OpenShift Container Platform 4.20

Operator APIs

Reference guide for Operator APIs

Last Updated: 2025-10-17

OpenShift Container Platform 4.20 Operator APIs

Reference guide for Operator APIs

Legal Notice

Copyright © 2025 Red Hat, Inc.

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, the Red Hat 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 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 describes the OpenShift Container Platform Operator API objects and their detailed specifications.

Table of Contents

CHAPTER 1. OPERATOR APIS 18
1.1. AUTHENTICATION [OPERATOR.OPENSHIFT.IO/V1] 18
1.2. CLOUDCREDENTIAL [OPERATOR.OPENSHIFT.IO/V1] 18
1.3. CLUSTERCSIDRIVER [OPERATOR.OPENSHIFT.IO/V1] 18
1.4. CONSOLE [OPERATOR.OPENSHIFT.IO/V1] 18
1.5. CONFIG [OPERATOR.OPENSHIFT.IO/V1] 18
1.6. CONFIG [IMAGEREGISTRY.OPERATOR.OPENSHIFT.IO/V1] 19
1.7. CONFIG [SAMPLES.OPERATOR.OPENSHIFT.IO/V1] 19
1.8. CSISNAPSHOTCONTROLLER [OPERATOR.OPENSHIFT.IO/V1] 19
1.9. DNS [OPERATOR.OPENSHIFT.IO/V1] 19
1.10. DNSRECORD [INGRESS.OPERATOR.OPENSHIFT.IO/V1] 20
1.11. ETCD [OPERATOR.OPENSHIFT.IO/V1] 20
1.12. IMAGECONTENTSOURCEPOLICY [OPERATOR.OPENSHIFT.IO/V1ALPHA1] 20
1.13. IMAGEPRUNER [IMAGEREGISTRY.OPERATOR.OPENSHIFT.IO/V1] 20
1.14. INGRESSCONTROLLER [OPERATOR.OPENSHIFT.IO/V1] 21
1.15. INSIGHTSOPERATOR [OPERATOR.OPENSHIFT.IO/V1] 21
1.16. KUBEAPISERVER [OPERATOR.OPENSHIFT.IO/V1] 21
1.17. KUBECONTROLLERMANAGER [OPERATOR.OPENSHIFT.IO/V1] 21
1.18. KUBESCHEDULER [OPERATOR.OPENSHIFT.IO/V1] 22
1.19. KUBESTORAGEVERSIONMIGRATOR [OPERATOR.OPENSHIFT.IO/V1] 22
1.20. MACHINECONFIGURATION [OPERATOR.OPENSHIFT.IO/V1] 22
1.21. NETWORK [OPERATOR.OPENSHIFT.IO/V1] 22
1.22. OPENSHIFTAPISERVER [OPERATOR.OPENSHIFT.IO/V1] 22
1.23. OPENSHIFTCONTROLLERMANAGER [OPERATOR.OPENSHIFT.IO/V1] 23
1.24. OPERATORPKI [NETWORK.OPERATOR.OPENSHIFT.IO/V1] 23
1.25. SERVICECA [OPERATOR.OPENSHIFT.IO/V1] 24
1.26. STORAGE [OPERATOR.OPENSHIFT.IO/V1] 24

CHAPTER 2. AUTHENTICATION [OPERATOR.OPENSHIFT.IO/V1] 25
2.1. SPECIFICATION 25
2.1.1. .spec 26
2.1.2. .status 27
2.1.3. .status.conditions 28
2.1.4. .status.conditions[] 28
2.1.5. .status.generations 29
2.1.6. .status.generations[] 29
2.1.7. .status.oauthAPIServer 30
2.2. API ENDPOINTS 30
2.2.1. /apis/operator.openshift.io/v1/authentications 31
2.2.2. /apis/operator.openshift.io/v1/authentications/{name} 33
2.2.3. /apis/operator.openshift.io/v1/authentications/{name}/status 35

CHAPTER 3. CLOUDCREDENTIAL [OPERATOR.OPENSHIFT.IO/V1] 39
3.1. SPECIFICATION 39
3.1.1. .spec 40
3.1.2. .status 41
3.1.3. .status.conditions 42
3.1.4. .status.conditions[] 42
3.1.5. .status.generations 43
3.1.6. .status.generations[] 43
3.2. API ENDPOINTS 44

3.2.1. /apis/operator.openshift.io/v1/cloudcredentials 45
3.2.2. /apis/operator.openshift.io/v1/cloudcredentials/{name} 46
3.2.3. /apis/operator.openshift.io/v1/cloudcredentials/{name}/status 49

CHAPTER 4. CLUSTERCSIDRIVER [OPERATOR.OPENSHIFT.IO/V1] 53
4.1. SPECIFICATION 53
4.1.1. .spec 54
4.1.2. .spec.driverConfig 55
4.1.3. .spec.driverConfig.aws 56
4.1.4. .spec.driverConfig.aws.efsVolumeMetrics 57
4.1.5. .spec.driverConfig.aws.efsVolumeMetrics.recursiveWalk 57
4.1.6. .spec.driverConfig.azure 58
4.1.7. .spec.driverConfig.azure.diskEncryptionSet 58
4.1.8. .spec.driverConfig.gcp 59
4.1.9. .spec.driverConfig.gcp.kmsKey 60
4.1.10. .spec.driverConfig.ibmcloud 61
4.1.11. .spec.driverConfig.vSphere 61
4.1.12. .status 63
4.1.13. .status.conditions 64
4.1.14. .status.conditions[] 64
4.1.15. .status.generations 65
4.1.16. .status.generations[] 65
4.2. API ENDPOINTS 66
4.2.1. /apis/operator.openshift.io/v1/clustercsidrivers 66
4.2.2. /apis/operator.openshift.io/v1/clustercsidrivers/{name} 68
4.2.3. /apis/operator.openshift.io/v1/clustercsidrivers/{name}/status 71

CHAPTER 5. CONSOLE [OPERATOR.OPENSHIFT.IO/V1] 74
5.1. SPECIFICATION 74
5.1.1. .spec 75
5.1.2. .spec.customization 77
5.1.3. .spec.customization.addPage 79
5.1.4. .spec.customization.capabilities 80
5.1.5. .spec.customization.capabilities[] 80
5.1.6. .spec.customization.capabilities[].visibility 80
5.1.7. .spec.customization.customLogoFile 81
5.1.8. .spec.customization.developerCatalog 81
5.1.9. .spec.customization.developerCatalog.categories 82
5.1.10. .spec.customization.developerCatalog.categories[] 82
5.1.11. .spec.customization.developerCatalog.categories[].subcategories 83
5.1.12. .spec.customization.developerCatalog.categories[].subcategories[] 83
5.1.13. .spec.customization.developerCatalog.types 84
5.1.14. .spec.customization.logos 85
5.1.15. .spec.customization.logos[] 85
5.1.16. .spec.customization.logos[].themes 86
5.1.17. .spec.customization.logos[].themes[] 86
5.1.18. .spec.customization.logos[].themes[].source 88
5.1.19. .spec.customization.logos[].themes[].source.configMap 88
5.1.20. .spec.customization.perspectives 89
5.1.21. .spec.customization.perspectives[] 89
5.1.22. .spec.customization.perspectives[].pinnedResources 90
5.1.23. .spec.customization.perspectives[].pinnedResources[] 90
5.1.24. .spec.customization.perspectives[].visibility 91

5.1.25. .spec.customization.perspectives[].visibility.accessReview 92
5.1.26. .spec.customization.perspectives[].visibility.accessReview.missing 93
5.1.27. .spec.customization.perspectives[].visibility.accessReview.missing[] 93
5.1.28. .spec.customization.perspectives[].visibility.accessReview.missing[].fieldSelector 94
5.1.29. .spec.customization.perspectives[].visibility.accessReview.missing[].fieldSelector.requirements 95
5.1.30. .spec.customization.perspectives[].visibility.accessReview.missing[].fieldSelector.requirements[] 95
5.1.31. .spec.customization.perspectives[].visibility.accessReview.missing[].labelSelector 96
5.1.32. .spec.customization.perspectives[].visibility.accessReview.missing[].labelSelector.requirements 97
5.1.33. .spec.customization.perspectives[].visibility.accessReview.missing[].labelSelector.requirements[] 97
5.1.34. .spec.customization.perspectives[].visibility.accessReview.required 98
5.1.35. .spec.customization.perspectives[].visibility.accessReview.required[] 98
5.1.36. .spec.customization.perspectives[].visibility.accessReview.required[].fieldSelector 99
5.1.37. .spec.customization.perspectives[].visibility.accessReview.required[].fieldSelector.requirements 100
5.1.38. .spec.customization.perspectives[].visibility.accessReview.required[].fieldSelector.requirements[] 100
5.1.39. .spec.customization.perspectives[].visibility.accessReview.required[].labelSelector 101
5.1.40. .spec.customization.perspectives[].visibility.accessReview.required[].labelSelector.requirements 102
5.1.41. .spec.customization.perspectives[].visibility.accessReview.required[].labelSelector.requirements[] 102
5.1.42. .spec.customization.projectAccess 103
5.1.43. .spec.customization.quickStarts 103
5.1.44. .spec.ingress 104
5.1.45. .spec.providers 104
5.1.46. .spec.providers.statuspage 105
5.1.47. .spec.route 105
5.1.48. .spec.route.secret 106
5.1.49. .status 106
5.1.50. .status.conditions 107
5.1.51. .status.conditions[] 107
5.1.52. .status.generations 108
5.1.53. .status.generations[] 108
5.2. API ENDPOINTS 109
5.2.1. /apis/operator.openshift.io/v1/consoles 110
5.2.2. /apis/operator.openshift.io/v1/consoles/{name} 111
5.2.3. /apis/operator.openshift.io/v1/consoles/{name}/status 114

CHAPTER 6. CONFIG [OPERATOR.OPENSHIFT.IO/V1] 118
6.1. SPECIFICATION 118
6.1.1. .spec 119
6.1.2. .status 120
6.1.3. .status.conditions 121
6.1.4. .status.conditions[] 121
6.1.5. .status.generations 122
6.1.6. .status.generations[] 122
6.2. API ENDPOINTS 123
6.2.1. /apis/operator.openshift.io/v1/configs 124
6.2.2. /apis/operator.openshift.io/v1/configs/{name} 125
6.2.3. /apis/operator.openshift.io/v1/configs/{name}/status 128

CHAPTER 7. CONFIG [IMAGEREGISTRY.OPERATOR.OPENSHIFT.IO/V1] 132
7.1. SPECIFICATION 132
7.1.1. .spec 133
7.1.2. .spec.affinity 136
7.1.3. .spec.affinity.nodeAffinity 136
7.1.4. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution 137

7.1.5. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[] 138
7.1.6. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference 138
7.1.7. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpressi 139
7.1.8. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpress 139
7.1.9. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields 140
7.1.10. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields[140
7.1.11. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution 141
7.1.12. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms 141
7.1.13. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[] 141
7.1.14. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].match 142
7.1.15. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].match 142
7.1.16. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].match 143
7.1.17. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].match 143
7.1.18. .spec.affinity.podAffinity 144
7.1.19. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution 145
7.1.20. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[] 145
7.1.21. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm 146
7.1.22. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSele 148
7.1.23. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSele 149
7.1.24. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSele 149
7.1.25. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespa 150
7.1.26. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespa 150
7.1.27. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namespa 151
7.1.28. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution 151
7.1.29. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[] 151
7.1.30. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector 154
7.1.31. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpress 154
7.1.32. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpres 155
7.1.33. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector 155
7.1.34. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.match 156
7.1.35. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.match 156
7.1.36. .spec.affinity.podAntiAffinity 157
7.1.37. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution 158
7.1.38. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[] 158
7.1.39. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm 159
7.1.40. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labe

161
7.1.41. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labels 162
7.1.42. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.label 162
7.1.43. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.name 163
7.1.44. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.name 163
7.1.45. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.name 164
7.1.46. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution 164
7.1.47. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[] 164
7.1.48. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector 167
7.1.49. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchEx 167
7.1.50. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchEx 168
7.1.51. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector 168
7.1.52. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.ma 169
7.1.53. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.ma 169
7.1.54. .spec.proxy 170
7.1.55. .spec.requests 170
7.1.56. .spec.requests.read 171
7.1.57. .spec.requests.write 171
7.1.58. .spec.resources 172
7.1.59. .spec.resources.claims 172
7.1.60. .spec.resources.claims[] 173
7.1.61. .spec.routes 173
7.1.62. .spec.routes[] 173
7.1.63. .spec.storage 174
7.1.64. .spec.storage.azure 175
7.1.65. .spec.storage.azure.networkAccess 175
7.1.66. .spec.storage.azure.networkAccess.internal 176
7.1.67. .spec.storage.emptyDir 178
7.1.68. .spec.storage.gcs 179
7.1.69. .spec.storage.ibmcos 179
7.1.70. .spec.storage.oss 180
7.1.71. .spec.storage.oss.encryption 181
7.1.72. .spec.storage.oss.encryption.kms 182
7.1.73. .spec.storage.pvc 182
7.1.74. .spec.storage.s3 183
7.1.75. .spec.storage.s3.cloudFront 184
7.1.76. .spec.storage.s3.cloudFront.privateKey 185
7.1.77. .spec.storage.s3.trustedCA 185
7.1.78. .spec.storage.swift 186
7.1.79. .spec.tolerations 187
7.1.80. .spec.tolerations[] 187
7.1.81. .spec.topologySpreadConstraints 188
7.1.82. .spec.topologySpreadConstraints[] 188
7.1.83. .spec.topologySpreadConstraints[].labelSelector 194

7.1.84. .spec.topologySpreadConstraints[].labelSelector.matchExpressions 195
7.1.85. .spec.topologySpreadConstraints[].labelSelector.matchExpressions[] 195
7.1.86. .status 196
7.1.87. .status.conditions 197
7.1.88. .status.conditions[] 197
7.1.89. .status.generations 198
7.1.90. .status.generations[] 198
7.1.91. .status.storage 199
7.1.92. .status.storage.azure 200
7.1.93. .status.storage.azure.networkAccess 201
7.1.94. .status.storage.azure.networkAccess.internal 202
7.1.95. .status.storage.emptyDir 204
7.1.96. .status.storage.gcs 205
7.1.97. .status.storage.ibmcos 205
7.1.98. .status.storage.oss 206
7.1.99. .status.storage.oss.encryption 207
7.1.100. .status.storage.oss.encryption.kms 208
7.1.101. .status.storage.pvc 208
7.1.102. .status.storage.s3 209
7.1.103. .status.storage.s3.cloudFront 210
7.1.104. .status.storage.s3.cloudFront.privateKey 211
7.1.105. .status.storage.s3.trustedCA 211
7.1.106. .status.storage.swift 212
7.2. API ENDPOINTS 213
7.2.1. /apis/imageregistry.operator.openshift.io/v1/configs 213
7.2.2. /apis/imageregistry.operator.openshift.io/v1/configs/{name} 215
7.2.3. /apis/imageregistry.operator.openshift.io/v1/configs/{name}/status 218

CHAPTER 8. CONFIG [SAMPLES.OPERATOR.OPENSHIFT.IO/V1] 222
8.1. SPECIFICATION 222
8.1.1. .spec 223
8.1.2. .status 225
8.1.3. .status.conditions 226
8.1.4. .status.conditions[] 226
8.2. API ENDPOINTS 227
8.2.1. /apis/samples.operator.openshift.io/v1/configs 228
8.2.2. /apis/samples.operator.openshift.io/v1/configs/{name} 229
8.2.3. /apis/samples.operator.openshift.io/v1/configs/{name}/status 232

CHAPTER 9. CSISNAPSHOTCONTROLLER [OPERATOR.OPENSHIFT.IO/V1] 236
9.1. SPECIFICATION 236
9.1.1. .spec 237
9.1.2. .status 238
9.1.3. .status.conditions 239
9.1.4. .status.conditions[] 239
9.1.5. .status.generations 240
9.1.6. .status.generations[] 240
9.2. API ENDPOINTS 241
9.2.1. /apis/operator.openshift.io/v1/csisnapshotcontrollers 242
9.2.2. /apis/operator.openshift.io/v1/csisnapshotcontrollers/{name} 243
9.2.3. /apis/operator.openshift.io/v1/csisnapshotcontrollers/{name}/status 246

CHAPTER 10. DNS [OPERATOR.OPENSHIFT.IO/V1] 250
10.1. SPECIFICATION 250

10.1.1. .spec 251
10.1.2. .spec.cache 253
10.1.3. .spec.nodePlacement 254
10.1.4. .spec.nodePlacement.tolerations 255
10.1.5. .spec.nodePlacement.tolerations[] 256
10.1.6. .spec.servers 257
10.1.7. .spec.servers[] 257
10.1.8. .spec.servers[].forwardPlugin 257
10.1.9. .spec.servers[].forwardPlugin.transportConfig 259
10.1.10. .spec.servers[].forwardPlugin.transportConfig.tls 260
10.1.11. .spec.servers[].forwardPlugin.transportConfig.tls.caBundle 261
10.1.12. .spec.upstreamResolvers 262
10.1.13. .spec.upstreamResolvers.transportConfig 264
10.1.14. .spec.upstreamResolvers.transportConfig.tls 265
10.1.15. .spec.upstreamResolvers.transportConfig.tls.caBundle 266
10.1.16. .spec.upstreamResolvers.upstreams 267
10.1.17. .spec.upstreamResolvers.upstreams[] 267
10.1.18. .status 268
10.1.19. .status.conditions 269
10.1.20. .status.conditions[] 269
10.2. API ENDPOINTS 270
10.2.1. /apis/operator.openshift.io/v1/dnses 270
10.2.2. /apis/operator.openshift.io/v1/dnses/{name} 272
10.2.3. /apis/operator.openshift.io/v1/dnses/{name}/status 275

CHAPTER 11. DNSRECORD [INGRESS.OPERATOR.OPENSHIFT.IO/V1] 279
11.1. SPECIFICATION 279
11.1.1. .spec 280
11.1.2. .status 281
11.1.3. .status.zones 282
11.1.4. .status.zones[] 282
11.1.5. .status.zones[].conditions 282
11.1.6. .status.zones[].conditions[] 282
11.1.7. .status.zones[].dnsZone 283
11.2. API ENDPOINTS 284
11.2.1. /apis/ingress.operator.openshift.io/v1/dnsrecords 285
11.2.2. /apis/ingress.operator.openshift.io/v1/namespaces/{namespace}/dnsrecords 285
11.2.3. /apis/ingress.operator.openshift.io/v1/namespaces/{namespace}/dnsrecords/{name} 287
11.2.4. /apis/ingress.operator.openshift.io/v1/namespaces/{namespace}/dnsrecords/{name}/status 290

CHAPTER 12. ETCD [OPERATOR.OPENSHIFT.IO/V1] 294
12.1. SPECIFICATION 294
12.1.1. .spec 295
12.1.2. .status 296
12.1.3. .status.conditions 298
12.1.4. .status.conditions[] 298
12.1.5. .status.generations 298
12.1.6. .status.generations[] 299
12.1.7. .status.nodeStatuses 299
12.1.8. .status.nodeStatuses[] 299
12.2. API ENDPOINTS 301
12.2.1. /apis/operator.openshift.io/v1/etcds 301
12.2.2. /apis/operator.openshift.io/v1/etcds/{name} 303

