n openshift-etcd

secret "etcd-serving-metrics-openshift-control-plane-2" deleted
```

iii. Delete the metrics secret:

```
$ oc delete secret etcd-serving-openshift-control-plane-2 -n openshift-etcd

secret "etcd-serving-openshift-control-plane-2" deleted
```

4. Obtain the machine for the unhealthy member.

In a terminal that has access to the cluster as a **cluster-admin** user, run the following command:

```
$ oc get machines -n openshift-machine-api -o wide
```

Example output

NAME	PHASE	TYPE	REGION	ZONE	AGE	NODE
examplecluster-control-plane-0	Running				3h11m	openshift-control-plane-0
baremetalhost:///openshift-machine-api/openshift-control-plane-0/da1ebe11-3ff2-41c5-b099-0aa41222964e	externally provisioned					
examplecluster-control-plane-1	Running				3h11m	openshift-control-plane-1
baremetalhost:///openshift-machine-api/openshift-control-plane-1/d9f9acbc-329c-475e-8d81-03b20280a3e1	externally provisioned					
examplecluster-control-plane-2	Running				3h11m	openshift-control-plane-2
baremetalhost:///openshift-machine-api/openshift-control-plane-2/3354bdac-61d8-410f-be5b-6a395b056135	externally provisioned					
examplecluster-compute-0	Running				165m	openshift-compute-0
baremetalhost:///openshift-machine-api/openshift-compute-0/3d685b81-7410-4bb3-80ec-13a31858241f	provisioned					

```
examplecluster-compute-1      Running      165m  openshift-compute-1
baremetalhost:///openshift-machine-api/openshift-compute-1/0fdae6eb-2066-4241-91dc-
e7ea72ab13b9      provisioned
```

- 1 This is the control plane machine for the unhealthy node, **examplecluster-control-plane-2**.

5. Ensure that the Bare Metal Operator is available by running the following command:

```
$ oc get clusteroperator baremetal
```

Example output

```
NAME      VERSION  AVAILABLE  PROGRESSING  DEGRADED  SINCE  MESSAGE
baremetal  4.20.0   True       False        False     3d15h
```

6. Remove the old **BareMetalHost** object by running the following command:

```
$ oc delete bmh openshift-control-plane-2 -n openshift-machine-api
```

Example output

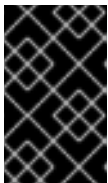
```
baremetalhost.metal3.io "openshift-control-plane-2" deleted
```

7. Delete the machine of the unhealthy member by running the following command:

```
$ oc delete machine -n openshift-machine-api examplecluster-control-plane-2
```

After you remove the **BareMetalHost** and **Machine** objects, then the **Machine** controller automatically deletes the **Node** object.

If deletion of the machine is delayed for any reason or the command is obstructed and delayed, you can force deletion by removing the machine object finalizer field.



IMPORTANT

Do not interrupt machine deletion by pressing **Ctrl+c**. You must allow the command to proceed to completion. Open a new terminal window to edit and delete the finalizer fields.

A new machine is automatically provisioned after deleting the machine of the unhealthy member.

- a. Edit the machine configuration by running the following command:

```
$ oc edit machine -n openshift-machine-api examplecluster-control-plane-2
```

- b. Delete the following fields in the **Machine** custom resource, and then save the updated file:

```
finalizers:
- machine.machine.openshift.io
```

Example output

```
machine.machine.openshift.io/examplecluster-control-plane-2 edited
```

8. Verify that the machine was deleted by running the following command:

```
$ oc get machines -n openshift-machine-api -o wide
```

Example output

NAME	PHASE	TYPE	REGION	ZONE	AGE	NODE
examplecluster-control-plane-0	Running				3h11m	openshift-control-plane-0
baremetalhost:///openshift-machine-api/openshift-control-plane-0/da1ebe11-3ff2-41c5-b099-0aa41222964e	externally provisioned					
examplecluster-control-plane-1	Running				3h11m	openshift-control-plane-1
baremetalhost:///openshift-machine-api/openshift-control-plane-1/d9f9acbc-329c-475e-8d81-03b20280a3e1	externally provisioned					
examplecluster-compute-0	Running				165m	openshift-compute-0
baremetalhost:///openshift-machine-api/openshift-compute-0/3d685b81-7410-4bb3-80ec-13a31858241f	provisioned					
examplecluster-compute-1	Running				165m	openshift-compute-1
baremetalhost:///openshift-machine-api/openshift-compute-1/0fdae6eb-2066-4241-91dc-e7ea72ab13b9	provisioned					

9. Verify that the node has been deleted by running the following command:

```
$ oc get nodes
```

NAME	STATUS	ROLES	AGE	VERSION
openshift-control-plane-0	Ready	master	3h24m	v1.33.4
openshift-control-plane-1	Ready	master	3h24m	v1.33.4
openshift-compute-0	Ready	worker	176m	v1.33.4
openshift-compute-1	Ready	worker	176m	v1.33.4

10. Create the new **BareMetalHost** object and the secret to store the BMC credentials:

```
$ cat <<EOF | oc apply -f -
apiVersion: v1
kind: Secret
metadata:
  name: openshift-control-plane-2-bmc-secret
  namespace: openshift-machine-api
data:
  password: <password>
  username: <username>
type: Opaque
---
apiVersion: metal3.io/v1alpha1
kind: BareMetalHost
metadata:
  name: openshift-control-plane-2
  namespace: openshift-machine-api
spec:
```

```

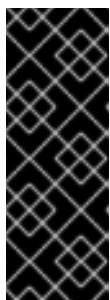
automatedCleaningMode: disabled
bmc:
  address: redfish://10.46.61.18:443/redfish/v1/Systems/1
  credentialsName: openshift-control-plane-2-bmc-secret
  disableCertificateVerification: true
  bootMACAddress: 48:df:37:b0:8a:a0
  bootMode: UEFI
  externallyProvisioned: false
  online: true
  rootDeviceHints:
    deviceName: /dev/disk/by-id/scsi-<serial_number>
  userData:
    name: master-user-data-managed
    namespace: openshift-machine-api
EOF

```



NOTE

The username and password can be found from the other bare metal host's secrets. The protocol to use in **bmc:address** can be taken from other bmh objects.



IMPORTANT

If you reuse the **BareMetalHost** object definition from an existing control plane host, do not leave the **externallyProvisioned** field set to **true**.

Existing control plane **BareMetalHost** objects may have the **externallyProvisioned** flag set to **true** if they were provisioned by the OpenShift Container Platform installation program.

After the inspection is complete, the **BareMetalHost** object is created and available to be provisioned.

11. Verify the creation process using available **BareMetalHost** objects:

```
$ oc get bmh -n openshift-machine-api
```

NAME	STATE	CONSUMER	ONLINE ERROR	AGE
openshift-control-plane-0	externally provisioned	examplecluster-control-plane-0	true	4h48m
openshift-control-plane-1	externally provisioned	examplecluster-control-plane-1	true	4h48m
openshift-control-plane-2	available	examplecluster-control-plane-3	true	47m
openshift-compute-0	provisioned	examplecluster-compute-0	true	4h48m
openshift-compute-1	provisioned	examplecluster-compute-1	true	4h48m

- a. Verify that a new machine has been created:

```
$ oc get machines -n openshift-machine-api -o wide
```

Example output

NAME	PHASE	TYPE	REGION	ZONE	AGE	NODE STATE
examplecluster-control-plane-0	Running				3h11m	openshift-control-plane-0
baremetalhost:///openshift-machine-api/openshift-control-plane-0/da1ebe11-3ff2-41c5-b099-0aa41222964e externally provisioned 1						
examplecluster-control-plane-1	Running				3h11m	openshift-control-plane-1
baremetalhost:///openshift-machine-api/openshift-control-plane-1/d9f9acbc-329c-475e-8d81-03b20280a3e1 externally provisioned						
examplecluster-control-plane-2	Running				3h11m	openshift-control-plane-2
baremetalhost:///openshift-machine-api/openshift-control-plane-2/3354bdac-61d8-410f-be5b-6a395b056135 externally provisioned						
examplecluster-compute-0	Running				165m	openshift-compute-0
baremetalhost:///openshift-machine-api/openshift-compute-0/3d685b81-7410-4bb3-80ec-13a31858241f provisioned						
examplecluster-compute-1	Running				165m	openshift-compute-1
baremetalhost:///openshift-machine-api/openshift-compute-1/0fdae6eb-2066-4241-91dc-e7ea72ab13b9 provisioned						

- 1** The new machine, **clustername-8qw5l-master-3** is being created and is ready after the phase changes from **Provisioning** to **Running**.

It should take a few minutes for the new machine to be created. The etcd cluster Operator will automatically sync when the machine or node returns to a healthy state.

- b. Verify that the bare metal host becomes provisioned and no error reported by running the following command:

```
$ oc get bmh -n openshift-machine-api
```

Example output

```
$ oc get bmh -n openshift-machine-api
```

NAME	STATE	CONSUMER	ONLINE	ERROR	AGE
openshift-control-plane-0	externally provisioned	examplecluster-control-plane-0	true		4h48m
openshift-control-plane-1	externally provisioned	examplecluster-control-plane-1	true		4h48m
openshift-control-plane-2	provisioned	examplecluster-control-plane-3	true		47m
openshift-compute-0	provisioned	examplecluster-compute-0	true		4h48m
openshift-compute-1	provisioned	examplecluster-compute-1	true		4h48m

- c. Verify that the new node is added and in a ready state by running this command:

```
$ oc get nodes
```

Example output

```
$ oc get nodes
```

NAME	STATUS	ROLES	AGE	VERSION
openshift-control-plane-0	Ready	master	4h26m	v1.33.4

```

openshift-control-plane-1 Ready master 4h26m v1.33.4
openshift-control-plane-2 Ready master 12m v1.33.4
openshift-compute-0 Ready worker 3h58m v1.33.4
openshift-compute-1 Ready worker 3h58m v1.33.4

```

12. Turn the quorum guard back on by entering the following command:

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides": null}}'
```

13. You can verify that the **unsupportedConfigOverrides** section is removed from the object by entering this command:

```
$ oc get etcd/cluster -oyaml
```

14. If you are using single-node OpenShift, restart the node. Otherwise, you might encounter the following error in the etcd cluster Operator:

Example output

```

EtcdCertSignerControllerDegraded: [Operation cannot be fulfilled on secrets "etcd-peer-sno-0": the object has been modified; please apply your changes to the latest version and try again, Operation cannot be fulfilled on secrets "etcd-serving-sno-0": the object has been modified; please apply your changes to the latest version and try again, Operation cannot be fulfilled on secrets "etcd-serving-metrics-sno-0": the object has been modified; please apply your changes to the latest version and try again]

```

Verification

1. Verify that all etcd pods are running properly.

In a terminal that has access to the cluster as a **cluster-admin** user, run the following command:

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd
```

Example output

```

etcd-openshift-control-plane-0 5/5 Running 0 105m
etcd-openshift-control-plane-1 5/5 Running 0 107m
etcd-openshift-control-plane-2 5/5 Running 0 103m

```

If the output from the previous command only lists two pods, you can manually force an etcd redeployment. In a terminal that has access to the cluster as a **cluster-admin** user, run the following command:

```
$ oc patch etcd cluster -p='{"spec": {"forceRedeploymentReason": "recovery-"$( date --rfc-3339=ns )"'}}' --type=merge 1
```

- 1** The **forceRedeploymentReason** value must be unique, which is why a timestamp is appended.

To verify there are exactly three etcd members, connect to the running etcd container, passing in the name of a pod that was not on the affected node. In a terminal that has access to the cluster as a **cluster-admin** user, run the following command:

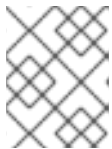
```
$ oc rsh -n openshift-etcd etcd-openshift-control-plane-0
```

2. View the member list:

```
sh-4.2# etcdctl member list -w table
```

Example output

```
+-----+-----+-----+-----+-----+
| ID | STATUS | NAME | PEER ADDRS | CLIENT ADDRS |
| IS LEARNER |
+-----+-----+-----+-----+-----+
| 7a8197040a5126c8 | started | openshift-control-plane-2 | https://192.168.10.11:2380 | https://192.168.10.11:2379 | false |
| 8d5abe9669a39192 | started | openshift-control-plane-1 | https://192.168.10.10:2380 | https://192.168.10.10:2379 | false |
| cc3830a72fc357f9 | started | openshift-control-plane-0 | https://192.168.10.9:2380 | https://192.168.10.9:2379 | false |
+-----+-----+-----+-----+-----+
```



NOTE

If the output from the previous command lists more than three etcd members, you must carefully remove the unwanted member.

3. Verify that all etcd members are healthy by running the following command:

```
# etcdctl endpoint health --cluster
```

Example output

```
https://192.168.10.10:2379 is healthy: successfully committed proposal: took = 8.973065ms
https://192.168.10.9:2379 is healthy: successfully committed proposal: took = 11.559829ms
https://192.168.10.11:2379 is healthy: successfully committed proposal: took = 11.665203ms
```

4. Validate that all nodes are at the latest revision by running the following command:

```
$ oc get etcd -o=jsonpath='{range.items[0].status.conditions[?(@.type=="NodeInstallerProgressing")]}{.reason}{"\n"}{.message}{"\n"}'
```

```
AllNodesAtLatestRevision
```

6.2.5. Additional resources

- [Quorum protection with machine lifecycle hooks](#)

6.3. DISASTER RECOVERY

6.3.1. About disaster recovery

The disaster recovery documentation provides information for administrators on how to recover from several disaster situations that might occur with their OpenShift Container Platform cluster. As an administrator, you might need to follow one or more of the following procedures to return your cluster to a working state.



IMPORTANT

Disaster recovery requires you to have at least one healthy control plane host.

Quorum restoration

This solution handles situations where you have lost the majority of your control plane hosts, leading to etcd quorum loss and the cluster going offline. This solution does not require an etcd backup.



NOTE

If you have a majority of your control plane nodes still available and have an etcd quorum, then [replace a single unhealthy etcd member](#).

Restoring to a previous cluster state

This solution handles situations where you want to restore your cluster to a previous state, for example, if an administrator deletes something critical. If you have taken an etcd backup, you can restore your cluster to a previous state.

If applicable, you might also need to [recover from expired control plane certificates](#).



WARNING

Restoring to a previous cluster state is a destructive and destabilizing action to take on a running cluster. This procedure should only be used as a last resort.

Prior to performing a restore, see [About restoring cluster state](#) for more information on the impact to the cluster.

Recovering from expired control plane certificates

This solution handles situations where your control plane certificates have expired. For example, if you shut down your cluster before the first certificate rotation, which occurs 24 hours after installation, your certificates will not be rotated and will expire. You can follow this procedure to recover from expired control plane certificates.

6.3.1.1. Testing restore procedures

Testing the restore procedure is important to ensure that your automation and workload handle the new cluster state gracefully. Due to the complex nature of etcd quorum and the etcd Operator attempting to mend automatically, it is often difficult to correctly bring your cluster into a broken enough state that it can be restored.



WARNING

You **must** have SSH access to the cluster. Your cluster might be entirely lost without SSH access.

Prerequisites

- You have SSH access to control plane hosts.
- You have installed the OpenShift CLI (**oc**).

Procedure

1. Use SSH to connect to each of your nonrecovery nodes and run the following commands to disable etcd and the **kubelet** service:

- a. Disable etcd by running the following command:

```
$ sudo /usr/local/bin/disable-etcd.sh
```

- b. Delete variable data for etcd by running the following command:

```
$ sudo rm -rf /var/lib/etcd
```

- c. Disable the **kubelet** service by running the following command:

```
$ sudo systemctl disable kubelet.service
```

2. Exit every SSH session.
3. Run the following command to ensure that your nonrecovery nodes are in a **NOT READY** state:

```
$ oc get nodes
```

4. Follow the steps in "Restoring to a previous cluster state" to restore your cluster.
5. After you restore the cluster and the API responds, use SSH to connect to each nonrecovery node and enable the **kubelet** service:

```
$ sudo systemctl enable kubelet.service
```

6. Exit every SSH session.
7. Run the following command to observe your nodes coming back into the **READY** state:

```
$ oc get nodes
```

8. Run the following command to verify that etcd is available:

```
$ oc get pods -n openshift-etcd
```

Additional resources

- [Restoring to a previous cluster state](#)

6.3.2. Quorum restoration

You can use the **quorum-restore.sh** script to restore etcd quorum on clusters that are offline due to quorum loss. When quorum is lost, the OpenShift Container Platform API becomes read-only. After quorum is restored, the OpenShift Container Platform API returns to read/write mode.

6.3.2.1. Restoring etcd quorum for high availability clusters

You can use the **quorum-restore.sh** script to restore etcd quorum on clusters that are offline due to quorum loss. When quorum is lost, the OpenShift Container Platform API becomes read-only. After quorum is restored, the OpenShift Container Platform API returns to read/write mode.

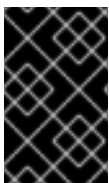
The **quorum-restore.sh** script instantly brings back a new single-member etcd cluster based on its local data directory and marks all other members as invalid by retiring the previous cluster identifier. No prior backup is required to restore the control plane from.

For high availability (HA) clusters, a three-node HA cluster requires you to shut down etcd on two hosts to avoid a cluster split. On four-node and five-node HA clusters, you must shut down three hosts. Quorum requires a simple majority of nodes. The minimum number of nodes required for quorum on a three-node HA cluster is two. On four-node and five-node HA clusters, the minimum number of nodes required for quorum is three. If you start a new cluster from backup on your recovery host, the other etcd members might still be able to form quorum and continue service.



WARNING

You might experience data loss if the host that runs the restoration does not have all data replicated to it.



IMPORTANT

Quorum restoration should not be used to decrease the number of nodes outside of the restoration process. Decreasing the number of nodes results in an unsupported cluster configuration.

Prerequisites

- You have SSH access to the node used to restore quorum.

Procedure

1. Select a control plane host to use as the recovery host. You run the restore operation on this host.

- a. List the running etcd pods by running the following command:

```
$ oc get pods -n openshift-etcd -l app=etcd --field-selector="status.phase==Running"
```

- b. Choose a pod and run the following command to obtain its IP address:

```
$ oc exec -n openshift-etcd <etcd-pod> -c etcdctl -- etcdctl endpoint status -w table
```

Note the IP address of a member that is not a learner and has the highest Raft index.

- c. Run the following command and note the node name that corresponds to the IP address of the chosen etcd member:

```
$ oc get nodes -o jsonpath='{range .items[*]}[{.metadata.name},{.status.addresses[?(@.type=="InternalIP")].address}]{end}'
```

2. Using SSH, connect to the chosen recovery node and run the following command to restore etcd quorum:

```
$ sudo -E /usr/local/bin/quorum-restore.sh
```

After a few minutes, the nodes that went down are automatically synchronized with the node that the recovery script was run on. Any remaining online nodes automatically rejoin the new etcd cluster created by the **quorum-restore.sh** script. This process takes a few minutes.

3. Exit the SSH session.
4. Return to a three-node configuration if any nodes are offline. Repeat the following steps for each node that is offline to delete and re-create them. After the machines are re-created, a new revision is forced and etcd automatically scales up.
 - If you use a user-provisioned bare-metal installation, you can re-create a control plane machine by using the same method that you used to originally create it. For more information, see "Installing a user-provisioned cluster on bare metal".



WARNING

Do not delete and re-create the machine for the recovery host.

- If you are running installer-provisioned infrastructure, or you used the Machine API to create your machines, follow these steps:

**WARNING**

Do not delete and re-create the machine for the recovery host.

For bare-metal installations on installer-provisioned infrastructure, control plane machines are not re-created. For more information, see "Replacing a bare-metal control plane node".

- a. Obtain the machine for one of the offline nodes.

In a terminal that has access to the cluster as a **cluster-admin** user, run the following command:

```
$ oc get machines -n openshift-machine-api -o wide
```

Example output

NAME NODE	PHASE PROVIDERID	TYPE	REGION STATE	ZONE	AGE
clustername-8qw5l-master-0 3h37m ip-10-0-131-183.ec2.internal	Running	m4.xlarge	us-east-1	us-east-1a	us-east-1a aws:///us-east-1a/i-0ec2782f8287dfb7e
stopped 1					
clustername-8qw5l-master-1 3h37m ip-10-0-143-125.ec2.internal	Running	m4.xlarge	us-east-1	us-east-1b	us-east-1b aws:///us-east-1b/i-096c349b700a19631
running					
clustername-8qw5l-master-2 3h37m ip-10-0-154-194.ec2.internal	Running	m4.xlarge	us-east-1	us-east-1c	us-east-1c aws:///us-east-1c/i-02626f1dba9ed5bba
running					
clustername-8qw5l-worker-us-east-1a-wbtgd 3h28m ip-10-0-129-226.ec2.internal	Running	m4.large	us-east-1	us-east-1a	us-east-1a aws:///us-east-1a/i-010ef6279b4662ced
running					
clustername-8qw5l-worker-us-east-1b-lrdxb 3h28m ip-10-0-144-248.ec2.internal	Running	m4.large	us-east-1	us-east-1b	us-east-1b aws:///us-east-1b/i-0cb45ac45a166173b
running					
clustername-8qw5l-worker-us-east-1c-pkg26 3h28m ip-10-0-170-181.ec2.internal	Running	m4.large	us-east-1	us-east-1c	us-east-1c aws:///us-east-1c/i-06861c00007751b0a
running					

- 1** This is the control plane machine for the offline node, **ip-10-0-131-183.ec2.internal**.

- b. Delete the machine of the offline node by running:

```
$ oc delete machine -n openshift-machine-api clustername-8qw5l-master-0 1
```

- 1** Specify the name of the control plane machine for the offline node.

A new machine is automatically provisioned after deleting the machine of the offline node.

- Verify that a new machine has been created by running:

```
$ oc get machines -n openshift-machine-api -o wide
```

Example output

NAME NODE	PHASE PROVIDERID	TYPE	REGION STATE	ZONE	AGE
clustername-8qw5l-master-1 3h37m ip-10-0-143-125.ec2.internal	Running aws:///us-east-1b/i-096c349b700a19631	Running	m4.xlarge	us-east-1	us-east-1b
clustername-8qw5l-master-2 3h37m ip-10-0-154-194.ec2.internal	Running aws:///us-east-1c/i-02626f1dba9ed5bba	Running	m4.xlarge	us-east-1	us-east-1c
clustername-8qw5l-master-3 ip-10-0-173-171.ec2.internal	Provisioning aws:///us-east-1a/i-015b0888fe17bc2c8	Provisioning	m4.xlarge	us-east-1	us-east-1a 85s
clustername-8qw5l-worker-us-east-1a-wbtgd 3h28m ip-10-0-129-226.ec2.internal	Running aws:///us-east-1a/i-010ef6279b4662ced	Running	m4.large	us-east-1	us-east-1a
clustername-8qw5l-worker-us-east-1b-lrdxb 3h28m ip-10-0-144-248.ec2.internal	Running aws:///us-east-1b/i-0cb45ac45a166173b	Running	m4.large	us-east-1	us-east-1b
clustername-8qw5l-worker-us-east-1c-pkg26 3h28m ip-10-0-170-181.ec2.internal	Running aws:///us-east-1c/i-06861c00007751b0a	Running	m4.large	us-east-1	us-east-1c

- The new machine, **clustername-8qw5l-master-3** is being created and is ready after the phase changes from **Provisioning** to **Running**.

It might take a few minutes for the new machine to be created. The etcd cluster Operator will automatically synchronize when the machine or node returns to a healthy state.

- Repeat these steps for each node that is offline.

- Wait until the control plane recovers by running the following command:

```
$ oc adm wait-for-stable-cluster
```



NOTE

It can take up to 15 minutes for the control plane to recover.

Troubleshooting

- If you see no progress rolling out the etcd static pods, you can force redeployment from the etcd cluster Operator by running the following command:

```
$ oc patch etcd cluster -p='{ "spec": { "forceRedeploymentReason": "recovery-"$(date --rfc-3339=ns )"' --type=merge
```

6.3.2.2. Additional resources

- [Installing a user-provisioned cluster on bare metal](#)
- [Replacing a bare-metal control plane node](#)

6.3.3. Restoring to a previous cluster state

To restore the cluster to a previous state, you must have previously backed up the **etcd** data by creating a snapshot. You will use this snapshot to restore the cluster state. For more information, see "Backing up etcd data".

6.3.3.1. About restoring to a previous cluster state

To restore the cluster to a previous state, you must have previously backed up the **etcd** data by creating a snapshot. You will use this snapshot to restore the cluster state. For more information, see "Backing up etcd data".

You can use an etcd backup to restore your cluster to a previous state. This can be used to recover from the following situations:

- The cluster has lost the majority of control plane hosts (quorum loss).
- An administrator has deleted something critical and must restore to recover the cluster.



WARNING

Restoring to a previous cluster state is a destructive and destabilizing action to take on a running cluster. This should only be used as a last resort.

If you are able to retrieve data using the Kubernetes API server, then etcd is available and you should not restore using an etcd backup.

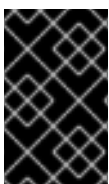
Restoring etcd effectively takes a cluster back in time and all clients will experience a conflicting, parallel history. This can impact the behavior of watching components like kubelets, Kubernetes controller managers, persistent volume controllers, and OpenShift Container Platform Operators, including the network Operator.

It can cause Operator churn when the content in etcd does not match the actual content on disk, causing Operators for the Kubernetes API server, Kubernetes controller manager, Kubernetes scheduler, and etcd to get stuck when files on disk conflict with content in etcd. This can require manual actions to resolve the issues.

In extreme cases, the cluster can lose track of persistent volumes, delete critical workloads that no longer exist, reimagine machines, and rewrite CA bundles with expired certificates.

6.3.3.2. Restoring to a previous cluster state for a single node

You can use a saved etcd backup to restore a previous cluster state on a single node.



IMPORTANT

When you restore your cluster, you must use an etcd backup that was taken from the same z-stream release. For example, an OpenShift Container Platform 4.20.2 cluster must use an etcd backup that was taken from 4.20.2.

Prerequisites

- Access to the cluster as a user with the **cluster-admin** role through a certificate-based **kubeconfig** file, like the one that was used during installation.
- You have SSH access to control plane hosts.
- A backup directory containing both the etcd snapshot and the resources for the static pods, which were from the same backup. The file names in the directory must be in the following formats: **snapshot_<timestamp>.db** and **static_kubernetes_<timestamp>.tar.gz**.

Procedure

1. Use SSH to connect to the single node and copy the etcd backup to the **/home/core** directory by running the following command:

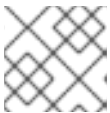
```
$ cp <etcd_backup_directory> /home/core
```

2. Run the following command in the single node to restore the cluster from a previous backup:

```
$ sudo -E /usr/local/bin/cluster-restore.sh /home/core/<etcd_backup_directory>
```

3. Exit the SSH session.
4. Monitor the recovery progress of the control plane by running the following command:

```
$ oc adm wait-for-stable-cluster
```



NOTE

It can take up to 15 minutes for the control plane to recover.

6.3.3.3. Restoring to a previous cluster state for more than one node

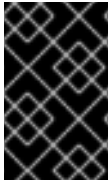
You can use a saved etcd backup to restore a previous cluster state or restore a cluster that has lost the majority of control plane hosts.

For high availability (HA) clusters, a three-node HA cluster requires you to shut down etcd on two hosts to avoid a cluster split. On four-node and five-node HA clusters, you must shut down three hosts. Quorum requires a simple majority of nodes. The minimum number of nodes required for quorum on a three-node HA cluster is two. On four-node and five-node HA clusters, the minimum number of nodes required for quorum is three. If you start a new cluster from backup on your recovery host, the other etcd members might still be able to form quorum and continue service.



NOTE

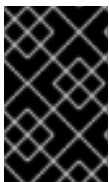
If your cluster uses a control plane machine set, see "Recovering a degraded etcd Operator" in "Troubleshooting the control plane machine set" for an etcd recovery procedure. For OpenShift Container Platform on a single node, see "Restoring to a previous cluster state for a single node".

**IMPORTANT**

When you restore your cluster, you must use an etcd backup that was taken from the same z-stream release. For example, an OpenShift Container Platform 4.20.2 cluster must use an etcd backup that was taken from 4.20.2.

Prerequisites

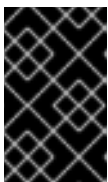
- Access to the cluster as a user with the **cluster-admin** role through a certificate-based **kubeconfig** file, like the one that was used during installation.
- A healthy control plane host to use as the recovery host.
- You have SSH access to control plane hosts.
- A backup directory containing both the **etcd** snapshot and the resources for the static pods, which were from the same backup. The file names in the directory must be in the following formats: **snapshot_<timestamp>.db** and **static_kubernetes_<timestamp>.tar.gz**.
- Nodes must be accessible or bootable.

**IMPORTANT**

For non-recovery control plane nodes, it is not required to establish SSH connectivity or to stop the static pods. You can delete and re-create other non-recovery, control plane machines, one by one.

Procedure

1. Select a control plane host to use as the recovery host. This is the host that you run the restore operation on.
2. Establish SSH connectivity to each of the control plane nodes, including the recovery host. **kube-apiserver** becomes inaccessible after the restore process starts, so you cannot access the control plane nodes. For this reason, it is recommended to establish SSH connectivity to each control plane host in a separate terminal.

**IMPORTANT**

If you do not complete this step, you will not be able to access the control plane hosts to complete the restore procedure, and you will be unable to recover your cluster from this state.

3. Using SSH, connect to each control plane node and run the following command to disable etcd:

```
$ sudo -E /usr/local/bin/disable-etcd.sh
```

4. Copy the etcd backup directory to the recovery control plane host.
This procedure assumes that you copied the **backup** directory containing the etcd snapshot and the resources for the static pods to the **/home/core/** directory of your recovery control plane host.

5. Use SSH to connect to the recovery host and restore the cluster from a previous backup by running the following command:

```
$ sudo -E /usr/local/bin/cluster-restore.sh /home/core/<etcd-backup-directory>
```

6. Exit the SSH session.
7. Once the API responds, turn off the etcd Operator quorum guard by running the following command:

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides": {"useUnsupportedUnsafeNonHANonProductionUnstableEtcd": true}}}'
```

8. Monitor the recovery progress of the control plane by running the following command:

```
$ oc adm wait-for-stable-cluster
```



NOTE

It can take up to 15 minutes for the control plane to recover.