12.2.3. /apis/operator.openshift.io/v1/etcds/{name}/status 306

CHAPTER 13. IMAGECONTENTSOURCEPOLICY [OPERATOR.OPENSHIFT.IO/V1ALPHA1] 309
13.1. SPECIFICATION 309
13.1.1. .spec 310
13.1.2. .spec.repositoryDigestMirrors 312
13.1.3. .spec.repositoryDigestMirrors[] 312
13.2. API ENDPOINTS 313
13.2.1. /apis/operator.openshift.io/v1alpha1/imagecontentsourcepolicies 313
13.2.2. /apis/operator.openshift.io/v1alpha1/imagecontentsourcepolicies/{name} 315
13.2.3. /apis/operator.openshift.io/v1alpha1/imagecontentsourcepolicies/{name}/status 318

CHAPTER 14. IMAGEPRUNER [IMAGEREGISTRY.OPERATOR.OPENSHIFT.IO/V1] 322
14.1. SPECIFICATION 322
14.1.1. .spec 323
14.1.2. .spec.affinity 325
14.1.3. .spec.affinity.nodeAffinity 325
14.1.4. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution 326
14.1.5. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[] 327
14.1.6. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference 327
14.1.7. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpress 328
14.1.8. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchExpress 328
14.1.9. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields 329
14.1.10. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[].preference.matchFields[329
14.1.11. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution 330
14.1.12. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms 330
14.1.13. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[] 330
14.1.14. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matc 331
14.1.15. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matc 331
14.1.16. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matc 332
14.1.17. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.nodeSelectorTerms[].matc 332
14.1.18. .spec.affinity.podAffinity 333
14.1.19. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution 334
14.1.20. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[] 334
14.1.21. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm 335
14.1.22. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSel 337
14.1.23. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSel 338
14.1.24. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labelSel 338
14.1.25. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namesp 339
14.1.26. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namesp 339
14.1.27. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.namesp

340
14.1.28. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution 340
14.1.29. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[] 340
14.1.30. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector 343
14.1.31. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpres 343
14.1.32. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchExpres 344
14.1.33. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector 344
14.1.34. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.match 345
14.1.35. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.match 345
14.1.36. .spec.affinity.podAntiAffinity 346
14.1.37. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution 347
14.1.38. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[] 347
14.1.39. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm 348
14.1.40. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labe 350
14.1.41. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labe 351
14.1.42. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.labe 351
14.1.43. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.nam 352
14.1.44. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.nam 352
14.1.45. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecution[].podAffinityTerm.nam 353
14.1.46. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution 353
14.1.47. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[] 353
14.1.48. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector 356
14.1.49. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchE 356
14.1.50. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].labelSelector.matchE 357
14.1.51. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector 357
14.1.52. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.m 358
14.1.53. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecution[].namespaceSelector.m 358
14.1.54. .spec.resources 359
14.1.55. .spec.resources.claims 360
14.1.56. .spec.resources.claims[] 360
14.1.57. .spec.tolerations 361
14.1.58. .spec.tolerations[] 361
14.1.59. .status 362
14.1.60. .status.conditions 362
14.1.61. .status.conditions[] 362
14.2. API ENDPOINTS 363
14.2.1. /apis/imageregistry.operator.openshift.io/v1/imagepruners 364
14.2.2. /apis/imageregistry.operator.openshift.io/v1/imagepruners/{name} 365

14.2.3. /apis/imageregistry.operator.openshift.io/v1/imagepruners/{name}/status 368

CHAPTER 15. INGRESSCONTROLLER [OPERATOR.OPENSHIFT.IO/V1] 372
15.1. SPECIFICATION 372
15.1.1. .spec 373
15.1.2. .spec.clientTLS 382
15.1.3. .spec.clientTLS.clientCA 383
15.1.4. .spec.defaultCertificate 384
15.1.5. .spec.endpointPublishingStrategy 384
15.1.6. .spec.endpointPublishingStrategy.hostNetwork 387
15.1.7. .spec.endpointPublishingStrategy.loadBalancer 389
15.1.8. .spec.endpointPublishingStrategy.loadBalancer.providerParameters 391
15.1.9. .spec.endpointPublishingStrategy.loadBalancer.providerParameters.aws 392
15.1.10. .spec.endpointPublishingStrategy.loadBalancer.providerParameters.aws.classicLoadBalancer 393
15.1.11. .spec.endpointPublishingStrategy.loadBalancer.providerParameters.aws.classicLoadBalancer.subnets 394
15.1.12. .spec.endpointPublishingStrategy.loadBalancer.providerParameters.aws.networkLoadBalancer 395
15.1.13. .spec.endpointPublishingStrategy.loadBalancer.providerParameters.aws.networkLoadBalancer.subnets 396
15.1.14. .spec.endpointPublishingStrategy.loadBalancer.providerParameters.gcp 397
15.1.15. .spec.endpointPublishingStrategy.loadBalancer.providerParameters.ibm 398
15.1.16. .spec.endpointPublishingStrategy.loadBalancer.providerParameters.openstack 399
15.1.17. .spec.endpointPublishingStrategy.nodePort 400
15.1.18. .spec.endpointPublishingStrategy.private 401
15.1.19. .spec.httpCompression 402
15.1.20. .spec.httpErrorCodePages 403
15.1.21. .spec.httpHeaders 403
15.1.22. .spec.httpHeaders.actions 406
15.1.23. .spec.httpHeaders.actions.request 408
15.1.24. .spec.httpHeaders.actions.request[] 408
15.1.25. .spec.httpHeaders.actions.request[].action 409
15.1.26. .spec.httpHeaders.actions.request[].action.set 410
15.1.27. .spec.httpHeaders.actions.response 410
15.1.28. .spec.httpHeaders.actions.response[] 411
15.1.29. .spec.httpHeaders.actions.response[].action 411
15.1.30. .spec.httpHeaders.actions.response[].action.set 412
15.1.31. .spec.httpHeaders.uniqueId 413
15.1.32. .spec.logging 414
15.1.33. .spec.logging.access 414
15.1.34. .spec.logging.access.destination 416
15.1.35. .spec.logging.access.destination.container 417
15.1.36. .spec.logging.access.destination.syslog 418
15.1.37. .spec.logging.access.httpCaptureHeaders 419
15.1.38. .spec.namespaceSelector 419
15.1.39. .spec.namespaceSelector.matchExpressions 420
15.1.40. .spec.namespaceSelector.matchExpressions[] 420
15.1.41. .spec.nodePlacement 421
15.1.42. .spec.nodePlacement.nodeSelector 423
15.1.43. .spec.nodePlacement.nodeSelector.matchExpressions 424
15.1.44. .spec.nodePlacement.nodeSelector.matchExpressions[] 424
15.1.45. .spec.nodePlacement.tolerations 425
15.1.46. .spec.nodePlacement.tolerations[] 425
15.1.47. .spec.routeAdmission 426

15.1.48. .spec.routeSelector 427
15.1.49. .spec.routeSelector.matchExpressions 428
15.1.50. .spec.routeSelector.matchExpressions[] 428
15.1.51. .spec.tlsSecurityProfile 429
15.1.52. .spec.tuningOptions 434
15.1.53. .status 441
15.1.54. .status.conditions 443
15.1.55. .status.conditions[] 444
15.1.56. .status.endpointPublishingStrategy 445
15.1.57. .status.endpointPublishingStrategy.hostNetwork 447
15.1.58. .status.endpointPublishingStrategy.loadBalancer 449
15.1.59. .status.endpointPublishingStrategy.loadBalancer.providerParameters 451
15.1.60. .status.endpointPublishingStrategy.loadBalancer.providerParameters.aws 452
15.1.61. .status.endpointPublishingStrategy.loadBalancer.providerParameters.aws.classicLoadBalancer 453
15.1.62. .status.endpointPublishingStrategy.loadBalancer.providerParameters.aws.classicLoadBalancer.subnets 454
15.1.63. .status.endpointPublishingStrategy.loadBalancer.providerParameters.aws.networkLoadBalancer 455
15.1.64. .status.endpointPublishingStrategy.loadBalancer.providerParameters.aws.networkLoadBalancer.subnets 456
15.1.65. .status.endpointPublishingStrategy.loadBalancer.providerParameters.gcp 457
15.1.66. .status.endpointPublishingStrategy.loadBalancer.providerParameters.ibm 458
15.1.67. .status.endpointPublishingStrategy.loadBalancer.providerParameters.openstack 459
15.1.68. .status.endpointPublishingStrategy.nodePort 460
15.1.69. .status.endpointPublishingStrategy.private 461
15.1.70. .status.namespaceSelector 462
15.1.71. .status.namespaceSelector.matchExpressions 463
15.1.72. .status.namespaceSelector.matchExpressions[] 463
15.1.73. .status.routeSelector 464
15.1.74. .status.routeSelector.matchExpressions 464
15.1.75. .status.routeSelector.matchExpressions[] 465
15.1.76. .status.tlsProfile 465
15.2. API ENDPOINTS 466
15.2.1. /apis/operator.openshift.io/v1/ingresscontrollers 467
15.2.2. /apis/operator.openshift.io/v1/namespaces/{namespace}/ingresscontrollers 467
15.2.3. /apis/operator.openshift.io/v1/namespaces/{namespace}/ingresscontrollers/{name} 469
15.2.4. /apis/operator.openshift.io/v1/namespaces/{namespace}/ingresscontrollers/{name}/scale 472
15.2.5. /apis/operator.openshift.io/v1/namespaces/{namespace}/ingresscontrollers/{name}/status 474

CHAPTER 16. INSIGHTSOPERATOR [OPERATOR.OPENSHIFT.IO/V1] 478
16.1. SPECIFICATION 478
16.1.1. .spec 479
16.1.2. .status 480
16.1.3. .status.conditions 481
16.1.4. .status.conditions[] 481
16.1.5. .status.gatherStatus 481
16.1.6. .status.gatherStatus.gatherers 482
16.1.7. .status.gatherStatus.gatherers[] 482
16.1.8. .status.gatherStatus.gatherers[].conditions 483
16.1.9. .status.gatherStatus.gatherers[].conditions[] 483
16.1.10. .status.generations 485
16.1.11. .status.generations[] 485
16.1.12. .status.insightsReport 486
16.1.13. .status.insightsReport.healthChecks 486

16.1.14. .status.insightsReport.healthChecks[] 486
16.2. API ENDPOINTS 487
16.2.1. /apis/operator.openshift.io/v1/insightsoperators 488
16.2.2. /apis/operator.openshift.io/v1/insightsoperators/{name} 490
16.2.3. /apis/operator.openshift.io/v1/insightsoperators/{name}/scale 492
16.2.4. /apis/operator.openshift.io/v1/insightsoperators/{name}/status 495

CHAPTER 17. KUBEAPISERVER [OPERATOR.OPENSHIFT.IO/V1] 499
17.1. SPECIFICATION 499
17.1.1. .spec 500
17.1.2. .status 501
17.1.3. .status.conditions 503
17.1.4. .status.conditions[] 503
17.1.5. .status.generations 504
17.1.6. .status.generations[] 504
17.1.7. .status.nodeStatuses 505
17.1.8. .status.nodeStatuses[] 505
17.1.9. .status.serviceAccountIssuers 506
17.1.10. .status.serviceAccountIssuers[] 507
17.2. API ENDPOINTS 507
17.2.1. /apis/operator.openshift.io/v1/kubeapiservers 507
17.2.2. /apis/operator.openshift.io/v1/kubeapiservers/{name} 509
17.2.3. /apis/operator.openshift.io/v1/kubeapiservers/{name}/status 512

CHAPTER 18. KUBECONTROLLERMANAGER [OPERATOR.OPENSHIFT.IO/V1] 516
18.1. SPECIFICATION 516
18.1.1. .spec 517
18.1.2. .status 519
18.1.3. .status.conditions 520
18.1.4. .status.conditions[] 520
18.1.5. .status.generations 521
18.1.6. .status.generations[] 521
18.1.7. .status.nodeStatuses 522
18.1.8. .status.nodeStatuses[] 522
18.2. API ENDPOINTS 523
18.2.1. /apis/operator.openshift.io/v1/kubecontrollermanagers 524
18.2.2. /apis/operator.openshift.io/v1/kubecontrollermanagers/{name} 526
18.2.3. /apis/operator.openshift.io/v1/kubecontrollermanagers/{name}/status 529

CHAPTER 19. KUBESCHEDULER [OPERATOR.OPENSHIFT.IO/V1] 532
19.1. SPECIFICATION 532
19.1.1. .spec 533
19.1.2. .status 534
19.1.3. .status.conditions 535
19.1.4. .status.conditions[] 536
19.1.5. .status.generations 536
19.1.6. .status.generations[] 536
19.1.7. .status.nodeStatuses 537
19.1.8. .status.nodeStatuses[] 537
19.2. API ENDPOINTS 538
19.2.1. /apis/operator.openshift.io/v1/kubeschedulers 539
19.2.2. /apis/operator.openshift.io/v1/kubeschedulers/{name} 541
19.2.3. /apis/operator.openshift.io/v1/kubeschedulers/{name}/status 544

CHAPTER 20. KUBESTORAGEVERSIONMIGRATOR [OPERATOR.OPENSHIFT.IO/V1] 547
20.1. SPECIFICATION 547
20.1.1. .spec 548
20.1.2. .status 549
20.1.3. .status.conditions 550
20.1.4. .status.conditions[] 550
20.1.5. .status.generations 551
20.1.6. .status.generations[] 551
20.2. API ENDPOINTS 552
20.2.1. /apis/operator.openshift.io/v1/kubestorageversionmigrators 552
20.2.2. /apis/operator.openshift.io/v1/kubestorageversionmigrators/{name} 554
20.2.3. /apis/operator.openshift.io/v1/kubestorageversionmigrators/{name}/status 557

CHAPTER 21. MACHINECONFIGURATION [OPERATOR.OPENSHIFT.IO/V1] 561
21.1. SPECIFICATION 561
21.1.1. .spec 562
21.1.2. .spec.managedBootImages 564
21.1.3. .spec.managedBootImages.machineManagers 565
21.1.4. .spec.managedBootImages.machineManagers[] 565
21.1.5. .spec.managedBootImages.machineManagers[].selection 566
21.1.6. .spec.managedBootImages.machineManagers[].selection.partial 567
21.1.7. .spec.managedBootImages.machineManagers[].selection.partial.machineResourceSelector 567
21.1.8. .spec.managedBootImages.machineManagers[].selection.partial.machineResourceSelector.matchExpress 568
21.1.9. .spec.managedBootImages.machineManagers[].selection.partial.machineResourceSelector.matchExpress 568
21.1.10. .spec.nodeDisruptionPolicy 569
21.1.11. .spec.nodeDisruptionPolicy.files 570
21.1.12. .spec.nodeDisruptionPolicy.files[] 570
21.1.13. .spec.nodeDisruptionPolicy.files[].actions 571
21.1.14. .spec.nodeDisruptionPolicy.files[].actions[] 571
21.1.15. .spec.nodeDisruptionPolicy.files[].actions[].reload 572
21.1.16. .spec.nodeDisruptionPolicy.files[].actions[].restart 572
21.1.17. .spec.nodeDisruptionPolicy.sshkey 573
21.1.18. .spec.nodeDisruptionPolicy.sshkey.actions 574
21.1.19. .spec.nodeDisruptionPolicy.sshkey.actions[] 574
21.1.20. .spec.nodeDisruptionPolicy.sshkey.actions[].reload 575
21.1.21. .spec.nodeDisruptionPolicy.sshkey.actions[].restart 575
21.1.22. .spec.nodeDisruptionPolicy.units 576
21.1.23. .spec.nodeDisruptionPolicy.units[] 576
21.1.24. .spec.nodeDisruptionPolicy.units[].actions 577
21.1.25. .spec.nodeDisruptionPolicy.units[].actions[] 578
21.1.26. .spec.nodeDisruptionPolicy.units[].actions[].reload 578
21.1.27. .spec.nodeDisruptionPolicy.units[].actions[].restart 579
21.1.28. .status 579
21.1.29. .status.conditions 580
21.1.30. .status.conditions[] 580
21.1.31. .status.managedBootImagesStatus 581
21.1.32. .status.managedBootImagesStatus.machineManagers 582
21.1.33. .status.managedBootImagesStatus.machineManagers[] 582
21.1.34. .status.managedBootImagesStatus.machineManagers[].selection 583
21.1.35. .status.managedBootImagesStatus.machineManagers[].selection.partial 584
21.1.36. .status.managedBootImagesStatus.machineManagers[].selection.partial.machineResourceSelector

584
21.1.37. .status.managedBootImagesStatus.machineManagers[].selection.partial.machineResourceSelector.matc 585
21.1.38. .status.managedBootImagesStatus.machineManagers[].selection.partial.machineResourceSelector.matc 585
21.1.39. .status.nodeDisruptionPolicyStatus 586
21.1.40. .status.nodeDisruptionPolicyStatus.clusterPolicies 586
21.1.41. .status.nodeDisruptionPolicyStatus.clusterPolicies.files 587
21.1.42. .status.nodeDisruptionPolicyStatus.clusterPolicies.files[] 587
21.1.43. .status.nodeDisruptionPolicyStatus.clusterPolicies.files[].actions 588
21.1.44. .status.nodeDisruptionPolicyStatus.clusterPolicies.files[].actions[] 588
21.1.45. .status.nodeDisruptionPolicyStatus.clusterPolicies.files[].actions[].reload 589
21.1.46. .status.nodeDisruptionPolicyStatus.clusterPolicies.files[].actions[].restart 589
21.1.47. .status.nodeDisruptionPolicyStatus.clusterPolicies.sshkey 590
21.1.48. .status.nodeDisruptionPolicyStatus.clusterPolicies.sshkey.actions 591
21.1.49. .status.nodeDisruptionPolicyStatus.clusterPolicies.sshkey.actions[] 591
21.1.50. .status.nodeDisruptionPolicyStatus.clusterPolicies.sshkey.actions[].reload 592
21.1.51. .status.nodeDisruptionPolicyStatus.clusterPolicies.sshkey.actions[].restart 592
21.1.52. .status.nodeDisruptionPolicyStatus.clusterPolicies.units 593
21.1.53. .status.nodeDisruptionPolicyStatus.clusterPolicies.units[] 593
21.1.54. .status.nodeDisruptionPolicyStatus.clusterPolicies.units[].actions 594
21.1.55. .status.nodeDisruptionPolicyStatus.clusterPolicies.units[].actions[] 595
21.1.56. .status.nodeDisruptionPolicyStatus.clusterPolicies.units[].actions[].reload 595
21.1.57. .status.nodeDisruptionPolicyStatus.clusterPolicies.units[].actions[].restart 596
21.2. API ENDPOINTS 596
21.2.1. /apis/operator.openshift.io/v1/machineconfigurations 597
21.2.2. /apis/operator.openshift.io/v1/machineconfigurations/{name} 599
21.2.3. /apis/operator.openshift.io/v1/machineconfigurations/{name}/status 602

CHAPTER 22. NETWORK [OPERATOR.OPENSHIFT.IO/V1] 605
22.1. SPECIFICATION 605
22.1.1. .spec 606
22.1.2. .spec.additionalNetworks 610
22.1.3. .spec.additionalNetworks[] 610
22.1.4. .spec.additionalNetworks[].simpleMacvlanConfig 611
22.1.5. .spec.additionalNetworks[].simpleMacvlanConfig.ipamConfig 612
22.1.6. .spec.additionalNetworks[].simpleMacvlanConfig.ipamConfig.staticIPAMConfig 612
22.1.7. .spec.additionalNetworks[].simpleMacvlanConfig.ipamConfig.staticIPAMConfig.addresses 612
22.1.8. .spec.additionalNetworks[].simpleMacvlanConfig.ipamConfig.staticIPAMConfig.addresses[] 613
22.1.9. .spec.additionalNetworks[].simpleMacvlanConfig.ipamConfig.staticIPAMConfig.dns 613
22.1.10. .spec.additionalNetworks[].simpleMacvlanConfig.ipamConfig.staticIPAMConfig.routes 613
22.1.11. .spec.additionalNetworks[].simpleMacvlanConfig.ipamConfig.staticIPAMConfig.routes[] 614
22.1.12. .spec.additionalRoutingCapabilities 614
22.1.13. .spec.clusterNetwork 614
22.1.14. .spec.clusterNetwork[] 615
22.1.15. .spec.defaultNetwork 615
22.1.16. .spec.defaultNetwork.openshiftSDNConfig 615
22.1.17. .spec.defaultNetwork.ovnKubernetesConfig 616
22.1.18. .spec.defaultNetwork.ovnKubernetesConfig.egressIPConfig 618
22.1.19. .spec.defaultNetwork.ovnKubernetesConfig.gatewayConfig 619
22.1.20. .spec.defaultNetwork.ovnKubernetesConfig.gatewayConfig.ipv4 620
22.1.21. .spec.defaultNetwork.ovnKubernetesConfig.gatewayConfig.ipv6 621
22.1.22. .spec.defaultNetwork.ovnKubernetesConfig.hybridOverlayConfig 622

22.1.23. .spec.defaultNetwork.ovnKubernetesConfig.hybridOverlayConfig.hybridClusterNetwork 623
22.1.24. .spec.defaultNetwork.ovnKubernetesConfig.hybridOverlayConfig.hybridClusterNetwork[] 623
22.1.25. .spec.defaultNetwork.ovnKubernetesConfig.ipsecConfig 623
22.1.26. .spec.defaultNetwork.ovnKubernetesConfig.ipsecConfig.full 624
22.1.27. .spec.defaultNetwork.ovnKubernetesConfig.ipv4 625
22.1.28. .spec.defaultNetwork.ovnKubernetesConfig.ipv6 626
22.1.29. .spec.defaultNetwork.ovnKubernetesConfig.policyAuditConfig 627
22.1.30. .spec.exportNetworkFlows 628
22.1.31. .spec.exportNetworkFlows.ipfix 629
22.1.32. .spec.exportNetworkFlows.netFlow 629
22.1.33. .spec.exportNetworkFlows.sFlow 629
22.1.34. .spec.kubeProxyConfig 630
22.1.35. .spec.kubeProxyConfig.proxyArguments 630
22.1.36. .spec.migration 630
22.1.37. .spec.migration.features 631
22.1.38. .spec.migration.mtu 632
22.1.39. .spec.migration.mtu.machine 632
22.1.40. .spec.migration.mtu.network 633
22.1.41. .status 633
22.1.42. .status.conditions 634
22.1.43. .status.conditions[] 634
22.1.44. .status.generations 635
22.1.45. .status.generations[] 635
22.2. API ENDPOINTS 636
22.2.1. /apis/operator.openshift.io/v1/networks 637
22.2.2. /apis/operator.openshift.io/v1/networks/{name} 638
22.2.3. /apis/operator.openshift.io/v1/networks/{name}/status 641

CHAPTER 23. OPENSHIFTAPISERVER [OPERATOR.OPENSHIFT.IO/V1] 645
23.1. SPECIFICATION 645
23.1.1. .spec 646
23.1.2. .status 647
23.1.3. .status.conditions 648
23.1.4. .status.conditions[] 648
23.1.5. .status.generations 649
23.1.6. .status.generations[] 649
23.2. API ENDPOINTS 650
23.2.1. /apis/operator.openshift.io/v1/openshiftapiservers 651
23.2.2. /apis/operator.openshift.io/v1/openshiftapiservers/{name} 652
23.2.3. /apis/operator.openshift.io/v1/openshiftapiservers/{name}/status 655

CHAPTER 24. OPENSHIFTCONTROLLERMANAGER [OPERATOR.OPENSHIFT.IO/V1] 659
24.1. SPECIFICATION 659
24.1.1. .spec 660
24.1.2. .status 661
24.1.3. .status.conditions 662
24.1.4. .status.conditions[] 662
24.1.5. .status.generations 663
24.1.6. .status.generations[] 663
24.2. API ENDPOINTS 664
24.2.1. /apis/operator.openshift.io/v1/openshiftcontrollermanagers 664
24.2.2. /apis/operator.openshift.io/v1/openshiftcontrollermanagers/{name} 666
24.2.3. /apis/operator.openshift.io/v1/openshiftcontrollermanagers/{name}/status 669

CHAPTER 25. OPERATORPKI [NETWORK.OPERATOR.OPENSHIFT.IO/V1] 673
25.1. SPECIFICATION 673
25.1.1. .spec 674
25.1.2. .spec.targetCert 675
25.1.3. .status 675
25.2. API ENDPOINTS 675
25.2.1. /apis/network.operator.openshift.io/v1/operatorpkis 676
25.2.2. /apis/network.operator.openshift.io/v1/namespaces/{namespace}/operatorpkis 676
25.2.3. /apis/network.operator.openshift.io/v1/namespaces/{namespace}/operatorpkis/{name} 678

CHAPTER 26. SERVICECA [OPERATOR.OPENSHIFT.IO/V1] 682
26.1. SPECIFICATION 682
26.1.1. .spec 683
26.1.2. .status 684
26.1.3. .status.conditions 685
26.1.4. .status.conditions[] 685
26.1.5. .status.generations 686
26.1.6. .status.generations[] 686
26.2. API ENDPOINTS 687
26.2.1. /apis/operator.openshift.io/v1/servicecas 687
26.2.2. /apis/operator.openshift.io/v1/servicecas/{name} 689
26.2.3. /apis/operator.openshift.io/v1/servicecas/{name}/status 692

CHAPTER 27. STORAGE [OPERATOR.OPENSHIFT.IO/V1] 696
27.1. SPECIFICATION 696
27.1.1. .spec 697
27.1.2. .status 698
27.1.3. .status.conditions 699
27.1.4. .status.conditions[] 699
27.1.5. .status.generations 700
27.1.6. .status.generations[] 700
27.2. API ENDPOINTS 701
27.2.1. /apis/operator.openshift.io/v1/storages 702
27.2.2. /apis/operator.openshift.io/v1/storages/{name} 703
27.2.3. /apis/operator.openshift.io/v1/storages/{name}/status 706

CHAPTER 1. OPERATOR APIS

1.1. AUTHENTICATION [OPERATOR.OPENSHIFT.IO/V1]

Description

Authentication provides information to configure an operator to manage authentication.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.2. CLOUDCREDENTIAL [OPERATOR.OPENSHIFT.IO/V1]

Description

CloudCredential provides a means to configure an operator to manage CredentialsRequests.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.3. CLUSTERCSIDRIVER [OPERATOR.OPENSHIFT.IO/V1]

Description

ClusterCSIDriver object allows management and configuration of a CSI driver operator installed by default in OpenShift. Name of the object must be name of the CSI driver it operates. See CSIDriverName type for list of allowed values.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.4. CONSOLE [OPERATOR.OPENSHIFT.IO/V1]

Description

Console provides a means to configure an operator to manage the console.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.5. CONFIG [OPERATOR.OPENSHIFT.IO/V1]

Description

Config specifies the behavior of the config operator which is responsible for creating the initial configuration of other components on the cluster. The operator also handles installation, migration or synchronization of cloud configurations for AWS and Azure cloud based clusters
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.6. CONFIG [IMAGEREGISTRY.OPERATOR.OPENSHIFT.IO/V1]

Description

Config is the configuration object for a registry instance managed by the registry operator
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.7. CONFIG [SAMPLES.OPERATOR.OPENSHIFT.IO/V1]

Description

Config contains the configuration and detailed condition status for the Samples Operator.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.8. CSISNAPSHOTCONTROLLER [OPERATOR.OPENSHIFT.IO/V1]

Description

CSISnapshotController provides a means to configure an operator to manage the CSI snapshots. **cluster** is the canonical name.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.9. DNS [OPERATOR.OPENSHIFT.IO/V1]

Description

DNS manages the CoreDNS component to provide a name resolution service for pods and services in the cluster. This supports the DNS-based service discovery specification: https://github.com/kubernetes/dns/blob/master/docs/specification.md More details: https://kubernetes.io/docs/tasks/administer-cluster/coredns Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.10. DNSRECORD [INGRESS.OPERATOR.OPENSHIFT.IO/V1]

Description

DNSRecord is a DNS record managed in the zones defined by dns.config.openshift.io/cluster .spec.publicZone and .spec.privateZone.
Cluster admin manipulation of this resource is not supported. This resource is only for internal communication of OpenShift operators.

If DNSManagementPolicy is "Unmanaged", the operator will not be responsible for managing the DNS records on the cloud provider.

Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.11. ETCD [OPERATOR.OPENSHIFT.IO/V1]

Description

Etcd provides information to configure an operator to manage etcd.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.12. IMAGECONTENTSOURCEPOLICY [OPERATOR.OPENSHIFT.IO/V1ALPHA1]

Description

ImageContentSourcePolicy holds cluster-wide information about how to handle registry mirror rules. When multiple policies are defined, the outcome of the behavior is defined on each field.
Compatibility level 4: No compatibility is provided, the API can change at any point for any reason. These capabilities should not be used by applications needing long term support.

Type

object

1.13. IMAGEPRUNER [IMAGEREGISTRY.OPERATOR.OPENSHIFT.IO/V1]

Description

ImagePruner is the configuration object for an image registry pruner managed by the registry operator.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.14. INGRESSCONTROLLER [OPERATOR.OPENSHIFT.IO/V1]

Description

IngressController describes a managed ingress controller for the cluster. The controller can service OpenShift Route and Kubernetes Ingress resources.
When an IngressController is created, a new ingress controller deployment is created to allow external traffic to reach the services that expose Ingress or Route resources. Updating this resource may lead to disruption for public facing network connections as a new ingress controller revision may be rolled out.

https://kubernetes.io/docs/concepts/services-networking/ingress-controllers

Whenever possible, sensible defaults for the platform are used. See each field for more details.

Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.15. INSIGHTSOPERATOR [OPERATOR.OPENSHIFT.IO/V1]

Description

InsightsOperator holds cluster-wide information about the Insights Operator. Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.16. KUBEAPISERVER [OPERATOR.OPENSHIFT.IO/V1]

Description

KubeAPIServer provides information to configure an operator to manage kube-apiserver.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.17. KUBECONTROLLERMANAGER [OPERATOR.OPENSHIFT.IO/V1]

Description

KubeControllerManager provides information to configure an operator to manage kube-controller-manager.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.18. KUBESCHEDULER [OPERATOR.OPENSHIFT.IO/V1]

Description

KubeScheduler provides information to configure an operator to manage scheduler.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.19. KUBESTORAGEVERSIONMIGRATOR [OPERATOR.OPENSHIFT.IO/V1]

Description

KubeStorageVersionMigrator provides information to configure an operator to manage kube-storage-version-migrator.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.20. MACHINECONFIGURATION [OPERATOR.OPENSHIFT.IO/V1]

Description

MachineConfiguration provides information to configure an operator to manage Machine Configuration.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.21. NETWORK [OPERATOR.OPENSHIFT.IO/V1]

Description

Network describes the cluster's desired network configuration. It is consumed by the cluster-network-operator.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.22. OPENSHIFTAPISERVER [OPERATOR.OPENSHIFT.IO/V1]

Description

OpenShiftAPIServer provides information to configure an operator to manage openshift-apiserver.

Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.23. OPENSHIFTCONTROLLERMANAGER [OPERATOR.OPENSHIFT.IO/V1]

Description

OpenShiftControllerManager provides information to configure an operator to manage openshift-controller-manager.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.24. OPERATORPKI [NETWORK.OPERATOR.OPENSHIFT.IO/V1]

Description

OperatorPKI is a simple certificate authority. It is not intended for external use - rather, it is internal to the network operator. The CNO creates a CA and a certificate signed by that CA. The certificate has both ClientAuth and ServerAuth extended usages enabled.

> More specifically, given an OperatorPKI with <name>, the CNO will manage:

- A Secret called <name>-ca with two data keys:
- tls.key - the private key
- tls.crt - the CA certificate
- A ConfigMap called <name>-ca with a single data key:
- cabundle.crt - the CA certificate(s)
- A Secret called <name>-cert with two data keys:
- tls.key - the private key
- tls.crt - the certificate, signed by the CA

The CA certificate will have a validity of 10 years, rotated after 9. The target certificate will have a validity of 6 months, rotated after 3

The CA certificate will have a CommonName of "<namespace>_<name>-ca@<timestamp>", where <timestamp> is the last rotation time.

Type

object

1.25. SERVICECA [OPERATOR.OPENSHIFT.IO/V1]

Description

ServiceCA provides information to configure an operator to manage the service cert controllers
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.26. STORAGE [OPERATOR.OPENSHIFT.IO/V1]

Description

Storage provides a means to configure an operator to manage the cluster storage operator. **cluster** is the canonical name.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

CHAPTER 2. AUTHENTICATION [OPERATOR.OPENSHIFT.IO/V1]

Description

Authentication provides information to configure an operator to manage authentication. Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

2.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	

Property	Type	Description
status	**object**	

2.1.1. .spec

Description

Type

object

Property	Type	Description
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
managementState	**string**	managementState indicates whether and how the operator should manage the component
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".

Property	Type	Description
unsupportedConfigOverrides	``	unsupportedConfigOverrides overrides the final configuration that was computed by the operator. Red Hat does not support the use of this field. Misuse of this field could lead to unexpected behavior or conflict with other configuration options. Seek guidance from the Red Hat support before using this field. Use of this property blocks cluster upgrades, it must be removed before upgrading your cluster.

2.1.2. .status

Description

Type

object

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status
conditions[]	**object**	OperatorCondition is just the standard condition fields.
generations	**array**	generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.
generations[]	**object**	GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.
latestAvailableRevision	**integer**	latestAvailableRevision is the deploymentID of the most recent deployment
oauthAPIServer	**object**	oauthAPIServer holds status specific only to oauth-apiserver

Property	Type	Description
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with
readyReplicas	**integer**	readyReplicas indicates how many replicas are ready and at the desired state
version	**string**	version is the level this availability applies to

2.1.3. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

2.1.4. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	

Property	Type	Description
reason	**string**	
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

2.1.5. .status.generations

Description

generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Type

array

2.1.6. .status.generations[]

Description

GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.

Type

object

Required

- **group**
- **name**
- **namespace**
- **resource**

Property	Type	Description
group	**string**	group is the group of the thing you're tracking
hash	**string**	hash is an optional field set for resources without generation that are content sensitive like secrets and configmaps

Property	Type	Description
lastGeneration	**integer**	lastGeneration is the last generation of the workload controller involved
name	**string**	name is the name of the thing you're tracking
namespace	**string**	namespace is where the thing you're tracking is
resource	**string**	resource is the resource type of the thing you're tracking

2.1.7. .status.oauthAPIServer

Description

oauthAPIServer holds status specific only to oauth-apiserver

Type

object

Property	Type	Description
latestAvailableRevision	**integer**	latestAvailableRevision is the latest revision used as suffix of revisioned secrets like encryption-config. A new revision causes a new deployment of pods.

2.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/authentications**
  - **DELETE**: delete collection of Authentication
  - **GET**: list objects of kind Authentication
  - **POST**: create an Authentication
- **/apis/operator.openshift.io/v1/authentications/{name}**
  - **DELETE**: delete an Authentication
  - **GET**: read the specified Authentication
  - **PATCH**: partially update the specified Authentication

- **PUT**: replace the specified Authentication

- **/apis/operator.openshift.io/v1/authentications/{name}/status**
  - **GET**: read status of the specified Authentication
  - **PATCH**: partially update status of the specified Authentication
  - **PUT**: replace status of the specified Authentication

2.2.1. /apis/operator.openshift.io/v1/authentications

HTTP method

DELETE

Description

delete collection of Authentication

Table 2.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind Authentication

Table 2.2. HTTP responses

HTTP code	Reponse body
200 - OK	**AuthenticationList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create an Authentication

Table 2.3. Query parameters

Parameter	Type	Description

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 2.4. Body parameters

Parameter	Type	Description
body	**Authentication** schema	

Table 2.5. HTTP responses

HTTP code	Reponse body
200 - OK	**Authentication** schema
201 - Created	**Authentication** schema
202 - Accepted	**Authentication** schema
401 - Unauthorized	Empty

2.2.2. /apis/operator.openshift.io/v1/authentications/{name}

Table 2.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the Authentication

HTTP method

DELETE

Description

delete an Authentication

Table 2.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 2.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified Authentication

Table 2.9. HTTP responses

HTTP code	Reponse body
200 - OK	**Authentication** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified Authentication

Table 2.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 2.11. HTTP responses

HTTP code	Reponse body
200 - OK	**Authentication** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified Authentication

Table 2.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 2.13. Body parameters

Parameter	Type	Description
body	**Authentication** schema	

Table 2.14. HTTP responses

HTTP code	Reponse body
200 - OK	**Authentication** schema
201 - Created	**Authentication** schema
401 - Unauthorized	Empty

2.2.3. /apis/operator.openshift.io/v1/authentications/{name}/status

Table 2.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the Authentication

HTTP method

GET

Description

read status of the specified Authentication

Table 2.16. HTTP responses

HTTP code	Reponse body
200 - OK	**Authentication** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified Authentication

Table 2.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 2.18. HTTP responses

HTTP code	Reponse body
200 - OK	**Authentication** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified Authentication

Table 2.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 2.20. Body parameters

Parameter	Type	Description
body	**Authentication** schema	

Table 2.21. HTTP responses

HTTP code	Reponse body
200 - OK	**Authentication** schema
201 - Created	**Authentication** schema
401 - Unauthorized	Empty

CHAPTER 3. CLOUDCREDENTIAL [OPERATOR.OPENSHIFT.IO/V1]

Description

CloudCredential provides a means to configure an operator to manage CredentialsRequests. Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

3.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	CloudCredentialSpec is the specification of the desired behavior of the cloud-credential-operator.
status	**object**	CloudCredentialStatus defines the observed status of the cloud-credential-operator.

3.1.1. .spec

Description

CloudCredentialSpec is the specification of the desired behavior of the cloud-credential-operator.

Type

object

Property	Type	Description
credentialsMode	**string**	credentialsMode allows informing CCO that it should not attempt to dynamically determine the root cloud credentials capabilities, and it should just run in the specified mode. It also allows putting the operator into "manual" mode if desired. Leaving the field in default mode runs CCO so that the cluster's cloud credentials will be dynamically probed for capabilities (on supported clouds/platforms). Supported modes: AWS/Azure/GCP: "" (Default), "Mint", "Passthrough", "Manual" Others: Do not set value as other platforms only support running in "Passthrough"
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".

Property	Type	Description
managementState	**string**	managementState indicates whether and how the operator should manage the component
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
unsupportedConfigOverrides	``	unsupportedConfigOverrides overrides the final configuration that was computed by the operator. Red Hat does not support the use of this field. Misuse of this field could lead to unexpected behavior or conflict with other configuration options. Seek guidance from the Red Hat support before using this field. Use of this property blocks cluster upgrades, it must be removed before upgrading your cluster.

3.1.2. .status

Description

CloudCredentialStatus defines the observed status of the cloud-credential-operator.

Type

object

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status

Property	Type	Description
conditions[]	**object**	OperatorCondition is just the standard condition fields.
generations	**array**	generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.
generations[]	**object**	GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.
latestAvailableRevision	**integer**	latestAvailableRevision is the deploymentID of the most recent deployment
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with
readyReplicas	**integer**	readyReplicas indicates how many replicas are ready and at the desired state
version	**string**	version is the level this availability applies to

3.1.3. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

3.1.4. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**

- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	
reason	**string**	
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

3.1.5. .status.generations

Description

generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Type

array

3.1.6. .status.generations[]

Description

GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.