9. Once recovered, enable the quorum guard by running the following command:

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides": null}}'
```

Troubleshooting

If you see no progress rolling out the etcd static pods, you can force redeployment from the **cluster-etcd-operator** by running the following command:

```
$ oc patch etcd cluster -p='{"spec": {"forceRedeploymentReason": "recovery-"$(date --rfc-3339=ns)""}}' --type=merge
```

Additional resources

- [Recovering a degraded etcd Operator](#)

6.3.3.4. Restoring a cluster manually from an etcd backup

The restore procedure described in the section "Restoring to a previous cluster state":

- Requires the complete recreation of 2 control plane nodes, which might be a complex procedure for clusters installed with the UPI installation method, since an UPI installation does not create any **Machine** or **ControlPlaneMachineset** for the control plane nodes.
- Uses the script `/usr/local/bin/cluster-restore.sh`, which starts a new single-member etcd cluster and then scales it to three members.

In contrast, this procedure:

- Does not require recreating any control plane nodes.
- Directly starts a three-member etcd cluster.

If the cluster uses a **MachineSet** for the control plane, it is suggested to use the "Restoring to a previous cluster state" for a simpler etcd recovery procedure.

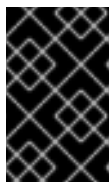
When you restore your cluster, you must use an etcd backup that was taken from the same z-stream release. For example, an OpenShift Container Platform 4.7.2 cluster must use an etcd backup that was taken from 4.7.2.

Prerequisites

- Access to the cluster as a user with the **cluster-admin** role; for example, the **kubeadmin** user.
- SSH access to *all* control plane hosts, with a host user allowed to become **root**; for example, the default **core** host user.
- A backup directory containing both a previous etcd snapshot and the resources for the static pods from the same backup. The file names in the directory must be in the following formats: **snapshot_<timestamp>.db** and **static_kubernetes_<timestamp>.tar.gz**.

Procedure

1. Use SSH to connect to each of the control plane nodes.
The Kubernetes API server becomes inaccessible after the restore process starts, so you cannot access the control plane nodes. For this reason, it is recommended to use a SSH connection for each control plane host you are accessing in a separate terminal.



IMPORTANT

If you do not complete this step, you will not be able to access the control plane hosts to complete the restore procedure, and you will be unable to recover your cluster from this state.

2. Copy the etcd backup directory to each control plane host.
This procedure assumes that you copied the **backup** directory containing the etcd snapshot and the resources for the static pods to the **/home/core/assets** directory of each control plane host. You might need to create such **assets** folder if it does not exist yet.
3. Stop the static pods on all the control plane nodes; one host at a time.
 - a. Move the existing Kubernetes API Server static pod manifest out of the kubelet manifest directory.

```
$ mkdir -p /root/manifests-backup
```

```
$ mv /etc/kubernetes/manifests/kube-apiserver-pod.yaml /root/manifests-backup/
```

- b. Verify that the Kubernetes API Server containers have stopped with the command:

```
$ crictl ps | grep kube-apiserver | grep -E -v "operator|guard"
```

The output of this command should be empty. If it is not empty, wait a few minutes and check again.

- c. If the Kubernetes API Server containers are still running, terminate them manually with the following command:

```
$ crictl stop <container_id>
```

- d. Repeat the same steps for **kube-controller-manager-pod.yaml**, **kube-scheduler-pod.yaml** and finally **etcd-pod.yaml**.

- i. Stop the **kube-controller-manager** pod with the following command:

```
$ mv /etc/kubernetes/manifests/kube-controller-manager-pod.yaml /root/manifests-backup/
```

- ii. Check if the containers are stopped using the following command:

```
$ crictl ps | grep kube-controller-manager | grep -E -v "operator|guard"
```

- iii. Stop the **kube-scheduler** pod using the following command:

```
$ mv /etc/kubernetes/manifests/kube-scheduler-pod.yaml /root/manifests-backup/
```

- iv. Check if the containers are stopped using the following command:

```
$ crictl ps | grep kube-scheduler | grep -E -v "operator|guard"
```

- v. Stop the **etcd** pod using the following command:

```
$ mv /etc/kubernetes/manifests/etcd-pod.yaml /root/manifests-backup/
```

- vi. Check if the containers are stopped using the following command:

```
$ crictl ps | grep etcd | grep -E -v "operator|guard"
```

4. On each control plane host, save the current **etcd** data, by moving it into the **backup** folder:

```
$ mkdir /home/core/assets/old-member-data
```

```
$ mv /var/lib/etcd/member /home/core/assets/old-member-data
```

This data will be useful in case the **etcd** backup restore does not work and the **etcd** cluster must be restored to the current state.

5. Find the correct etcd parameters for each control plane host.

- a. The value for **<ETCD_NAME>** is unique for the each control plane host, and it is equal to the value of the **ETCD_NAME** variable in the manifest **/etc/kubernetes/static-pod-resources/etcd-certs/configmaps/restore-etcd-pod/pod.yaml** file in the specific control plane host. It can be found with the command:

```
RESTORE_ETCD_POD_YAML="/etc/kubernetes/static-pod-resources/etcd-certs/configmaps/restore-etcd-pod/pod.yaml"
cat $RESTORE_ETCD_POD_YAML | \
  grep -A 1 $(cat $RESTORE_ETCD_POD_YAML | grep 'export ETCD_NAME' | grep -Eo 'NODE_.+_ETCD_NAME') | \
  grep -Po '(?<=value: ").+(?=")'
```

- b. The value for **<UUID>** can be generated in a control plane host with the command:

```
$ uuidgen
```



NOTE

The value for **<UUID>** must be generated only once. After generating **UUID** on one control plane host, do not generate it again on the others. The same **UUID** will be used in the next steps on all control plane hosts.

- c. The value for **ETCD_NODE_PEER_URL** should be set like the following example:

```
https://<IP_CURRENT_HOST>:2380
```

The correct IP can be found from the **<ETCD_NAME>** of the specific control plane host, with the command:

```
$ echo <ETCD_NAME> | \
  sed -E 's/[.]/_/g' | \
  xargs -l {} grep {} /etc/kubernetes/static-pod-resources/etcd-certs/configmaps/etcd-
scripts/etcd.env | \
  grep "IP" | grep -Po '(?<=)".+(?<=)'
```

- d. The value for **<ETCD_INITIAL_CLUSTER>** should be set like the following, where **<ETCD_NAME_n>** is the **<ETCD_NAME>** of each control plane host.



NOTE

The port used must be 2380 and not 2379. The port 2379 is used for etcd database management and is configured directly in etcd start command in container.

Example output