Type

object

Required

- **group**
- **name**
- **namespace**
- **resource**

Property	Type	Description
group	**string**	group is the group of the thing you're tracking
hash	**string**	hash is an optional field set for resources without generation that are content sensitive like secrets and configmaps
lastGeneration	**integer**	lastGeneration is the last generation of the workload controller involved
name	**string**	name is the name of the thing you're tracking
namespace	**string**	namespace is where the thing you're tracking is
resource	**string**	resource is the resource type of the thing you're tracking

3.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/cloudcredentials**
  - **DELETE**: delete collection of CloudCredential
  - **GET**: list objects of kind CloudCredential
  - **POST**: create a CloudCredential
- **/apis/operator.openshift.io/v1/cloudcredentials/{name}**
  - **DELETE**: delete a CloudCredential
  - **GET**: read the specified CloudCredential
  - **PATCH**: partially update the specified CloudCredential
  - **PUT**: replace the specified CloudCredential
- **/apis/operator.openshift.io/v1/cloudcredentials/{name}/status**
  - **GET**: read status of the specified CloudCredential
  - **PATCH**: partially update status of the specified CloudCredential
  - **PUT**: replace status of the specified CloudCredential

3.2.1. /apis/operator.openshift.io/v1/cloudcredentials

HTTP method

DELETE

Description

delete collection of CloudCredential

Table 3.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind CloudCredential

Table 3.2. HTTP responses

HTTP code	Reponse body
200 - OK	**CloudCredentialList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a CloudCredential

Table 3.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 3.4. Body parameters

Parameter	Type	Description
body	**CloudCredential** schema	

Table 3.5. HTTP responses

HTTP code	Reponse body
200 - OK	**CloudCredential** schema
201 - Created	**CloudCredential** schema
202 - Accepted	**CloudCredential** schema
401 - Unauthorized	Empty

3.2.2. /apis/operator.openshift.io/v1/cloudcredentials/{name}

Table 3.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the CloudCredential

HTTP method

DELETE

Description

delete a CloudCredential

Table 3.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 3.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified CloudCredential

Table 3.9. HTTP responses

HTTP code	Reponse body
200 - OK	**CloudCredential** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified CloudCredential

Table 3.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 3.11. HTTP responses

HTTP code	Reponse body
200 - OK	**CloudCredential** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified CloudCredential

Table 3.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 3.13. Body parameters

Parameter	Type	Description
body	**CloudCredential** schema	

Table 3.14. HTTP responses

HTTP code	Reponse body
200 - OK	**CloudCredential** schema
201 - Created	**CloudCredential** schema
401 - Unauthorized	Empty

3.2.3. /apis/operator.openshift.io/v1/cloudcredentials/{name}/status

Table 3.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the CloudCredential

HTTP method

GET

Description

read status of the specified CloudCredential

Table 3.16. HTTP responses

HTTP code	Reponse body
200 - OK	**CloudCredential** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified CloudCredential

Table 3.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 3.18. HTTP responses

HTTP code	Reponse body
200 - OK	**CloudCredential** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified CloudCredential

Table 3.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 3.20. Body parameters

Parameter	Type	Description
body	**CloudCredential** schema	

Table 3.21. HTTP responses

HTTP code	Reponse body
200 - OK	**CloudCredential** schema
201 - Created	**CloudCredential** schema
401 - Unauthorized	Empty

CHAPTER 4. CLUSTERCSIDRIVER [OPERATOR.OPENSHIFT.IO/V1]

Description

ClusterCSIDriver object allows management and configuration of a CSI driver operator installed by default in OpenShift. Name of the object must be name of the CSI driver it operates. See CSIDriverName type for list of allowed values.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

4.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	spec holds user settable values for configuration
status	**object**	status holds observed values from the cluster. They may not be overridden.

4.1.1. .spec

Description

spec holds user settable values for configuration

Type

object

Property	Type	Description
driverConfig	**object**	driverConfig can be used to specify platform specific driver configuration. When omitted, this means no opinion and the platform is left to choose reasonable defaults. These defaults are subject to change over time.
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
managementState	**string**	managementState indicates whether and how the operator should manage the component
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator

Property	Type	Description
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
storageClassState	**string**	storageClassState determines if CSI operator should create and manage storage classes. If this field value is empty or Managed - CSI operator will continuously reconcile storage class and create if necessary. If this field value is Unmanaged - CSI operator will not reconcile any previously created storage class. If this field value is Removed - CSI operator will delete the storage class it created previously. When omitted, this means the user has no opinion and the platform chooses a reasonable default, which is subject to change over time. The current default behaviour is Managed.
unsupportedConfigOverrides	``	unsupportedConfigOverrides overrides the final configuration that was computed by the operator. Red Hat does not support the use of this field. Misuse of this field could lead to unexpected behavior or conflict with other configuration options. Seek guidance from the Red Hat support before using this field. Use of this property blocks cluster upgrades, it must be removed before upgrading your cluster.

4.1.2. .spec.driverConfig

Description

driverConfig can be used to specify platform specific driver configuration. When omitted, this means no opinion and the platform is left to choose reasonable defaults. These defaults are subject to change over time.

Type

object

Required

- **driverType**

Property	Type	Description
aws	**object**	aws is used to configure the AWS CSI driver.
azure	**object**	azure is used to configure the Azure CSI driver.
driverType	**string**	driverType indicates type of CSI driver for which the driverConfig is being applied to. Valid values are: AWS, Azure, GCP, IBMCloud, vSphere and omitted. Consumers should treat unknown values as a NO-OP.
gcp	**object**	gcp is used to configure the GCP CSI driver.
ibmcloud	**object**	ibmcloud is used to configure the IBM Cloud CSI driver.
vSphere	**object**	vSphere is used to configure the vsphere CSI driver.

4.1.3. .spec.driverConfig.aws

Description

aws is used to configure the AWS CSI driver.

Type

object

Property	Type	Description
efsVolumeMetrics	**object**	efsVolumeMetrics sets the configuration for collecting metrics from EFS volumes used by the EFS CSI Driver.

Property	Type	Description
kmsKeyARN	**string**	kmsKeyARN sets the cluster default storage class to encrypt volumes with a user-defined KMS key, rather than the default KMS key used by AWS. The value may be either the ARN or Alias ARN of a KMS key.

4.1.4. .spec.driverConfig.aws.efsVolumeMetrics

Description

efsVolumeMetrics sets the configuration for collecting metrics from EFS volumes used by the EFS CSI Driver.

Type

object

Required

- **state**

Property	Type	Description
recursiveWalk	**object**	recursiveWalk provides additional configuration for collecting volume metrics in the AWS EFS CSI Driver when the state is set to RecursiveWalk.
state	**string**	state defines the state of metric collection in the AWS EFS CSI Driver. This field is required and must be set to one of the following values: Disabled or RecursiveWalk. Disabled means no metrics collection will be performed. This is the default value. RecursiveWalk means the AWS EFS CSI Driver will recursively scan volumes to collect metrics. This process may result in high CPU and memory usage, depending on the volume size.

4.1.5. .spec.driverConfig.aws.efsVolumeMetrics.recursiveWalk

Description

recursiveWalk provides additional configuration for collecting volume metrics in the AWS EFS CSI Driver when the state is set to RecursiveWalk.

Type

object

Property	Type	Description
fsRateLimit	**integer**	fsRateLimit defines the rate limit, in goroutines per file system, for processing volume metrics. When omitted, this means no opinion and the platform is left to choose a reasonable default, which is subject to change over time. The current default is 5. The valid range is from 1 to 100 goroutines.
refreshPeriodMinutes	**integer**	refreshPeriodMinutes specifies the frequency, in minutes, at which volume metrics are refreshed. When omitted, this means no opinion and the platform is left to choose a reasonable default, which is subject to change over time. The current default is 240. The valid range is from 1 to 43200 minutes (30 days).

4.1.6. .spec.driverConfig.azure

Description

azure is used to configure the Azure CSI driver.

Type

object

Property	Type	Description
diskEncryptionSet	**object**	diskEncryptionSet sets the cluster default storage class to encrypt volumes with a customer-managed encryption set, rather than the default platform-managed keys.

4.1.7. .spec.driverConfig.azure.diskEncryptionSet

Description

diskEncryptionSet sets the cluster default storage class to encrypt volumes with a customer-managed encryption set, rather than the default platform-managed keys.

Type

object

Required

- **name**
- **resourceGroup**
- **subscriptionID**

Property	Type	Description
name	**string**	name is the name of the disk encryption set that will be set on the default storage class. The value should consist of only alphanumberic characters, underscores (_), hyphens, and be at most 80 characters in length.
resourceGroup	**string**	resourceGroup defines the Azure resource group that contains the disk encryption set. The value should consist of only alphanumberic characters, underscores (_), parentheses, hyphens and periods. The value should not end in a period and be at most 90 characters in length.
subscriptionID	**string**	subscriptionID defines the Azure subscription that contains the disk encryption set. The value should meet the following conditions: 1. It should be a 128-bit number. 2. It should be 36 characters (32 hexadecimal characters and 4 hyphens) long. 3. It should be displayed in five groups separated by hyphens (-). 4. The first group should be 8 characters long. 5. The second, third, and fourth groups should be 4 characters long. 6. The fifth group should be 12 characters long. An Example SubscrionID: f2007bbf-f802-4a47-9336-cf7c6b89b378

4.1.8. .spec.driverConfig.gcp

Description

gcp is used to configure the GCP CSI driver.

Type

object

Property	Type	Description
kmsKey	**object**	kmsKey sets the cluster default storage class to encrypt volumes with customer-supplied encryption keys, rather than the default keys managed by GCP.

4.1.9. .spec.driverConfig.gcp.kmsKey

Description

kmsKey sets the cluster default storage class to encrypt volumes with customer-supplied encryption keys, rather than the default keys managed by GCP.

Type

object

Required

- **keyRing**
- **name**
- **projectID**

Property	Type	Description
keyRing	**string**	keyRing is the name of the KMS Key Ring which the KMS Key belongs to. The value should correspond to an existing KMS key ring and should consist of only alphanumeric characters, hyphens (-) and underscores (_), and be at most 63 characters in length.
location	**string**	location is the GCP location in which the Key Ring exists. The value must match an existing GCP location, or "global". Defaults to global, if not set.

Property	Type	Description
name	**string**	name is the name of the customer-managed encryption key to be used for disk encryption. The value should correspond to an existing KMS key and should consist of only alphanumeric characters, hyphens (-) and underscores (_), and be at most 63 characters in length.
projectID	**string**	projectID is the ID of the Project in which the KMS Key Ring exists. It must be 6 to 30 lowercase letters, digits, or hyphens. It must start with a letter. Trailing hyphens are prohibited.

4.1.10. .spec.driverConfig.ibmcloud

Description

ibmcloud is used to configure the IBM Cloud CSI driver.

Type

object

Required

- **encryptionKeyCRN**

Property	Type	Description
encryptionKeyCRN	**string**	encryptionKeyCRN is the IBM Cloud CRN of the customer-managed root key to use for disk encryption of volumes for the default storage classes.

4.1.11. .spec.driverConfig.vSphere

Description

vSphere is used to configure the vsphere CSI driver.

Type

object

Property	Type	Description
globalMaxSnapshotsPerBlockVolume	**integer**	globalMaxSnapshotsPerBlockVolume is a global configuration parameter that applies to volumes on all kinds of datastores. If omitted, the platform chooses a default, which is subject to change over time, currently that default is 3. Snapshots can not be disabled using this parameter. Increasing number of snapshots above 3 can have negative impact on performance, for more details see: https://kb.vmware.com/s/article/1025279 Volume snapshot documentation: https://docs.vmware.com/en/VMware-vSphere-Container-Storage-Plug-in/3.0/vmware-vsphere-csp-getting-started/GUID-E0B41C69-7EEB-450F-A73D-5FD2FF39E891.html
granularMaxSnapshotsPerBlockVolumeInVSAN	**integer**	granularMaxSnapshotsPerBlockVolumeInVSAN is a granular configuration parameter on vSAN datastore only. It overrides GlobalMaxSnapshotsPerBlockVolume if set, while it falls back to the global constraint if unset. Snapshots for VSAN can not be disabled using this parameter.
granularMaxSnapshotsPerBlockVolumeInVVOL	**integer**	granularMaxSnapshotsPerBlockVolumeInVVOL is a granular configuration parameter on Virtual Volumes datastore only. It overrides GlobalMaxSnapshotsPerBlockVolume if set, while it falls back to the global constraint if unset. Snapshots for VVOL can not be disabled using this parameter.

Property	Type	Description
topologyCategories	**array (string)**	topologyCategories indicates tag categories with which vcenter resources such as hostcluster or datacenter were tagged with. If cluster Infrastructure object has a topology, values specified in Infrastructure object will be used and modifications to topologyCategories will be rejected.

4.1.12. .status

Description

status holds observed values from the cluster. They may not be overridden.

Type

object

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status
conditions[]	**object**	OperatorCondition is just the standard condition fields.
generations	**array**	generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.
generations[]	**object**	GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.
latestAvailableRevision	**integer**	latestAvailableRevision is the deploymentID of the most recent deployment
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with
readyReplicas	**integer**	readyReplicas indicates how many replicas are ready and at the desired state

Property	Type	Description
version	**string**	version is the level this availability applies to

4.1.13. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

4.1.14. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	
reason	**string**	
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

4.1.15. .status.generations

Description

generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Type

array

4.1.16. .status.generations[]

Description

GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.

Type

object

Required

- **group**
- **name**
- **namespace**
- **resource**

Property	Type	Description
group	**string**	group is the group of the thing you're tracking
hash	**string**	hash is an optional field set for resources without generation that are content sensitive like secrets and configmaps
lastGeneration	**integer**	lastGeneration is the last generation of the workload controller involved
name	**string**	name is the name of the thing you're tracking
namespace	**string**	namespace is where the thing you're tracking is
resource	**string**	resource is the resource type of the thing you're tracking

4.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/clustercsidrivers**
  - **DELETE**: delete collection of ClusterCSIDriver
  - **GET**: list objects of kind ClusterCSIDriver
  - **POST**: create a ClusterCSIDriver
- **/apis/operator.openshift.io/v1/clustercsidrivers/{name}**
  - **DELETE**: delete a ClusterCSIDriver
  - **GET**: read the specified ClusterCSIDriver
  - **PATCH**: partially update the specified ClusterCSIDriver
  - **PUT**: replace the specified ClusterCSIDriver
- **/apis/operator.openshift.io/v1/clustercsidrivers/{name}/status**
  - **GET**: read status of the specified ClusterCSIDriver
  - **PATCH**: partially update status of the specified ClusterCSIDriver
  - **PUT**: replace status of the specified ClusterCSIDriver

4.2.1. /apis/operator.openshift.io/v1/clustercsidrivers

HTTP method

DELETE

Description

delete collection of ClusterCSIDriver

Table 4.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind ClusterCSIDriver

Table 4.2. HTTP responses

HTTP code	Reponse body
200 - OK	**ClusterCSIDriverList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a ClusterCSIDriver

Table 4.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 4.4. Body parameters

Parameter	Type	Description
body	**ClusterCSIDriver** schema	

Table 4.5. HTTP responses

HTTP code	Reponse body
200 - OK	**ClusterCSIDriver** schema
201 - Created	**ClusterCSIDriver** schema
202 - Accepted	**ClusterCSIDriver** schema
401 - Unauthorized	Empty

4.2.2. /apis/operator.openshift.io/v1/clustercsidrivers/{name}

Table 4.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the ClusterCSIDriver

HTTP method

DELETE

Description

delete a ClusterCSIDriver

Table 4.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 4.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified ClusterCSIDriver

Table 4.9. HTTP responses

HTTP code	Reponse body
200 - OK	**ClusterCSIDriver** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified ClusterCSIDriver

Table 4.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 4.11. HTTP responses

HTTP code	Reponse body
200 - OK	**ClusterCSIDriver** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified ClusterCSIDriver

Table 4.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 4.13. Body parameters

Parameter	Type	Description
body	**ClusterCSIDriver** schema	

Table 4.14. HTTP responses

HTTP code	Reponse body
200 - OK	**ClusterCSIDriver** schema
201 - Created	**ClusterCSIDriver** schema
401 - Unauthorized	Empty

4.2.3. /apis/operator.openshift.io/v1/clustercsidrivers/{name}/status

Table 4.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the ClusterCSIDriver

HTTP method

GET

Description

read status of the specified ClusterCSIDriver

Table 4.16. HTTP responses

HTTP code	Reponse body
200 - OK	**ClusterCSIDriver** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified ClusterCSIDriver

Table 4.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 4.18. HTTP responses

HTTP code	Reponse body
200 - OK	**ClusterCSIDriver** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified ClusterCSIDriver

Table 4.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 4.20. Body parameters

Parameter	Type	Description
body	**ClusterCSIDriver** schema	

Table 4.21. HTTP responses

HTTP code	Reponse body
200 - OK	**ClusterCSIDriver** schema
201 - Created	**ClusterCSIDriver** schema
401 - Unauthorized	Empty

CHAPTER 5. CONSOLE [OPERATOR.OPENSHIFT.IO/V1]

Description

Console provides a means to configure an operator to manage the console.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

5.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	ConsoleSpec is the specification of the desired behavior of the Console.

Property	Type	Description
status	**object**	ConsoleStatus defines the observed status of the Console.

5.1.1. .spec

Description

ConsoleSpec is the specification of the desired behavior of the Console.

Type

object

Property	Type	Description
customization	**object**	customization is used to optionally provide a small set of customization options to the web console.
ingress	**object**	ingress allows to configure the alternative ingress for the console. This field is intended for clusters without ingress capability, where access to routes is not possible.
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
managementState	**string**	managementState indicates whether and how the operator should manage the component
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator

Property	Type	Description
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
plugins	**array (string)**	plugins defines a list of enabled console plugin names.
providers	**object**	providers contains configuration for using specific service providers.
route	**object**	route contains hostname and secret reference that contains the serving certificate. If a custom route is specified, a new route will be created with the provided hostname, under which console will be available. In case of custom hostname uses the default routing suffix of the cluster, the Secret specification for a serving certificate will not be needed. In case of custom hostname points to an arbitrary domain, manual DNS configurations steps are necessary. The default console route will be maintained to reserve the default hostname for console if the custom route is removed. If not specified, default route will be used. DEPRECATED

Property	Type	Description
unsupportedConfigOverrides	``	unsupportedConfigOverrides overrides the final configuration that was computed by the operator. Red Hat does not support the use of this field. Misuse of this field could lead to unexpected behavior or conflict with other configuration options. Seek guidance from the Red Hat support before using this field. Use of this property blocks cluster upgrades, it must be removed before upgrading your cluster.

5.1.2. .spec.customization

Description

customization is used to optionally provide a small set of customization options to the web console.

Type

object

Property	Type	Description
addPage	**object**	addPage allows customizing actions on the Add page in developer perspective.
brand	**string**	brand is the default branding of the web console which can be overridden by providing the brand field. There is a limited set of specific brand options. This field controls elements of the console such as the logo. Invalid value will prevent a console rollout.
capabilities	**array**	capabilities defines an array of capabilities that can be interacted with in the console UI. Each capability defines a visual state that can be interacted with the console to render in the UI. Available capabilities are LightspeedButton and GettingStartedBanner. Each of the available capabilities may appear only once in the list.

Property	Type	Description
capabilities[]	**object**	Capabilities contains set of UI capabilities and their state in the console UI.
customLogoFile	**object**	customLogoFile replaces the default OpenShift logo in the masthead and about dialog. It is a reference to a Only one of customLogoFile or logos can be set at a time. ConfigMap in the openshift-config namespace. This can be created with a command like 'oc create configmap custom-logo --from-file=/path/to/file -n openshift-config'. Image size must be less than 1 MB due to constraints on the ConfigMap size. The ConfigMap key should include a file extension so that the console serves the file with the correct MIME type. The recommended file format for the logo is SVG, but other file formats are allowed if supported by the browser. Deprecated: Use logos instead.
customProductName	**string**	customProductName is the name that will be displayed in page titles, logo alt text, and the about dialog instead of the normal OpenShift product name.
developerCatalog	**object**	developerCatalog allows to configure the shown developer catalog categories (filters) and types (sub-catalogs).
documentationBaseURL	**string**	documentationBaseURL links to external documentation are shown in various sections of the web console. Providing documentationBaseURL will override the default documentation URL. Invalid value will prevent a console rollout.

Property	Type	Description
logos	**array**	logos is used to replace the OpenShift Masthead and Favicon logos in the console UI with custom logos. logos is an optional field that allows a list of logos. Only one of logos or customLogoFile can be set at a time. If logos is set, customLogoFile must be unset. When specified, there must be at least one entry and no more than 2 entries. Each type must appear only once in the list.
logos[]	**object**	Logo defines a configuration based on theme modes for the console UI logo.
perspectives	**array**	perspectives allows enabling/disabling of perspective(s) that user can see in the Perspective switcher dropdown.
perspectives[]	**object**	Perspective defines a perspective that cluster admins want to show/hide in the perspective switcher dropdown
projectAccess	**object**	projectAccess allows customizing the available list of ClusterRoles in the Developer perspective Project access page which can be used by a project admin to specify roles to other users and restrict access within the project. If set, the list will replace the default ClusterRole options.
quickStarts	**object**	quickStarts allows customization of available ConsoleQuickStart resources in console.

5.1.3. .spec.customization.addPage

Description

addPage allows customizing actions on the Add page in developer perspective.

Type

object

Property	Type	Description
disabledActions	**array (string)**	disabledActions is a list of actions that are not shown to users. Each action in the list is represented by its ID.

5.1.4. .spec.customization.capabilities

Description

capabilities defines an array of capabilities that can be interacted with in the console UI. Each capability defines a visual state that can be interacted with the console to render in the UI. Available capabilities are LightspeedButton and GettingStartedBanner. Each of the available capabilities may appear only once in the list.

Type

array

5.1.5. .spec.customization.capabilities[]

Description

Capabilities contains set of UI capabilities and their state in the console UI.

Type

object

Required

- **name**
- **visibility**

Property	Type	Description
name	**string**	name is the unique name of a capability. Available capabilities are LightspeedButton and GettingStartedBanner.
visibility	**object**	visibility defines the visibility state of the capability.

5.1.6. .spec.customization.capabilities[].visibility

Description

visibility defines the visibility state of the capability.

Type

object

Required

- **state**

Property	Type	Description
state	**string**	state defines if the capability is enabled or disabled in the console UI. Enabling the capability in the console UI is represented by the "Enabled" value. Disabling the capability in the console UI is represented by the "Disabled" value.

5.1.7. .spec.customization.customLogoFile

Description

customLogoFile replaces the default OpenShift logo in the masthead and about dialog. It is a reference to a Only one of customLogoFile or logos can be set at a time. ConfigMap in the openshift-config namespace. This can be created with a command like 'oc create configmap custom-logo --from-file=/path/to/file -n openshift-config'. Image size must be less than 1 MB due to constraints on the ConfigMap size. The ConfigMap key should include a file extension so that the console serves the file with the correct MIME type. The recommended file format for the logo is SVG, but other file formats are allowed if supported by the browser. Deprecated: Use logos instead.

Type

object

Property	Type	Description
key	**string**	key allows pointing to a specific key/value inside of the configmap. This is useful for logical file references.
name	**string**	

5.1.8. .spec.customization.developerCatalog

Description

developerCatalog allows to configure the shown developer catalog categories (filters) and types (sub-catalogs).

Type

object

Property	Type	Description
categories	**array**	categories which are shown in the developer catalog.

Property	Type	Description
categories[]	**object**	DeveloperConsoleCatalogCategory for the developer console catalog.
types	**object**	types allows enabling or disabling of sub-catalog types that user can see in the Developer catalog. When omitted, all the sub-catalog types will be shown.

5.1.9. .spec.customization.developerCatalog.categories

Description

categories which are shown in the developer catalog.

Type

array

5.1.10. .spec.customization.developerCatalog.categories[]

Description

DeveloperConsoleCatalogCategory for the developer console catalog.

Type

object

Required

- **id**
- **label**

Property	Type	Description
id	**string**	id is an identifier used in the URL to enable deep linking in console. ID is required and must have 1-32 URL safe (A-Z, a-z, 0-9, - and _) characters.
label	**string**	label defines a category display label. It is required and must have 1-64 characters.
subcategories	**array**	subcategories defines a list of child categories.

Property	Type	Description
subcategories[]	**object**	DeveloperConsoleCatalogCategoryMeta are the key identifiers of a developer catalog category.
tags	**array (string)**	tags is a list of strings that will match the category. A selected category show all items which has at least one overlapping tag between category and item.

5.1.11. .spec.customization.developerCatalog.categories[].subcategories

Description

subcategories defines a list of child categories.

Type

array

5.1.12. .spec.customization.developerCatalog.categories[].subcategories[]

Description

DeveloperConsoleCatalogCategoryMeta are the key identifiers of a developer catalog category.

Type

object

Required

- **id**
- **label**

Property	Type	Description
id	**string**	id is an identifier used in the URL to enable deep linking in console. ID is required and must have 1-32 URL safe (A-Z, a-z, 0-9, - and _) characters.
label	**string**	label defines a category display label. It is required and must have 1-64 characters.

Property	Type	Description
tags	**array (string)**	tags is a list of strings that will match the category. A selected category show all items which has at least one overlapping tag between category and item.

5.1.13. .spec.customization.developerCatalog.types

Description

types allows enabling or disabling of sub-catalog types that user can see in the Developer catalog. When omitted, all the sub-catalog types will be shown.

Type

object

Required

- **state**

Property	Type	Description
disabled	**array (string)**	disabled is a list of developer catalog types (sub-catalogs IDs) that are not shown to users. Types (sub-catalogs) are added via console plugins, the available types (sub-catalog IDs) are available in the console on the cluster configuration page, or when editing the YAML in the console. Example: "Devfile", "HelmChart", "BuilderImage" If the list is empty or all the available sub-catalog types are added, then the complete developer catalog should be hidden.

Property	Type	Description
enabled	**array (string)**	enabled is a list of developer catalog types (sub-catalogs IDs) that will be shown to users. Types (sub-catalogs) are added via console plugins, the available types (sub-catalog IDs) are available in the console on the cluster configuration page, or when editing the YAML in the console. Example: "Devfile", "HelmChart", "BuilderImage" If the list is non-empty, a new type will not be shown to the user until it is added to list. If the list is empty the complete developer catalog will be shown.
state	**string**	state defines if a list of catalog types should be enabled or disabled.

5.1.14. .spec.customization.logos

Description

logos is used to replace the OpenShift Masthead and Favicon logos in the console UI with custom logos. logos is an optional field that allows a list of logos. Only one of logos or customLogoFile can be set at a time. If logos is set, customLogoFile must be unset. When specified, there must be at least one entry and no more than 2 entries. Each type must appear only once in the list.

Type

array

5.1.15. .spec.customization.logos[]

Description

Logo defines a configuration based on theme modes for the console UI logo.

Type

object

Required

- **themes**
- **type**

Property	Type	Description

Property	Type	Description
themes	**array**	themes specifies the themes for the console UI logo. themes is a required field that allows a list of themes. Each item in the themes list must have a unique mode and a source field. Each mode determines whether the logo is for the dark or light mode of the console UI. If a theme is not specified, the default OpenShift logo will be displayed for that theme. There must be at least one entry and no more than 2 entries.
themes[]	**object**	Theme defines a theme mode for the console UI.
type	**string**	type specifies the type of the logo for the console UI. It determines whether the logo is for the masthead or favicon. type is a required field that allows values of Masthead and Favicon. When set to "Masthead", the logo will be used in the masthead and about modal of the console UI. When set to "Favicon", the logo will be used as the favicon of the console UI.

5.1.16. .spec.customization.logos[].themes

Description

themes specifies the themes for the console UI logo. themes is a required field that allows a list of themes. Each item in the themes list must have a unique mode and a source field. Each mode determines whether the logo is for the dark or light mode of the console UI. If a theme is not specified, the default OpenShift logo will be displayed for that theme. There must be at least one entry and no more than 2 entries.

Type

array

5.1.17. .spec.customization.logos[].themes[]

Description

Theme defines a theme mode for the console UI.

Type

object

Required

- **mode**
- **source**

Property	Type	Description
mode	**string**	mode is used to specify what theme mode a logo will apply to in the console UI. mode is a required field that allows values of Dark and Light. When set to Dark, the logo file referenced in the 'file' field will be used when an end-user of the console UI enables the Dark mode. When set to Light, the logo file referenced in the 'file' field will be used when an end-user of the console UI enables the Light mode.
source	**object**	source is used by the console to locate the specified file containing a custom logo. source is a required field that references a ConfigMap name and key that contains the custom logo file in the openshift-config namespace. You can create it with a command like: - 'oc create configmap custom-logos-config --namespace=openshift-config --from-file=/path/to/file' The ConfigMap key must include the file extension so that the console serves the file with the correct MIME type. The recommended file format for the Masthead and Favicon logos is SVG, but other file formats are allowed if supported by the browser. The logo image size must be less than 1 MB due to constraints on the ConfigMap size. For more information, see the documentation: https://docs.redhat.com/en/documentation/openshift_container_platform/4.19/html/web_console/customizing-web-console#customizing-web-console

5.1.18. .spec.customization.logos[].themes[].source

Description

source is used by the console to locate the specified file containing a custom logo. source is a required field that references a ConfigMap name and key that contains the custom logo file in the openshift-config namespace. You can create it with a command like: - 'oc create configmap custom-logos-config --namespace=openshift-config --from-file=/path/to/file' The ConfigMap key must include the file extension so that the console serves the file with the correct MIME type. The recommended file format for the Masthead and Favicon logos is SVG, but other file formats are allowed if supported by the browser. The logo image size must be less than 1 MB due to constraints on the ConfigMap size. For more information, see the documentation: https://docs.redhat.com/en/documentation/openshift_container_platform/4.19/html/web_console/cu web-console#customizing-web-console

Type

object

Required

- **from**

Property	Type	Description
configMap	**object**	configMap specifies the ConfigMap sourcing details such as the name of the ConfigMap and the key for the file. The ConfigMap must exist in the openshift-config namespace. Required when from is "ConfigMap", and forbidden otherwise.
from	**string**	from is a required field to specify the source type of the file reference. Allowed values are ConfigMap. When set to ConfigMap, the file will be sourced from a ConfigMap in the openshift-config namespace. The configMap field must be set when from is set to ConfigMap.

5.1.19. .spec.customization.logos[].themes[].source.configMap

Description

configMap specifies the ConfigMap sourcing details such as the name of the ConfigMap and the key for the file. The ConfigMap must exist in the openshift-config namespace. Required when from is "ConfigMap", and forbidden otherwise.

Type

object

Required

- **key**
- **name**

Property	Type	Description
key	**string**	key is the logo key inside the referenced ConfigMap. Must consist only of alphanumeric characters, dashes (-), underscores (_), and periods (.). Must be at most 253 characters in length. Must end in a valid file extension. A valid file extension must consist of a period followed by 2 to 5 alpha characters.
name	**string**	name is the name of the ConfigMap. name is a required field. Must consist of lower case alphanumeric characters, '-' or '.', and must start and end with an alphanumeric character. Must be at most 253 characters in length.

5.1.20. .spec.customization.perspectives

Description

perspectives allows enabling/disabling of perspective(s) that user can see in the Perspective switcher dropdown.

Type

array

5.1.21. .spec.customization.perspectives[]

Description

Perspective defines a perspective that cluster admins want to show/hide in the perspective switcher dropdown

Type

object

Required

- **id**
- **visibility**

Property	Type	Description
id	**string**	id defines the id of the perspective. Example: "dev", "admin". The available perspective ids can be found in the code snippet section next to the yaml editor. Incorrect or unknown ids will be ignored.
pinnedResources	**array**	pinnedResources defines the list of default pinned resources that users will see on the perspective navigation if they have not customized these pinned resources themselves. The list of available Kubernetes resources could be read via **kubectl api-resources**. The console will also provide a configuration UI and a YAML snippet that will list the available resources that can be pinned to the navigation. Incorrect or unknown resources will be ignored.
pinnedResources[]	**object**	PinnedResourceReference includes the group, version and type of resource
visibility	**object**	visibility defines the state of perspective along with access review checks if needed for that perspective.

5.1.22. .spec.customization.perspectives[].pinnedResources

Description

pinnedResources defines the list of default pinned resources that users will see on the perspective navigation if they have not customized these pinned resources themselves. The list of available Kubernetes resources could be read via **kubectl api-resources**. The console will also provide a configuration UI and a YAML snippet that will list the available resources that can be pinned to the navigation. Incorrect or unknown resources will be ignored.

Type

array

5.1.23. .spec.customization.perspectives[].pinnedResources[]

Description

PinnedResourceReference includes the group, version and type of resource

Type

object

Required

- **group**
- **resource**
- **version**

Property	Type	Description
group	**string**	group is the API Group of the Resource. Enter empty string for the core group. This value should consist of only lowercase alphanumeric characters, hyphens and periods. Example: "", "apps", "build.openshift.io", etc.
resource	**string**	resource is the type that is being referenced. It is normally the plural form of the resource kind in lowercase. This value should consist of only lowercase alphanumeric characters and hyphens. Example: "deployments", "deploymentconfigs", "pods", etc.
version	**string**	version is the API Version of the Resource. This value should consist of only lowercase alphanumeric characters. Example: "v1", "v1beta1", etc.

5.1.24. .spec.customization.perspectives[].visibility

Description

visibility defines the state of perspective along with access review checks if needed for that perspective.

Type

object

Required

- **state**

Property	Type	Description
accessReview	**object**	accessReview defines required and missing access review checks.
state	**string**	state defines the perspective is enabled or disabled or access review check is required.

5.1.25. .spec.customization.perspectives[].visibility.accessReview

Description

accessReview defines required and missing access review checks.

Type

object

Property	Type	Description
missing	**array**	missing defines a list of permission checks. The perspective will only be shown when at least one check fails. When omitted, the access review is skipped and the perspective will not be shown unless it is required to do so based on the configuration of the required access review list.
missing[]	**object**	ResourceAttributes includes the authorization attributes available for resource requests to the Authorizer interface
required	**array**	required defines a list of permission checks. The perspective will only be shown when all checks are successful. When omitted, the access review is skipped and the perspective will not be shown unless it is required to do so based on the configuration of the missing access review list.
required[]	**object**	ResourceAttributes includes the authorization attributes available for resource requests to the Authorizer interface

5.1.26. .spec.customization.perspectives[].visibility.accessReview.missing

Description

missing defines a list of permission checks. The perspective will only be shown when at least one check fails. When omitted, the access review is skipped and the perspective will not be shown unless it is required to do so based on the configuration of the required access review list.

Type

array

5.1.27. .spec.customization.perspectives[].visibility.accessReview.missing[]

Description

ResourceAttributes includes the authorization attributes available for resource requests to the Authorizer interface

Type

object

Property	Type	Description
fieldSelector	**object**	fieldSelector describes the limitation on access based on field. It can only limit access, not broaden it. This field is alpha-level. To use this field, you must enable the **AuthorizeWithSelectors** feature gate (disabled by default).
group	**string**	Group is the API Group of the Resource. "*" means all.
labelSelector	**object**	labelSelector describes the limitation on access based on labels. It can only limit access, not broaden it. This field is alpha-level. To use this field, you must enable the **AuthorizeWithSelectors** feature gate (disabled by default).
name	**string**	Name is the name of the resource being requested for a "get" or deleted for a "delete". "" (empty) means all.

Property	Type	Description
namespace	**string**	Namespace is the namespace of the action being requested. Currently, there is no distinction between no namespace and all namespaces "" (empty) is defaulted for LocalSubjectAccessReviews "" (empty) is empty for cluster-scoped resources "" (empty) means "all" for namespace scoped resources from a SubjectAccessReview or SelfSubjectAccessReview
resource	**string**	Resource is one of the existing resource types. "*" means all.
subresource	**string**	Subresource is one of the existing resource types. "" means none.
verb	**string**	Verb is a kubernetes resource API verb, like: get, list, watch, create, update, delete, proxy. "*" means all.
version	**string**	Version is the API Version of the Resource. "*" means all.

5.1.28. .spec.customization.perspectives[].visibility.accessReview.missing[].fieldSelect

Description

fieldSelector describes the limitation on access based on field. It can only limit access, not broaden it. This field is alpha-level. To use this field, you must enable the **AuthorizeWithSelectors** feature gate (disabled by default).

Type

object

Property	Type	Description

Property	Type	Description
rawSelector	**string**	rawSelector is the serialization of a field selector that would be included in a query parameter. Webhook implementations are encouraged to ignore rawSelector. The kube-apiserver's *SubjectAccessReview will parse the rawSelector as long as the requirements are not present.
requirements	**array**	requirements is the parsed interpretation of a field selector. All requirements must be met for a resource instance to match the selector. Webhook implementations should handle requirements, but how to handle them is up to the webhook. Since requirements can only limit the request, it is safe to authorize as unlimited request if the requirements are not understood.
requirements[]	**object**	FieldSelectorRequirement is a selector that contains values, a key, and an operator that relates the key and values.

5.1.29. .spec.customization.perspectives[].visibility.accessReview.missing[].fieldSelect

Description

requirements is the parsed interpretation of a field selector. All requirements must be met for a resource instance to match the selector. Webhook implementations should handle requirements, but how to handle them is up to the webhook. Since requirements can only limit the request, it is safe to authorize as unlimited request if the requirements are not understood.

Type

array

5.1.30. .spec.customization.perspectives[].visibility.accessReview.missing[].fieldSelect

Description

FieldSelectorRequirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the field selector key that the requirement applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. The list of operators may grow in the future.
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty.

5.1.31. .spec.customization.perspectives[].visibility.accessReview.missing[].labelSelect

Description

labelSelector describes the limitation on access based on labels. It can only limit access, not broaden it.
This field is alpha-level. To use this field, you must enable the **AuthorizeWithSelectors** feature gate (disabled by default).

Type

object

Property	Type	Description
rawSelector	**string**	rawSelector is the serialization of a field selector that would be included in a query parameter. Webhook implementations are encouraged to ignore rawSelector. The kube-apiserver's *SubjectAccessReview will parse the rawSelector as long as the requirements are not present.

Property	Type	Description
requirements	**array**	requirements is the parsed interpretation of a label selector. All requirements must be met for a resource instance to match the selector. Webhook implementations should handle requirements, but how to handle them is up to the webhook. Since requirements can only limit the request, it is safe to authorize as unlimited request if the requirements are not understood.
requirements[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

5.1.32. .spec.customization.perspectives[].visibility.accessReview.missing[].labelSelect

Description

requirements is the parsed interpretation of a label selector. All requirements must be met for a resource instance to match the selector. Webhook implementations should handle requirements, but how to handle them is up to the webhook. Since requirements can only limit the request, it is safe to authorize as unlimited request if the requirements are not understood.

Type

array

5.1.33. .spec.customization.perspectives[].visibility.accessReview.missing[].labelSelect

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

5.1.34. .spec.customization.perspectives[].visibility.accessReview.required

Description

required defines a list of permission checks. The perspective will only be shown when all checks are successful. When omitted, the access review is skipped and the perspective will not be shown unless it is required to do so based on the configuration of the missing access review list.