```
<ETCD_NAME_0>=<ETCD_NODE_PEER_URL_0>,<ETCD_NAME_1>=
<ETCD_NODE_PEER_URL_1>,<ETCD_NAME_2>=<ETCD_NODE_PEER_URL_2>
```

1

- 1 Specifies the **ETCD_NODE_PEER_URL** values from each control plane host.

The **<ETCD_INITIAL_CLUSTER>** value remains same across all control plane hosts. The same value is required in the next steps on every control plane host.

6. Regenerate the etcd database from the backup.
Such operation must be executed on each control plane host.

- a. Copy the **etcd** backup to **/var/lib/etcd** directory with the command:

```
$ cp /home/core/assets/backup/<snapshot_yyyy-mm-dd_hhmmss>.db /var/lib/etcd
```

- b. Identify the correct **etcdctl** image before proceeding. Use the following command to retrieve the image from the backup of the pod manifest:

```
$ jq -r '.spec.containers[]|select(.name=="etcdctl")|.image' /root/manifests-backup/etcd-pod.yaml
```

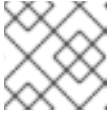
```
$ podman run --rm -it --entrypoint="/bin/bash" -v /var/lib/etcd:/var/lib/etcd:z <image-hash>
```

- c. Check that the version of the **etcdctl** tool is the version of the **etcd** server where the backup was created:

```
$ etcdctl version
```

- d. Run the following command to regenerate the **etcd** database, using the correct values for the current host:

```
$ ETCDCTL_API=3 /usr/bin/etcdctl snapshot restore /var/lib/etcd/<snapshot_yyyy-mm-dd_hhmmss>.db \
--name "<ETCD_NAME>" \
--initial-cluster="<ETCD_INITIAL_CLUSTER>" \
--initial-cluster-token "openshift-etcd-<UUID>" \
--initial-advertise-peer-urls "<ETCD_NODE_PEER_URL>" \
--data-dir="/var/lib/etcd/restore-<UUID>" \
--skip-hash-check=true
```



NOTE

The quotes are mandatory when regenerating the **etcd** database.

7. Record the values printed in the **added member** logs; for example:

Example output

```
2022-06-28T19:52:43Z info membership/cluster.go:421 added member {"cluster-id":
"c5996b7c11c30d6b", "local-member-id": "0", "added-peer-id": "56cd73b614699e7", "added-
peer-peer-urls": ["https://10.0.91.5:2380"], "added-peer-is-learner": false}
2022-06-28T19:52:43Z info membership/cluster.go:421 added member {"cluster-id":
"c5996b7c11c30d6b", "local-member-id": "0", "added-peer-id": "1f63d01b31bb9a9e", "added-
peer-peer-urls": ["https://10.0.90.221:2380"], "added-peer-is-learner": false}
2022-06-28T19:52:43Z info membership/cluster.go:421 added member {"cluster-id":
"c5996b7c11c30d6b", "local-member-id": "0", "added-peer-id": "fdc2725b3b70127c", "added-
peer-peer-urls": ["https://10.0.94.214:2380"], "added-peer-is-learner": false}
```

- a. Exit from the container.
 - b. Repeat these steps on the other control plane hosts, checking that the values printed in the **added member** logs are the same for all control plane hosts.
8. Move the regenerated **etcd** database to the default location.
Such operation must be executed on each control plane host.

- a. Move the regenerated database (the **member** folder created by the previous **etcdctl snapshot restore** command) to the default etcd location **/var/lib/etcd**:

```
$ mv /var/lib/etcd/restore-<UUID>/member /var/lib/etcd
```

- b. Restore the SELinux context for **/var/lib/etcd/member** folder on **/var/lib/etcd** directory:

```
$ restorecon -vR /var/lib/etcd/
```

- c. Remove the leftover files and directories:

```
$ rm -rf /var/lib/etcd/restore-<UUID>
```

```
$ rm /var/lib/etcd/<snapshot_yyyy-mm-dd_hhmmss>.db
```



IMPORTANT

When you are finished the **/var/lib/etcd** directory must contain only the folder **member**.

- d. Repeat these steps on the other control plane hosts.
9. Restart the etcd cluster.
 - a. The following steps must be executed on all control plane hosts, but **one host at a time**
 - b. Move the **etcd** static pod manifest back to the kubelet manifest directory, in order to make kubelet start the related containers :

```
$ mv /root/manifests-backup/etcd-pod.yaml /etc/kubernetes/manifests
```

- c. Verify that all the **etcd** containers have started:

```
$ crictl ps | grep etcd | grep -v operator
```

Example output

```
38c814767ad983
f79db5a8799fd2c08960ad9ee22f784b9fbe23babe008e8a3bf68323f004c840
28 seconds ago    Running          etcd-health-monitor          2
fe4b9c3d6483c
e1646b15207c6
9d28c15860870e85c91d0e36b45f7a6edd3da757b113ec4abb4507df88b17f06
About a minute ago  Running          etcd-metrics                  0
fe4b9c3d6483c
08ba29b1f58a7
9d28c15860870e85c91d0e36b45f7a6edd3da757b113ec4abb4507df88b17f06
About a minute ago  Running          etcd                          0
fe4b9c3d6483c
2ddc9eda16f53
9d28c15860870e85c91d0e36b45f7a6edd3da757b113ec4abb4507df88b17f06
About a minute ago  Running          etcdctl
```

If the output of this command is empty, wait a few minutes and check again.

10. Check the status of the **etcd** cluster.

- a. On any of the control plane hosts, check the status of the **etcd** cluster with the following command:

```
$ crictl exec -it $(crictl ps | grep etcdctl | awk '{print $1}') etcdctl endpoint status -w table
```

Example output