Type

array

5.1.35. .spec.customization.perspectives[].visibility.accessReview.required[]

Description

ResourceAttributes includes the authorization attributes available for resource requests to the Authorizer interface

Type

object

Property	Type	Description
fieldSelector	**object**	fieldSelector describes the limitation on access based on field. It can only limit access, not broaden it. This field is alpha-level. To use this field, you must enable the **AuthorizeWithSelectors** feature gate (disabled by default).

Property	Type	Description
group	**string**	Group is the API Group of the Resource. "*" means all.
labelSelector	**object**	labelSelector describes the limitation on access based on labels. It can only limit access, not broaden it. This field is alpha-level. To use this field, you must enable the **AuthorizeWithSelectors** feature gate (disabled by default).
name	**string**	Name is the name of the resource being requested for a "get" or deleted for a "delete". "" (empty) means all.
namespace	**string**	Namespace is the namespace of the action being requested. Currently, there is no distinction between no namespace and all namespaces "" (empty) is defaulted for LocalSubjectAccessReviews "" (empty) is empty for cluster-scoped resources "" (empty) means "all" for namespace scoped resources from a SubjectAccessReview or SelfSubjectAccessReview
resource	**string**	Resource is one of the existing resource types. "*" means all.
subresource	**string**	Subresource is one of the existing resource types. "" means none.
verb	**string**	Verb is a kubernetes resource API verb, like: get, list, watch, create, update, delete, proxy. "*" means all.
version	**string**	Version is the API Version of the Resource. "*" means all.

5.1.36. .spec.customization.perspectives[].visibility.accessReview.required[].fieldSelec

Description

fieldSelector describes the limitation on access based on field. It can only limit access, not broaden it. This field is alpha-level. To use this field, you must enable the **AuthorizeWithSelectors** feature gate (disabled by default).

Type

object

Property	Type	Description
rawSelector	**string**	rawSelector is the serialization of a field selector that would be included in a query parameter. Webhook implementations are encouraged to ignore rawSelector. The kube-apiserver's *SubjectAccessReview will parse the rawSelector as long as the requirements are not present.
requirements	**array**	requirements is the parsed interpretation of a field selector. All requirements must be met for a resource instance to match the selector. Webhook implementations should handle requirements, but how to handle them is up to the webhook. Since requirements can only limit the request, it is safe to authorize as unlimited request if the requirements are not understood.
requirements[]	**object**	FieldSelectorRequirement is a selector that contains values, a key, and an operator that relates the key and values.

5.1.37. .spec.customization.perspectives[].visibility.accessReview.required[].fieldSelec

Description

requirements is the parsed interpretation of a field selector. All requirements must be met for a resource instance to match the selector. Webhook implementations should handle requirements, but how to handle them is up to the webhook. Since requirements can only limit the request, it is safe to authorize as unlimited request if the requirements are not understood.

Type

array

5.1.38. .spec.customization.perspectives[].visibility.accessReview.required[].fieldSelec

Description

FieldSelectorRequirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the field selector key that the requirement applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. The list of operators may grow in the future.
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty.

5.1.39. .spec.customization.perspectives[].visibility.accessReview.required[].labelSelec

Description

labelSelector describes the limitation on access based on labels. It can only limit access, not broaden it.

This field is alpha-level. To use this field, you must enable the **AuthorizeWithSelectors** feature gate (disabled by default).

Type

object

Property	Type	Description

Property	Type	Description
rawSelector	**string**	rawSelector is the serialization of a field selector that would be included in a query parameter. Webhook implementations are encouraged to ignore rawSelector. The kube-apiserver's *SubjectAccessReview will parse the rawSelector as long as the requirements are not present.
requirements	**array**	requirements is the parsed interpretation of a label selector. All requirements must be met for a resource instance to match the selector. Webhook implementations should handle requirements, but how to handle them is up to the webhook. Since requirements can only limit the request, it is safe to authorize as unlimited request if the requirements are not understood.
requirements[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

5.1.40. .spec.customization.perspectives[].visibility.accessReview.required[].labelSelec

Description

requirements is the parsed interpretation of a label selector. All requirements must be met for a resource instance to match the selector. Webhook implementations should handle requirements, but how to handle them is up to the webhook. Since requirements can only limit the request, it is safe to authorize as unlimited request if the requirements are not understood.

Type

array

5.1.41. .spec.customization.perspectives[].visibility.accessReview.required[].labelSelec

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

5.1.42. .spec.customization.projectAccess

Description

projectAccess allows customizing the available list of ClusterRoles in the Developer perspective Project access page which can be used by a project admin to specify roles to other users and restrict access within the project. If set, the list will replace the default ClusterRole options.

Type

object

Property	Type	Description
availableClusterRoles	**array (string)**	availableClusterRoles is the list of ClusterRole names that are assignable to users through the project access tab.

5.1.43. .spec.customization.quickStarts

Description

quickStarts allows customization of available ConsoleQuickStart resources in console.

Type

object

Property	Type	Description
disabled	**array (string)**	disabled is a list of ConsoleQuickStart resource names that are not shown to users.

5.1.44. .spec.ingress

Description

ingress allows to configure the alternative ingress for the console. This field is intended for clusters without ingress capability, where access to routes is not possible.

Type

object

Property	Type	Description
clientDownloadsURL	**string**	clientDownloadsURL is a URL to be used as the address to download client binaries. If not specified, the downloads route hostname will be used. This field is required for clusters without ingress capability, where access to routes is not possible. The console operator will monitor the URL and may go degraded if it's unreachable for an extended period. Must use the HTTPS scheme.
consoleURL	**string**	consoleURL is a URL to be used as the base console address. If not specified, the console route hostname will be used. This field is required for clusters without ingress capability, where access to routes is not possible. Make sure that appropriate ingress is set up at this URL. The console operator will monitor the URL and may go degraded if it's unreachable for an extended period. Must use the HTTPS scheme.

5.1.45. .spec.providers

Description

providers contains configuration for using specific service providers.

Type

object

Property	Type	Description
statuspage	**object**	statuspage contains ID for statuspage.io page that provides status info about.

5.1.46. .spec.providers.statuspage

Description

statuspage contains ID for statuspage.io page that provides status info about.

Type

object

Property	Type	Description
pageID	**string**	pageID is the unique ID assigned by Statuspage for your page. This must be a public page.

5.1.47. .spec.route

Description

route contains hostname and secret reference that contains the serving certificate. If a custom route is specified, a new route will be created with the provided hostname, under which console will be available. In case of custom hostname uses the default routing suffix of the cluster, the Secret specification for a serving certificate will not be needed. In case of custom hostname points to an arbitrary domain, manual DNS configurations steps are necessary. The default console route will be maintained to reserve the default hostname for console if the custom route is removed. If not specified, default route will be used. DEPRECATED

Type

object

Property	Type	Description
hostname	**string**	hostname is the desired custom domain under which console will be available.

Property	Type	Description
secret	**object**	secret points to secret in the openshift-config namespace that contains custom certificate and key and needs to be created manually by the cluster admin. Referenced Secret is required to contain following key value pairs: - "tls.crt" - to specifies custom certificate - "tls.key" - to specifies private key of the custom certificate If the custom hostname uses the default routing suffix of the cluster, the Secret specification for a serving certificate will not be needed.

5.1.48. .spec.route.secret

Description

secret points to secret in the openshift-config namespace that contains custom certificate and key and needs to be created manually by the cluster admin. Referenced Secret is required to contain following key value pairs: - "tls.crt" - to specifies custom certificate - "tls.key" - to specifies private key of the custom certificate If the custom hostname uses the default routing suffix of the cluster, the Secret specification for a serving certificate will not be needed.

Type

object

Required

- **name**

Property	Type	Description
name	**string**	name is the metadata.name of the referenced secret

5.1.49. .status

Description

ConsoleStatus defines the observed status of the Console.

Type

object

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status
conditions[]	**object**	OperatorCondition is just the standard condition fields.
generations	**array**	generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.
generations[]	**object**	GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.
latestAvailableRevision	**integer**	latestAvailableRevision is the deploymentID of the most recent deployment
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with
readyReplicas	**integer**	readyReplicas indicates how many replicas are ready and at the desired state
version	**string**	version is the level this availability applies to

5.1.50. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

5.1.51. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	
reason	**string**	
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

5.1.52. .status.generations

Description

generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Type

array

5.1.53. .status.generations[]

Description

GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.

Type

object

Required

- **group**
- **name**
- **namespace**

- **resource**

Property	Type	Description
group	**string**	group is the group of the thing you're tracking
hash	**string**	hash is an optional field set for resources without generation that are content sensitive like secrets and configmaps
lastGeneration	**integer**	lastGeneration is the last generation of the workload controller involved
name	**string**	name is the name of the thing you're tracking
namespace	**string**	namespace is where the thing you're tracking is
resource	**string**	resource is the resource type of the thing you're tracking

5.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/consoles**
  - **DELETE**: delete collection of Console
  - **GET**: list objects of kind Console
  - **POST**: create a Console
- **/apis/operator.openshift.io/v1/consoles/{name}**
  - **DELETE**: delete a Console
  - **GET**: read the specified Console
  - **PATCH**: partially update the specified Console
  - **PUT**: replace the specified Console
- **/apis/operator.openshift.io/v1/consoles/{name}/status**
  - **GET**: read status of the specified Console
  - **PATCH**: partially update status of the specified Console

- **PUT**: replace status of the specified Console

5.2.1. /apis/operator.openshift.io/v1/consoles

HTTP method

DELETE

Description

delete collection of Console

Table 5.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind Console

Table 5.2. HTTP responses

HTTP code	Reponse body
200 - OK	**ConsoleList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a Console

Table 5.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 5.4. Body parameters

Parameter	Type	Description
body	**Console** schema	

Table 5.5. HTTP responses

HTTP code	Reponse body
200 - OK	**Console** schema
201 - Created	**Console** schema
202 - Accepted	**Console** schema
401 - Unauthorized	Empty

5.2.2. /apis/operator.openshift.io/v1/consoles/{name}

Table 5.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the Console

HTTP method

DELETE

Description

delete a Console

Table 5.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 5.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified Console

Table 5.9. HTTP responses

HTTP code	Reponse body
200 - OK	**Console** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified Console

Table 5.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 5.11. HTTP responses

HTTP code	Reponse body
200 - OK	**Console** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified Console

Table 5.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 5.13. Body parameters

Parameter	Type	Description
body	**Console** schema	

Table 5.14. HTTP responses

HTTP code	Reponse body
200 - OK	**Console** schema
201 - Created	**Console** schema
401 - Unauthorized	Empty

5.2.3. /apis/operator.openshift.io/v1/consoles/{name}/status

Table 5.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the Console

HTTP method

GET

Description

read status of the specified Console

Table 5.16. HTTP responses

HTTP code	Reponse body
200 - OK	**Console** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified Console

Table 5.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 5.18. HTTP responses

HTTP code	Reponse body
200 - OK	**Console** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified Console

Table 5.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 5.20. Body parameters

Parameter	Type	Description
body	**Console** schema	

Table 5.21. HTTP responses

HTTP code	Reponse body
200 - OK	**Console** schema
201 - Created	**Console** schema
401 - Unauthorized	Empty

CHAPTER 6. CONFIG [OPERATOR.OPENSHIFT.IO/V1]

Description

Config specifies the behavior of the config operator which is responsible for creating the initial configuration of other components on the cluster. The operator also handles installation, migration or synchronization of cloud configurations for AWS and Azure cloud based clusters

Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

6.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	spec is the specification of the desired behavior of the Config Operator.
status	**object**	status defines the observed status of the Config Operator.

6.1.1. .spec

Description

spec is the specification of the desired behavior of the Config Operator.

Type

object

Property	Type	Description
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
managementState	**string**	managementState indicates whether and how the operator should manage the component
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator

Property	Type	Description
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
unsupportedConfigOverrides	``	unsupportedConfigOverrides overrides the final configuration that was computed by the operator. Red Hat does not support the use of this field. Misuse of this field could lead to unexpected behavior or conflict with other configuration options. Seek guidance from the Red Hat support before using this field. Use of this property blocks cluster upgrades, it must be removed before upgrading your cluster.

6.1.2. .status

Description

status defines the observed status of the Config Operator.

Type

object

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status
conditions[]	**object**	OperatorCondition is just the standard condition fields.
generations	**array**	generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Property	Type	Description
generations[]	**object**	GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.
latestAvailableRevision	**integer**	latestAvailableRevision is the deploymentID of the most recent deployment
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with
readyReplicas	**integer**	readyReplicas indicates how many replicas are ready and at the desired state
version	**string**	version is the level this availability applies to

6.1.3. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

6.1.4. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	
reason	**string**	
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

6.1.5. .status.generations

Description

generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Type

array

6.1.6. .status.generations[]

Description

GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.

Type

object

Required

- **group**
- **name**
- **namespace**
- **resource**

Property	Type	Description
group	**string**	group is the group of the thing you're tracking
hash	**string**	hash is an optional field set for resources without generation that are content sensitive like secrets and configmaps
lastGeneration	**integer**	lastGeneration is the last generation of the workload controller involved
name	**string**	name is the name of the thing you're tracking
namespace	**string**	namespace is where the thing you're tracking is
resource	**string**	resource is the resource type of the thing you're tracking

6.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/configs**
  - **DELETE**: delete collection of Config
  - **GET**: list objects of kind Config
  - **POST**: create a Config
- **/apis/operator.openshift.io/v1/configs/{name}**
  - **DELETE**: delete a Config
  - **GET**: read the specified Config
  - **PATCH**: partially update the specified Config
  - **PUT**: replace the specified Config
- **/apis/operator.openshift.io/v1/configs/{name}/status**
  - **GET**: read status of the specified Config
  - **PATCH**: partially update status of the specified Config
  - **PUT**: replace status of the specified Config

6.2.1. /apis/operator.openshift.io/v1/configs

HTTP method

DELETE

Description

delete collection of Config

Table 6.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind Config

Table 6.2. HTTP responses

HTTP code	Reponse body
200 - OK	**ConfigList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a Config

Table 6.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 6.4. Body parameters

Parameter	Type	Description
body	**Config** schema	

Table 6.5. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
201 - Created	**Config** schema
202 - Accepted	**Config** schema
401 - Unauthorized	Empty

6.2.2. /apis/operator.openshift.io/v1/configs/{name}

Table 6.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the Config

HTTP method

DELETE

Description

delete a Config

Table 6.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 6.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified Config

Table 6.9. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified Config

Table 6.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 6.11. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified Config

Table 6.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 6.13. Body parameters

Parameter	Type	Description
body	**Config** schema	

Table 6.14. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
201 - Created	**Config** schema
401 - Unauthorized	Empty

6.2.3. /apis/operator.openshift.io/v1/configs/{name}/status

Table 6.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the Config

HTTP method

GET

Description

read status of the specified Config

Table 6.16. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified Config

Table 6.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 6.18. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified Config

Table 6.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 6.20. Body parameters

Parameter	Type	Description
body	**Config** schema	

Table 6.21. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
201 - Created	**Config** schema
401 - Unauthorized	Empty

CHAPTER 7. CONFIG [IMAGEREGISTRY.OPERATOR.OPENSHIFT.IO/V1]

Description

Config is the configuration object for a registry instance managed by the registry operator Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **metadata**
- **spec**

7.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	ImageRegistrySpec defines the specs for the running registry.
status	**object**	ImageRegistryStatus reports image registry operational status.

7.1.1. .spec

Description

ImageRegistrySpec defines the specs for the running registry.

Type

object

Required

- **replicas**

Property	Type	Description
affinity	**object**	affinity is a group of node affinity scheduling rules for the image registry pod(s).
defaultRoute	**boolean**	defaultRoute indicates whether an external facing route for the registry should be created using the default generated hostname.
disableRedirect	**boolean**	disableRedirect controls whether to route all data through the Registry, rather than redirecting to the backend.
httpSecret	**string**	httpSecret is the value needed by the registry to secure uploads, generated by default.

Property	Type	Description
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
logging	**integer**	logging is deprecated, use logLevel instead.
managementState	**string**	managementState indicates whether and how the operator should manage the component
nodeSelector	**object (string)**	nodeSelector defines the node selection constraints for the registry pod.
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
proxy	**object**	proxy defines the proxy to be used when calling master api, upstream registries, etc.

Property	Type	Description
readOnly	**boolean**	readOnly indicates whether the registry instance should reject attempts to push new images or delete existing ones.
replicas	**integer**	replicas determines the number of registry instances to run.
requests	**object**	requests controls how many parallel requests a given registry instance will handle before queuing additional requests.
resources	**object**	resources defines the resource requests+limits for the registry pod.
rolloutStrategy	**string**	rolloutStrategy defines rollout strategy for the image registry deployment.
routes	**array**	routes defines additional external facing routes which should be created for the registry.
routes[]	**object**	ImageRegistryConfigRoute holds information on external route access to image registry.
storage	**object**	storage details for configuring registry storage, e.g. S3 bucket coordinates.
tolerations	**array**	tolerations defines the tolerations for the registry pod.
tolerations[]	**object**	The pod this Toleration is attached to tolerates any taint that matches the triple <key,value,effect> using the matching operator <operator>.
topologySpreadConstraints	**array**	topologySpreadConstraints specify how to spread matching pods among the given topology.

Property	Type	Description
topologySpreadConstraints[]	**object**	TopologySpreadConstraint specifies how to spread matching pods among the given topology.
unsupportedConfigOverrides	``	unsupportedConfigOverrides overrides the final configuration that was computed by the operator. Red Hat does not support the use of this field. Misuse of this field could lead to unexpected behavior or conflict with other configuration options. Seek guidance from the Red Hat support before using this field. Use of this property blocks cluster upgrades, it must be removed before upgrading your cluster.

7.1.2. .spec.affinity

Description

affinity is a group of node affinity scheduling rules for the image registry pod(s).

Type

object

Property	Type	Description
nodeAffinity	**object**	Describes node affinity scheduling rules for the pod.
podAffinity	**object**	Describes pod affinity scheduling rules (e.g. co-locate this pod in the same node, zone, etc. as some other pod(s)).
podAntiAffinity	**object**	Describes pod anti-affinity scheduling rules (e.g. avoid putting this pod in the same node, zone, etc. as some other pod(s)).

7.1.3. .spec.affinity.nodeAffinity

Description

Describes node affinity scheduling rules for the pod.

Type

object

Property	Type	Description
preferredDuringSchedulingIgnoredDuringExecution	**array**	The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node matches the corresponding matchExpressions; the node(s) with the highest sum are the most preferred.
preferredDuringSchedulingIgnoredDuringExecution[]	**object**	An empty preferred scheduling term matches all objects with implicit weight 0 (i.e. it's a no-op). A null preferred scheduling term matches no objects (i.e. is also a no-op).
requiredDuringSchedulingIgnoredDuringExecution	**object**	If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to an update), the system may or may not try to eventually evict the pod from its node.

7.1.4. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution

Description

The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node matches the corresponding matchExpressions; the node(s) with the highest sum are the most preferred.

Type

array

7.1.5. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

An empty preferred scheduling term matches all objects with implicit weight 0 (i.e. it's a no-op). A null preferred scheduling term matches no objects (i.e. is also a no-op).

Type

object

Required

- **preference**
- **weight**

Property	Type	Description
preference	**object**	A node selector term, associated with the corresponding weight.
weight	**integer**	Weight associated with matching the corresponding nodeSelectorTerm, in the range 1-100.

7.1.6. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

A node selector term, associated with the corresponding weight.

Type

object

Property	Type	Description
matchExpressions	**array**	A list of node selector requirements by node's labels.
matchExpressions[]	**object**	A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchFields	**array**	A list of node selector requirements by node's fields.

Property	Type	Description
matchFields[]	**object**	A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

7.1.7. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

A list of node selector requirements by node's labels.

Type

array

7.1.8. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	The label key that the selector applies to.
operator	**string**	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.

Property	Type	Description
values	**array (string)**	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

7.1.9. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

A list of node selector requirements by node's fields.

Type

array

7.1.10. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	The label key that the selector applies to.
operator	**string**	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.

Property	Type	Description
values	**array (string)**	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

7.1.11. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution

Description

If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to an update), the system may or may not try to eventually evict the pod from its node.

Type

object

Required

- **nodeSelectorTerms**

Property	Type	Description
nodeSelectorTerms	**array**	Required. A list of node selector terms. The terms are ORed.
nodeSelectorTerms[]	**object**	A null or empty node selector term matches no objects. The requirements of them are ANDed. The TopologySelectorTerm type implements a subset of the NodeSelectorTerm.

7.1.12. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.n

Description

Required. A list of node selector terms. The terms are ORed.

Type

array

7.1.13. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.n

Description

A null or empty node selector term matches no objects. The requirements of them are ANDed. The TopologySelectorTerm type implements a subset of the NodeSelectorTerm.

Type

object

Property	Type	Description
matchExpressions	**array**	A list of node selector requirements by node's labels.
matchExpressions[]	**object**	A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchFields	**array**	A list of node selector requirements by node's fields.
matchFields[]	**object**	A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

7.1.14. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.n

Description

A list of node selector requirements by node's labels.

Type

array

7.1.15. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.n

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	The label key that the selector applies to.
operator	**string**	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.
values	**array (string)**	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

7.1.16. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.n

Description

A list of node selector requirements by node's fields.

Type

array

7.1.17. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.n

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	The label key that the selector applies to.

Property	Type	Description
operator	**string**	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.
values	**array (string)**	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

7.1.18. .spec.affinity.podAffinity

Description

Describes pod affinity scheduling rules (e.g. co-locate this pod in the same node, zone, etc. as some other pod(s)).

Type

object

Property	Type	Description
preferredDuringSchedulingIgnoredDuringExecution	**array**	The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.

Property	Type	Description
preferredDuringSchedulingIgnoredDuringExecution[]	**object**	The weights of all of the matched WeightedPodAffinityTerm fields are added per-node to find the most preferred node(s)
requiredDuringSchedulingIgnoredDuringExecution	**array**	If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.
requiredDuringSchedulingIgnoredDuringExecution[]	**object**	Defines a set of pods (namely those matching the labelSelector relative to the given namespace(s)) that this pod should be co-located (affinity) or not co-located (anti-affinity) with, where co-located is defined as running on a node whose value of the label with key <topologyKey> matches that of any node on which a pod of the set of pods is running

7.1.19. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution

Description

The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.

Type

array

7.1.20. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

The weights of all of the matched WeightedPodAffinityTerm fields are added per-node to find the most preferred node(s)

Type

object

Required

- **podAffinityTerm**
- **weight**

Property	Type	Description
podAffinityTerm	**object**	Required. A pod affinity term, associated with the corresponding weight.
weight	**integer**	weight associated with matching the corresponding podAffinityTerm, in the range 1-100.

7.1.21. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[].

Description

Required. A pod affinity term, associated with the corresponding weight.

Type

object

Required

- **topologyKey**

Property	Type	Description
labelSelector	**object**	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Property	Type	Description
matchLabelKeys	**array (string)**	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with **labelSelector** as **key in (value)** to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).
mismatchLabelKeys	**array (string)**	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with **labelSelector** as **key notin (value)** to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).

Property	Type	Description
namespaceSelector	**object**	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
namespaces	**array (string)**	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".
topologyKey	**string**	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

7.1.22. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.

Property	Type	Description
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

7.1.23. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

7.1.24. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

7.1.25. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

7.1.26. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

7.1.27. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

7.1.28. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution

Description

If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.

Type

array

7.1.29. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[]

Description

Defines a set of pods (namely those matching the labelSelector relative to the given namespace(s)) that this pod should be co-located (affinity) or not co-located (anti-affinity) with, where co-located is defined as running on a node whose value of the label with key <topologyKey> matches that of any

node on which a pod of the set of pods is running

Type

object

Required

- **topologyKey**

Property	Type	Description
labelSelector	**object**	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.
matchLabelKeys	**array (string)**	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with **labelSelector** as **key in (value)** to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).

Property	Type	Description
mismatchLabelKeys	**array (string)**	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with **labelSelector** as **key notin (value)** to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).
namespaceSelector	**object**	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
namespaces	**array (string)**	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Property	Type	Description
topologyKey	**string**	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

7.1.30. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].l

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

7.1.31. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].la

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

7.1.32. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].l

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

7.1.33. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].n

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.

Property	Type	Description
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

7.1.34. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].l

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

7.1.35. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].l

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

7.1.36. .spec.affinity.podAntiAffinity

Description

Describes pod anti-affinity scheduling rules (e.g. avoid putting this pod in the same node, zone, etc. as some other pod(s)).

Type

object

Property	Type	Description
preferredDuringSchedulingIgnoredDuringExecution	**array**	The scheduler will prefer to schedule pods to nodes that satisfy the anti-affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling anti-affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.
preferredDuringSchedulingIgnoredDuringExecution[]	**object**	The weights of all of the matched WeightedPodAffinityTerm fields are added per-node to find the most preferred node(s)

Property	Type	Description
requiredDuringSchedulingIgnoredDuringExecution	**array**	If the anti-affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the anti-affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.
requiredDuringSchedulingIgnoredDuringExecution[]	**object**	Defines a set of pods (namely those matching the labelSelector relative to the given namespace(s)) that this pod should be co-located (affinity) or not co-located (anti-affinity) with, where co-located is defined as running on a node whose value of the label with key <topologyKey> matches that of any node on which a pod of the set of pods is running

7.1.37. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecuti

Description

The scheduler will prefer to schedule pods to nodes that satisfy the anti-affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling anti-affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.

Type

array

7.1.38. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecuti

Description

The weights of all of the matched WeightedPodAffinityTerm fields are added per-node to find the most preferred node(s)

Type

object

Required

- **podAffinityTerm**
- **weight**

Property	Type	Description
podAffinityTerm	**object**	Required. A pod affinity term, associated with the corresponding weight.
weight	**integer**	weight associated with matching the corresponding podAffinityTerm, in the range 1-100.

7.1.39. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecuti

Description

Required. A pod affinity term, associated with the corresponding weight.

Type

object

Required

- **topologyKey**

Property	Type	Description
labelSelector	**object**	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Property	Type	Description
matchLabelKeys	**array (string)**	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with **labelSelector** as **key in (value)** to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).
mismatchLabelKeys	**array (string)**	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with **labelSelector** as **key notin (value)** to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).

Property	Type	Description
namespaceSelector	**object**	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
namespaces	**array (string)**	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".
topologyKey	**string**	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

7.1.40. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecut

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.

Property	Type	Description
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

7.1.41. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecuti

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

7.1.42. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecuti

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

7.1.43. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecuti

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

7.1.44. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecut

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

7.1.45. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecuti

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

7.1.46. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecutic

Description

If the anti-affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the anti-affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.

Type

array

7.1.47. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecutic

Description

Defines a set of pods (namely those matching the labelSelector relative to the given namespace(s))

that this pod should be co-located (affinity) or not co-located (anti-affinity) with, where co-located is defined as running on a node whose value of the label with key <topologyKey> matches that of any node on which a pod of the set of pods is running

Type

object

Required

- **topologyKey**

Property	Type	Description
labelSelector	**object**	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.
matchLabelKeys	**array (string)**	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with **labelSelector** as **key in (value)** to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).

Property	Type	Description
mismatchLabelKeys	**array (string)**	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with **labelSelector** as **key notin (value)** to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).
namespaceSelector	**object**	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
namespaces	**array (string)**	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Property	Type	Description
topologyKey	**string**	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

7.1.48. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecutic

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

7.1.49. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecutic

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

7.1.50. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecutic

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

7.1.51. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecutio

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.

Property	Type	Description
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

7.1.52. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecutio

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

7.1.53. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecutic

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

7.1.54. .spec.proxy

Description

proxy defines the proxy to be used when calling master api, upstream registries, etc.

Type

object

Property	Type	Description
http	**string**	http defines the proxy to be used by the image registry when accessing HTTP endpoints.
https	**string**	https defines the proxy to be used by the image registry when accessing HTTPS endpoints.
noProxy	**string**	noProxy defines a comma-separated list of host names that shouldn't go through any proxy.

7.1.55. .spec.requests

Description

requests controls how many parallel requests a given registry instance will handle before queuing additional requests.

Type

object

Property	Type	Description
read	**object**	read defines limits for image registry's reads.

Property	Type	Description
write	**object**	write defines limits for image registry's writes.

7.1.56. .spec.requests.read

Description

read defines limits for image registry's reads.

Type

object

Property	Type	Description
maxInQueue	**integer**	maxInQueue sets the maximum queued api requests to the registry.
maxRunning	**integer**	maxRunning sets the maximum in flight api requests to the registry.
maxWaitInQueue	**string**	maxWaitInQueue sets the maximum time a request can wait in the queue before being rejected.

7.1.57. .spec.requests.write

Description

write defines limits for image registry's writes.

Type

object

Property	Type	Description
maxInQueue	**integer**	maxInQueue sets the maximum queued api requests to the registry.
maxRunning	**integer**	maxRunning sets the maximum in flight api requests to the registry.
maxWaitInQueue	**string**	maxWaitInQueue sets the maximum time a request can wait in the queue before being rejected.

7.1.58. .spec.resources

Description

resources defines the resource requests+limits for the registry pod.

Type

object

Property	Type	Description
claims	**array**	Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container. This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.
claims[]	**object**	ResourceClaim references one entry in PodSpec.ResourceClaims.
limits	**integer-or-string**	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/
requests	**integer-or-string**	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

7.1.59. .spec.resources.claims

Description

Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container. This is an alpha field and requires enabling the DynamicResourceAllocation feature gate.

This field is immutable. It can only be set for containers.

Type

array

7.1.60. .spec.resources.claims[]

Description

ResourceClaim references one entry in PodSpec.ResourceClaims.

Type

object

Required

- **name**

Property	Type	Description
name	**string**	Name must match the name of one entry in pod.spec.resourceClaims of the Pod where this field is used. It makes that resource available inside a container.
request	**string**	Request is the name chosen for a request in the referenced claim. If empty, everything from the claim is made available, otherwise only the result of this request.

7.1.61. .spec.routes

Description

routes defines additional external facing routes which should be created for the registry.

Type

array

7.1.62. .spec.routes[]

Description

ImageRegistryConfigRoute holds information on external route access to image registry.

Type

object

Required

- **name**

Property	Type	Description
hostname	**string**	hostname for the route.
name	**string**	name of the route to be created.
secretName	**string**	secretName points to secret containing the certificates to be used by the route.

7.1.63. .spec.storage

Description

storage details for configuring registry storage, e.g. S3 bucket coordinates.

Type

object

Property	Type	Description
azure	**object**	azure represents configuration that uses Azure Blob Storage.
emptyDir	**object**	emptyDir represents ephemeral storage on the pod's host node. WARNING: this storage cannot be used with more than 1 replica and is not suitable for production use. When the pod is removed from a node for any reason, the data in the emptyDir is deleted forever.
gcs	**object**	gcs represents configuration that uses Google Cloud Storage.
ibmcos	**object**	ibmcos represents configuration that uses IBM Cloud Object Storage.
managementState	**string**	managementState indicates if the operator manages the underlying storage unit. If Managed the operator will remove the storage when this operator gets Removed.
oss	**object**	Oss represents configuration that uses Alibaba Cloud Object Storage Service.

Property	Type	Description
pvc	**object**	pvc represents configuration that uses a PersistentVolumeClaim.
s3	**object**	s3 represents configuration that uses Amazon Simple Storage Service.
swift	**object**	swift represents configuration that uses OpenStack Object Storage.

7.1.64. .spec.storage.azure

Description

azure represents configuration that uses Azure Blob Storage.

Type

object

Property	Type	Description
accountName	**string**	accountName defines the account to be used by the registry.
cloudName	**string**	cloudName is the name of the Azure cloud environment to be used by the registry. If empty, the operator will set it based on the infrastructure object.
container	**string**	container defines Azure's container to be used by registry.
networkAccess	**object**	networkAccess defines the network access properties for the storage account. Defaults to type: External.

7.1.65. .spec.storage.azure.networkAccess

Description

networkAccess defines the network access properties for the storage account. Defaults to type: External.

Type

object

Property	Type	Description
internal	**object**	internal defines the vnet and subnet names to configure a private endpoint and connect it to the storage account in order to make it private. when type: Internal and internal is unset, the image registry operator will discover vnet and subnet names, and generate a private endpoint name.
type	**string**	type is the network access level to be used for the storage account. type: Internal means the storage account will be private, type: External means the storage account will be publicly accessible. Internal storage accounts are only exposed within the cluster's vnet. External storage accounts are publicly exposed on the internet. When type: Internal is used, a vnetName, subNetName and privateEndpointName may optionally be specified. If unspecificed, the image registry operator will discover vnet and subnet names, and generate a privateEndpointName. Defaults to "External".

7.1.66. .spec.storage.azure.networkAccess.internal

Description

internal defines the vnet and subnet names to configure a private endpoint and connect it to the storage account in order to make it private. when type: Internal and internal is unset, the image registry operator will discover vnet and subnet names, and generate a private endpoint name.

Type

object

Property	Type	Description

Property	Type	Description
networkResourceGroupName	**string**	networkResourceGroupName is the resource group name where the cluster's vnet and subnet are. When omitted, the registry operator will use the cluster resource group (from in the infrastructure status). If you set a networkResourceGroupName on your install-config.yaml, that value will be used automatically (for clusters configured with publish:Internal). Note that both vnet and subnet must be in the same resource group. It must be between 1 and 90 characters in length and must consist only of alphanumeric characters, hyphens (-), periods (.) and underscores (_), and not end with a period.
privateEndpointName	**string**	privateEndpointName is the name of the private endpoint for the registry. When provided, the registry will use it as the name of the private endpoint it will create for the storage account. When omitted, the registry will generate one. It must be between 2 and 64 characters in length and must consist only of alphanumeric characters, hyphens (-), periods (.) and underscores (_). It must start with an alphanumeric character and end with an alphanumeric character or an underscore.

Property	Type	Description
subnetName	**string**	subnetName is the name of the subnet the registry operates in. When omitted, the registry operator will discover and set this by using the **kubernetes.io_cluster.<cluster-id>** tag in the vnet resource, then using one of listed subnets. Advanced cluster network configurations that use network security groups to protect subnets should ensure the provided subnetName has access to Azure Storage service. It must be between 1 and 80 characters in length and must consist only of alphanumeric characters, hyphens (-), periods (.) and underscores (_).
vnetName	**string**	vnetName is the name of the vnet the registry operates in. When omitted, the registry operator will discover and set this by using the **kubernetes.io_cluster.<cluster-id>** tag in the vnet resource. This tag is set automatically by the installer. Commonly, this will be the same vnet as the cluster. Advanced cluster network configurations should ensure the provided vnetName is the vnet of the nodes where the image registry pods are running from. It must be between 2 and 64 characters in length and must consist only of alphanumeric characters, hyphens (-), periods (.) and underscores (_). It must start with an alphanumeric character and end with an alphanumeric character or an underscore.

7.1.67. .spec.storage.emptyDir

Description

emptyDir represents ephemeral storage on the pod's host node. WARNING: this storage cannot be used with more than 1 replica and is not suitable for production use. When the pod is removed from a node for any reason, the data in the emptyDir is deleted forever.

Type

object

7.1.68. .spec.storage.gcs

Description

gcs represents configuration that uses Google Cloud Storage.

Type

object

Property	Type	Description
bucket	**string**	bucket is the bucket name in which you want to store the registry's data. Optional, will be generated if not provided.
keyID	**string**	keyID is the KMS key ID to use for encryption. Optional, buckets are encrypted by default on GCP. This allows for the use of a custom encryption key.
projectID	**string**	projectID is the Project ID of the GCP project that this bucket should be associated with.
region	**string**	region is the GCS location in which your bucket exists. Optional, will be set based on the installed GCS Region.

7.1.69. .spec.storage.ibmcos

Description

ibmcos represents configuration that uses IBM Cloud Object Storage.

Type

object

Property	Type	Description
bucket	**string**	bucket is the bucket name in which you want to store the registry's data. Optional, will be generated if not provided.

Property	Type	Description
location	**string**	location is the IBM Cloud location in which your bucket exists. Optional, will be set based on the installed IBM Cloud location.
resourceGroupName	**string**	resourceGroupName is the name of the IBM Cloud resource group that this bucket and its service instance is associated with. Optional, will be set based on the installed IBM Cloud resource group.
resourceKeyCRN	**string**	resourceKeyCRN is the CRN of the IBM Cloud resource key that is created for the service instance. Commonly referred as a service credential and must contain HMAC type credentials. Optional, will be computed if not provided.
serviceInstanceCRN	**string**	serviceInstanceCRN is the CRN of the IBM Cloud Object Storage service instance that this bucket is associated with. Optional, will be computed if not provided.

7.1.70. .spec.storage.oss

Description

Oss represents configuration that uses Alibaba Cloud Object Storage Service.

Type

object

Property	Type	Description

Property	Type	Description
bucket	**string**	Bucket is the bucket name in which you want to store the registry's data. About Bucket naming, more details you can look at the [official documentation] (https://www.alibabacloud.com/help/doc-detail/257087.htm) Empty value means no opinion and the platform chooses the a default, which is subject to change over time. Currently the default will be autogenerated in the form of <clusterid>-image-registry-<region>-<random string 27 chars>
encryption	**object**	Encryption specifies whether you would like your data encrypted on the server side. More details, you can look cat the [official documentation] (https://www.alibabacloud.com/help/doc-detail/117914.htm)
endpointAccessibility	**string**	EndpointAccessibility specifies whether the registry use the OSS VPC internal endpoint Empty value means no opinion and the platform chooses the a default, which is subject to change over time. Currently the default is **Internal**.
region	**string**	Region is the Alibaba Cloud Region in which your bucket exists. For a list of regions, you can look at the [official documentation] (https://www.alibabacloud.com/help/doc-detail/31837.html). Empty value means no opinion and the platform chooses the a default, which is subject to change over time. Currently the default will be based on the installed Alibaba Cloud Region.

7.1.71. .spec.storage.oss.encryption

Description

Encryption specifies whether you would like your data encrypted on the server side. More details, you can look cat the [official documentation](https://www.alibabacloud.com/help/doc-detail/117914.htm)

Type

object

Property	Type	Description
kms	**object**	KMS (key management service) is an encryption type that holds the struct for KMS KeyID
method	**string**	Method defines the different encrytion modes available Empty value means no opinion and the platform chooses the a default, which is subject to change over time. Currently the default is **AES256**.

7.1.72. .spec.storage.oss.encryption.kms

Description

KMS (key management service) is an encryption type that holds the struct for KMS KeyID

Type

object

Required

- **keyID**

Property	Type	Description
keyID	**string**	KeyID holds the KMS encryption key ID

7.1.73. .spec.storage.pvc

Description

pvc represents configuration that uses a PersistentVolumeClaim.

Type

object

Property	Type	Description
claim	**string**	claim defines the Persisent Volume Claim's name to be used.

7.1.74. .spec.storage.s3

Description

s3 represents configuration that uses Amazon Simple Storage Service.