```
+-----+-----+-----+-----+-----+-----+
+-----+-----+
|   ENDPOINT   |   ID   | VERSION | DB SIZE | IS LEADER | IS LEARNER |
| RAFT TERM | RAFT INDEX | RAFT APPLIED INDEX | ERRORS |
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+
| https://10.0.89.133:2379 | 682e4a83a0cec6c0 | 3.5.0 | 67 MB | true | false |
2 | 218 | 218 | |
| https://10.0.92.74:2379 | 450bcf6999538512 | 3.5.0 | 67 MB | false | false |
2 | 218 | 218 | |
| https://10.0.93.129:2379 | 358efa9c1d91c3d6 | 3.5.0 | 67 MB | false | false |
2 | 218 | 218 | |
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+
```

11. Restart the other static pods.

The following steps must be executed on all control plane hosts, but one host at a time.

- a. Move the Kubernetes API Server static pod manifest back to the kubelet manifest directory to make kubelet start the related containers with the command:

```
$ mv /root/manifests-backup/kube-apiserver-pod.yaml /etc/kubernetes/manifests
```

- b. Verify that all the Kubernetes API Server containers have started:

```
$ crictl ps | grep kube-apiserver | grep -v operator
```



NOTE

if the output of the following command is empty, wait a few minutes and check again.

- c. Repeat the same steps for **kube-controller-manager-pod.yaml** and **kube-scheduler-pod.yaml** files.

- i. Restart the kubelets in all nodes using the following command:

```
$ systemctl restart kubelet
```

- ii. Start the remaining control plane pods using the following command:

```
$ mv /root/manifests-backup/kube-* /etc/kubernetes/manifests/
```

- iii. Check if the **kube-apiserver**, **kube-scheduler** and **kube-controller-manager** pods start correctly:

```
$ crictl ps | grep -E 'kube-(apiserver|scheduler|controller-manager)' | grep -v -E 'operator|guard'
```

- iv. Wipe the OVN databases using the following commands:

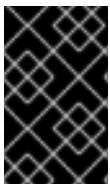
```
for NODE in $(oc get node -o name | sed 's:node/:g')
do
  oc debug node/${NODE} -- chroot /host /bin/bash -c 'rm -f /var/lib/ovn-ic/etc/ovn*.db && systemctl restart ovs-vswitchd ovnsdb-server'
  oc -n openshift-ovn-kubernetes delete pod -l app=ovnkube-node --field-selector=spec.nodeName=${NODE} --wait
  oc -n openshift-ovn-kubernetes wait pod -l app=ovnkube-node --field-selector=spec.nodeName=${NODE} --for condition=ContainersReady --timeout=600s
done
```

6.3.3.5. Additional resources

- [Backing up etcd data](#)
- [Installing a user-provisioned cluster on bare metal](#)
- [Creating a bastion host to access OpenShift Container Platform instances and the control plane nodes with SSH](#)
- [Replacing a bare-metal control plane node](#)

6.3.3.6. Issues and workarounds for restoring a persistent storage state

If your OpenShift Container Platform cluster uses persistent storage of any form, a state of the cluster is typically stored outside etcd. When you restore from an etcd backup, the status of the workloads in OpenShift Container Platform is also restored. However, if the etcd snapshot is old, the status might be invalid or outdated.



IMPORTANT

The contents of persistent volumes (PVs) are never part of the etcd snapshot. When you restore an OpenShift Container Platform cluster from an etcd snapshot, non-critical workloads might gain access to critical data, or vice-versa.

The following are some example scenarios that produce an out-of-date status:

- MySQL database is running in a pod backed up by a PV object. Restoring OpenShift Container Platform from an etcd snapshot does not bring back the volume on the storage provider, and does not produce a running MySQL pod, despite the pod repeatedly attempting to start. You must manually restore this pod by restoring the volume on the storage provider, and then editing the PV to point to the new volume.
- Pod P1 is using volume A, which is attached to node X. If the etcd snapshot is taken while another pod uses the same volume on node Y, then when the etcd restore is performed, pod P1 might not be able to start correctly due to the volume still being attached to node Y. OpenShift

Container Platform is not aware of the attachment, and does not automatically detach it. When this occurs, the volume must be manually detached from node Y so that the volume can attach on node X, and then pod P1 can start.

- Cloud provider or storage provider credentials were updated after the etcd snapshot was taken. This causes any CSI drivers or Operators that depend on the those credentials to not work. You might have to manually update the credentials required by those drivers or Operators.
- A device is removed or renamed from OpenShift Container Platform nodes after the etcd snapshot is taken. The Local Storage Operator creates symlinks for each PV that it manages from **/dev/disk/by-id** or **/dev** directories. This situation might cause the local PVs to refer to devices that no longer exist.
To fix this problem, an administrator must:

1. Manually remove the PVs with invalid devices.
2. Remove symlinks from respective nodes.
3. Delete **LocalVolume** or **LocalVolumeSet** objects (see *Storage → Configuring persistent storage → Persistent storage using local volumes → Deleting the Local Storage Operator Resources*).

6.3.4. Recovering from expired control plane certificates

6.3.4.1. Recovering from expired control plane certificates

The cluster can automatically recover from expired control plane certificates.

However, you must manually approve the pending **node-bootstrapper** certificate signing requests (CSRs) to recover kubelet certificates. For user-provisioned installations, you might also need to approve pending kubelet serving CSRs.

Use the following steps to approve the pending CSRs:

Procedure

1. Get the list of current CSRs:

```
$ oc get csr
```

Example output

NAME	AGE	SIGNERNAME	REQUESTOR	CONDITION
csr-2s94x	8m3s	kubernetes.io/kubelet-serving	system:node:<node_name>	Pending 1
csr-4bd6t	8m3s	kubernetes.io/kubelet-serving	system:node:<node_name>	Pending
csr-4hl85	13m	kubernetes.io/kube-apiserver-client-kubelet	system:serviceaccount:openshift-machine-config-operator:node-bootstrapper	Pending 2
csr-zhthp	3m8s	kubernetes.io/kube-apiserver-client-kubelet	system:serviceaccount:openshift-machine-config-operator:node-bootstrapper	Pending
...				

- 1 A pending kubelet service CSR (for user-provisioned installations).
- 2 A pending **node-bootstrapper** CSR.

2. Review the details of a CSR to verify that it is valid:

```
$ oc describe csr <csr_name> 1
```

- 1 **<csr_name>** is the name of a CSR from the list of current CSRs.

3. Approve each valid **node-bootstrapper** CSR:

```
$ oc adm certificate approve <csr_name>
```

4. For user-provisioned installations, approve each valid kubelet serving CSR:

```
$ oc adm certificate approve <csr_name>
```