Type

object

Property	Type	Description
bucket	**string**	bucket is the bucket name in which you want to store the registry's data. Optional, will be generated if not provided.
chunkSizeMiB	**integer**	chunkSizeMiB defines the size of the multipart upload chunks of the S3 API. The S3 API requires multipart upload chunks to be at least 5MiB. When omitted, this means no opinion and the platform is left to choose a reasonable default, which is subject to change over time. The current default value is 10 MiB. The value is an integer number of MiB. The minimum value is 5 and the maximum value is 5120 (5 GiB).
cloudFront	**object**	cloudFront configures Amazon Cloudfront as the storage middleware in a registry.
encrypt	**boolean**	encrypt specifies whether the registry stores the image in encrypted format or not. Optional, defaults to false.
keyID	**string**	keyID is the KMS key ID to use for encryption. Optional, Encrypt must be true, or this parameter is ignored.
region	**string**	region is the AWS region in which your bucket exists. Optional, will be set based on the installed AWS Region.

Property	Type	Description
regionEndpoint	**string**	regionEndpoint is the endpoint for S3 compatible storage services. It should be a valid URL with scheme, e.g. https://s3.example.com. Optional, defaults based on the Region that is provided.
trustedCA	**object**	trustedCA is a reference to a config map containing a CA bundle. The image registry and its operator use certificates from this bundle to verify S3 server certificates. The namespace for the config map referenced by trustedCA is "openshift-config". The key for the bundle in the config map is "ca-bundle.crt".
virtualHostedStyle	**boolean**	virtualHostedStyle enables using S3 virtual hosted style bucket paths with a custom RegionEndpoint Optional, defaults to false.

7.1.75. .spec.storage.s3.cloudFront

Description

cloudFront configures Amazon Cloudfront as the storage middleware in a registry.

Type

object

Required

- **baseURL**
- **keypairID**
- **privateKey**

Property	Type	Description
baseURL	**string**	baseURL contains the SCHEME://HOST[/PATH] at which Cloudfront is served.

Property	Type	Description
duration	**string**	duration is the duration of the Cloudfront session.
keypairID	**string**	keypairID is key pair ID provided by AWS.
privateKey	**object**	privateKey points to secret containing the private key, provided by AWS.

7.1.76. .spec.storage.s3.cloudFront.privateKey

Description

privateKey points to secret containing the private key, provided by AWS.

Type

object

Required

- **key**

Property	Type	Description
key	**string**	The key of the secret to select from. Must be a valid secret key.
name	**string**	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names
optional	**boolean**	Specify whether the Secret or its key must be defined

7.1.77. .spec.storage.s3.trustedCA

Description

trustedCA is a reference to a config map containing a CA bundle. The image registry and its operator use certificates from this bundle to verify S3 server certificates.

The namespace for the config map referenced by trustedCA is "openshift-config". The key for the bundle in the config map is "ca-bundle.crt".

Type

object

Property	Type	Description
name	**string**	name is the metadata.name of the referenced config map. This field must adhere to standard config map naming restrictions. The name must consist solely of alphanumeric characters, hyphens (-) and periods (.). It has a maximum length of 253 characters. If this field is not specified or is empty string, the default trust bundle will be used.

7.1.78. .spec.storage.swift

Description

swift represents configuration that uses OpenStack Object Storage.

Type

object

Property	Type	Description
authURL	**string**	authURL defines the URL for obtaining an authentication token.
authVersion	**string**	authVersion specifies the OpenStack Auth's version.
container	**string**	container defines the name of Swift container where to store the registry's data.
domain	**string**	domain specifies Openstack's domain name for Identity v3 API.
domainID	**string**	domainID specifies Openstack's domain id for Identity v3 API.
regionName	**string**	regionName defines Openstack's region in which container exists.
tenant	**string**	tenant defines Openstack tenant name to be used by registry.

Property	Type	Description
tenantID	**string**	tenant defines Openstack tenant id to be used by registry.

7.1.79. .spec.tolerations

Description

tolerations defines the tolerations for the registry pod.

Type

array

7.1.80. .spec.tolerations[]

Description

The pod this Toleration is attached to tolerates any taint that matches the triple <key,value,effect> using the matching operator <operator>.

Type

object

Property	Type	Description
effect	**string**	Effect indicates the taint effect to match. Empty means match all taint effects. When specified, allowed values are NoSchedule, PreferNoSchedule and NoExecute.
key	**string**	Key is the taint key that the toleration applies to. Empty means match all taint keys. If the key is empty, operator must be Exists; this combination means to match all values and all keys.
operator	**string**	Operator represents a key's relationship to the value. Valid operators are Exists and Equal. Defaults to Equal. Exists is equivalent to wildcard for value, so that a pod can tolerate all taints of a particular category.

Property	Type	Description
tolerationSeconds	**integer**	TolerationSeconds represents the period of time the toleration (which must be of effect NoExecute, otherwise this field is ignored) tolerates the taint. By default, it is not set, which means tolerate the taint forever (do not evict). Zero and negative values will be treated as 0 (evict immediately) by the system.
value	**string**	Value is the taint value the toleration matches to. If the operator is Exists, the value should be empty, otherwise just a regular string.

7.1.81. .spec.topologySpreadConstraints

Description

topologySpreadConstraints specify how to spread matching pods among the given topology.

Type

array

7.1.82. .spec.topologySpreadConstraints[]

Description

TopologySpreadConstraint specifies how to spread matching pods among the given topology.

Type

object

Required

- **maxSkew**
- **topologyKey**
- **whenUnsatisfiable**

Property	Type	Description
labelSelector	**object**	LabelSelector is used to find matching pods. Pods that match this label selector are counted to determine the number of pods in their corresponding topology domain.

Property	Type	Description
matchLabelKeys	**array (string)**	MatchLabelKeys is a set of pod label keys to select the pods over which spreading will be calculated. The keys are used to lookup values from the incoming pod labels, those key-value labels are ANDed with labelSelector to select the group of existing pods over which spreading will be calculated for the incoming pod. The same key is forbidden to exist in both MatchLabelKeys and LabelSelector. MatchLabelKeys cannot be set when LabelSelector isn't set. Keys that don't exist in the incoming pod labels will be ignored. A null or empty list means only match against labelSelector. This is a beta field and requires the MatchLabelKeysInPodTopologySpread feature gate to be enabled (enabled by default).

Property	Type	Description
maxSkew	**integer**	MaxSkew describes the degree to which pods may be unevenly distributed. When **whenUnsatisfiable=DoNotSchedule**, it is the maximum permitted difference between the number of matching pods in the target topology and the global minimum. The global minimum is the minimum number of matching pods in an eligible domain or zero if the number of eligible domains is less than MinDomains. For example, in a 3-zone cluster, MaxSkew is set to 1, and pods with the same labelSelector spread as 2/2/1: In this case, the global minimum is 1. | zone1 | zone2 | zone3 | | P P | P P | P | - if MaxSkew is 1, incoming pod can only be scheduled to zone3 to become 2/2/2; scheduling it onto zone1(zone2) would make the ActualSkew(3-1) on zone1(zone2) violate MaxSkew(1). - if MaxSkew is 2, incoming pod can be scheduled onto any zone. When **whenUnsatisfiable=ScheduleAnyway**, it is used to give higher precedence to topologies that satisfy it. It's a required field. Default value is 1 and 0 is not allowed.

Property	Type	Description
minDomains	**integer**	MinDomains indicates a minimum number of eligible domains. When the number of eligible domains with matching topology keys is less than minDomains, Pod Topology Spread treats "global minimum" as 0, and then the calculation of Skew is performed. And when the number of eligible domains with matching topology keys equals or greater than minDomains, this value has no effect on scheduling. As a result, when the number of eligible domains is less than minDomains, scheduler won't schedule more than maxSkew Pods to those domains. If value is nil, the constraint behaves as if MinDomains is equal to 1. Valid values are integers greater than 0. When value is not nil, WhenUnsatisfiable must be DoNotSchedule. For example, in a 3-zone cluster, MaxSkew is set to 2, MinDomains is set to 5 and pods with the same labelSelector spread as 2/2/2: | zone1 | zone2 | zone3 | | P P | P P | P P | The number of domains is less than 5(MinDomains), so "global minimum" is treated as 0. In this situation, new pod with the same labelSelector cannot be scheduled, because computed skew will be 3(3 - 0) if new Pod is scheduled to any of the three zones, it will violate MaxSkew.

Property	Type	Description
nodeAffinityPolicy	**string**	NodeAffinityPolicy indicates how we will treat Pod's nodeAffinity/nodeSelector when calculating pod topology spread skew. Options are: - Honor: only nodes matching nodeAffinity/nodeSelector are included in the calculations. - Ignore: nodeAffinity/nodeSelector are ignored. All nodes are included in the calculations. If this value is nil, the behavior is equivalent to the Honor policy. This is a beta-level feature default enabled by the NodeInclusionPolicyInPodTopologySpread feature flag.
nodeTaintsPolicy	**string**	NodeTaintsPolicy indicates how we will treat node taints when calculating pod topology spread skew. Options are: - Honor: nodes without taints, along with tainted nodes for which the incoming pod has a toleration, are included. - Ignore: node taints are ignored. All nodes are included. If this value is nil, the behavior is equivalent to the Ignore policy. This is a beta-level feature default enabled by the NodeInclusionPolicyInPodTopologySpread feature flag.

Property	Type	Description
topologyKey	**string**	TopologyKey is the key of node labels. Nodes that have a label with this key and identical values are considered to be in the same topology. We consider each <key, value> as a "bucket", and try to put balanced number of pods into each bucket. We define a domain as a particular instance of a topology. Also, we define an eligible domain as a domain whose nodes meet the requirements of nodeAffinityPolicy and nodeTaintsPolicy. e.g. If TopologyKey is "kubernetes.io/hostname", each Node is a domain of that topology. And, if TopologyKey is "topology.kubernetes.io/zone", each zone is a domain of that topology. It’s a required field.

Property	Type	Description
whenUnsatisfiable	**string**	WhenUnsatisfiable indicates how to deal with a pod if it doesn't satisfy the spread constraint. - DoNotSchedule (default) tells the scheduler not to schedule it. - ScheduleAnyway tells the scheduler to schedule the pod in any location, but giving higher precedence to topologies that would help reduce the skew. A constraint is considered "Unsatisfiable" for an incoming pod if and only if every possible node assignment for that pod would violate "MaxSkew" on some topology. For example, in a 3-zone cluster, MaxSkew is set to 1, and pods with the same labelSelector spread as 3/1/1: | zone1 | zone2 | zone3 | | P P P | P | P | If WhenUnsatisfiable is set to DoNotSchedule, incoming pod can only be scheduled to zone2(zone3) to become 3/2/1(3/1/2) as ActualSkew(2-1) on zone2(zone3) satisfies MaxSkew(1). In other words, the cluster can still be imbalanced, but scheduler won't make it **more** imbalanced. It's a required field.

7.1.83. .spec.topologySpreadConstraints[].labelSelector

Description

LabelSelector is used to find matching pods. Pods that match this label selector are counted to determine the number of pods in their corresponding topology domain.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Property	Type	Description
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

7.1.84. .spec.topologySpreadConstraints[].labelSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

7.1.85. .spec.topologySpreadConstraints[].labelSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

7.1.86. .status

Description

ImageRegistryStatus reports image registry operational status.

Type

object

Required

- **storage**
- **storageManaged**

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status
conditions[]	**object**	OperatorCondition is just the standard condition fields.
generations	**array**	generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.
generations[]	**object**	GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.
latestAvailableRevision	**integer**	latestAvailableRevision is the deploymentID of the most recent deployment

Property	Type	Description
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with
readyReplicas	**integer**	readyReplicas indicates how many replicas are ready and at the desired state
storage	**object**	storage indicates the current applied storage configuration of the registry.
storageManaged	**boolean**	storageManaged is deprecated, please refer to Storage.managementState
version	**string**	version is the level this availability applies to

7.1.87. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

7.1.88. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	
reason	**string**	
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

7.1.89. .status.generations

Description

generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Type

array

7.1.90. .status.generations[]

Description

GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.

Type

object

Required

- **group**
- **name**
- **namespace**
- **resource**

Property	Type	Description
group	**string**	group is the group of the thing you're tracking
hash	**string**	hash is an optional field set for resources without generation that are content sensitive like secrets and configmaps
lastGeneration	**integer**	lastGeneration is the last generation of the workload controller involved
name	**string**	name is the name of the thing you're tracking
namespace	**string**	namespace is where the thing you're tracking is
resource	**string**	resource is the resource type of the thing you're tracking

7.1.91. .status.storage

Description

storage indicates the current applied storage configuration of the registry.

Type

object

Property	Type	Description
azure	**object**	azure represents configuration that uses Azure Blob Storage.
emptyDir	**object**	emptyDir represents ephemeral storage on the pod's host node. WARNING: this storage cannot be used with more than 1 replica and is not suitable for production use. When the pod is removed from a node for any reason, the data in the emptyDir is deleted forever.
gcs	**object**	gcs represents configuration that uses Google Cloud Storage.

Property	Type	Description
ibmcos	**object**	ibmcos represents configuration that uses IBM Cloud Object Storage.
managementState	**string**	managementState indicates if the operator manages the underlying storage unit. If Managed the operator will remove the storage when this operator gets Removed.
oss	**object**	Oss represents configuration that uses Alibaba Cloud Object Storage Service.
pvc	**object**	pvc represents configuration that uses a PersistentVolumeClaim.
s3	**object**	s3 represents configuration that uses Amazon Simple Storage Service.
swift	**object**	swift represents configuration that uses OpenStack Object Storage.

7.1.92. .status.storage.azure

Description

azure represents configuration that uses Azure Blob Storage.

Type

object

Property	Type	Description
accountName	**string**	accountName defines the account to be used by the registry.
cloudName	**string**	cloudName is the name of the Azure cloud environment to be used by the registry. If empty, the operator will set it based on the infrastructure object.

Property	Type	Description
container	**string**	container defines Azure's container to be used by registry.
networkAccess	**object**	networkAccess defines the network access properties for the storage account. Defaults to type: External.

7.1.93. .status.storage.azure.networkAccess

Description

networkAccess defines the network access properties for the storage account. Defaults to type: External.

Type

object

Property	Type	Description
internal	**object**	internal defines the vnet and subnet names to configure a private endpoint and connect it to the storage account in order to make it private. when type: Internal and internal is unset, the image registry operator will discover vnet and subnet names, and generate a private endpoint name.

Property	Type	Description
type	**string**	type is the network access level to be used for the storage account. type: Internal means the storage account will be private, type: External means the storage account will be publicly accessible. Internal storage accounts are only exposed within the cluster's vnet. External storage accounts are publicly exposed on the internet. When type: Internal is used, a vnetName, subNetName and privateEndpointName may optionally be specified. If unspecificed, the image registry operator will discover vnet and subnet names, and generate a privateEndpointName. Defaults to "External".

7.1.94. .status.storage.azure.networkAccess.internal

Description

internal defines the vnet and subnet names to configure a private endpoint and connect it to the storage account in order to make it private. when type: Internal and internal is unset, the image registry operator will discover vnet and subnet names, and generate a private endpoint name.

Type

object

Property	Type	Description

Property	Type	Description
networkResourceGroupName	**string**	networkResourceGroupName is the resource group name where the cluster's vnet and subnet are. When omitted, the registry operator will use the cluster resource group (from in the infrastructure status). If you set a networkResourceGroupName on your install-config.yaml, that value will be used automatically (for clusters configured with publish:Internal). Note that both vnet and subnet must be in the same resource group. It must be between 1 and 90 characters in length and must consist only of alphanumeric characters, hyphens (-), periods (.) and underscores (_), and not end with a period.
privateEndpointName	**string**	privateEndpointName is the name of the private endpoint for the registry. When provided, the registry will use it as the name of the private endpoint it will create for the storage account. When omitted, the registry will generate one. It must be between 2 and 64 characters in length and must consist only of alphanumeric characters, hyphens (-), periods (.) and underscores (_). It must start with an alphanumeric character and end with an alphanumeric character or an underscore.

Property	Type	Description
subnetName	**string**	subnetName is the name of the subnet the registry operates in. When omitted, the registry operator will discover and set this by using the **kubernetes.io_cluster.<cluster-id>** tag in the vnet resource, then using one of listed subnets. Advanced cluster network configurations that use network security groups to protect subnets should ensure the provided subnetName has access to Azure Storage service. It must be between 1 and 80 characters in length and must consist only of alphanumeric characters, hyphens (-), periods (.) and underscores (_).
vnetName	**string**	vnetName is the name of the vnet the registry operates in. When omitted, the registry operator will discover and set this by using the **kubernetes.io_cluster.<cluster-id>** tag in the vnet resource. This tag is set automatically by the installer. Commonly, this will be the same vnet as the cluster. Advanced cluster network configurations should ensure the provided vnetName is the vnet of the nodes where the image registry pods are running from. It must be between 2 and 64 characters in length and must consist only of alphanumeric characters, hyphens (-), periods (.) and underscores (_). It must start with an alphanumeric character and end with an alphanumeric character or an underscore.

7.1.95. .status.storage.emptyDir

Description

emptyDir represents ephemeral storage on the pod's host node. WARNING: this storage cannot be used with more than 1 replica and is not suitable for production use. When the pod is removed from a node for any reason, the data in the emptyDir is deleted forever.

Type

object

7.1.96. .status.storage.gcs

Description

gcs represents configuration that uses Google Cloud Storage.

Type

object

Property	Type	Description
bucket	**string**	bucket is the bucket name in which you want to store the registry's data. Optional, will be generated if not provided.
keyID	**string**	keyID is the KMS key ID to use for encryption. Optional, buckets are encrypted by default on GCP. This allows for the use of a custom encryption key.
projectID	**string**	projectID is the Project ID of the GCP project that this bucket should be associated with.
region	**string**	region is the GCS location in which your bucket exists. Optional, will be set based on the installed GCS Region.

7.1.97. .status.storage.ibmcos

Description

ibmcos represents configuration that uses IBM Cloud Object Storage.

Type

object

Property	Type	Description
bucket	**string**	bucket is the bucket name in which you want to store the registry's data. Optional, will be generated if not provided.

Property	Type	Description
location	**string**	location is the IBM Cloud location in which your bucket exists. Optional, will be set based on the installed IBM Cloud location.
resourceGroupName	**string**	resourceGroupName is the name of the IBM Cloud resource group that this bucket and its service instance is associated with. Optional, will be set based on the installed IBM Cloud resource group.
resourceKeyCRN	**string**	resourceKeyCRN is the CRN of the IBM Cloud resource key that is created for the service instance. Commonly referred as a service credential and must contain HMAC type credentials. Optional, will be computed if not provided.
serviceInstanceCRN	**string**	serviceInstanceCRN is the CRN of the IBM Cloud Object Storage service instance that this bucket is associated with. Optional, will be computed if not provided.

7.1.98. .status.storage.oss

Description

Oss represents configuration that uses Alibaba Cloud Object Storage Service.

Type

object

Property	Type	Description

Property	Type	Description
bucket	**string**	Bucket is the bucket name in which you want to store the registry's data. About Bucket naming, more details you can look at the [official documentation] (https://www.alibabacloud.com/help/doc-detail/257087.htm) Empty value means no opinion and the platform chooses the a default, which is subject to change over time. Currently the default will be autogenerated in the form of <clusterid>-image-registry-<region>-<random string 27 chars>
encryption	**object**	Encryption specifies whether you would like your data encrypted on the server side. More details, you can look cat the [official documentation] (https://www.alibabacloud.com/help/doc-detail/117914.htm)
endpointAccessibility	**string**	EndpointAccessibility specifies whether the registry use the OSS VPC internal endpoint Empty value means no opinion and the platform chooses the a default, which is subject to change over time. Currently the default is **Internal**.
region	**string**	Region is the Alibaba Cloud Region in which your bucket exists. For a list of regions, you can look at the [official documentation] (https://www.alibabacloud.com/help/doc-detail/31837.html). Empty value means no opinion and the platform chooses the a default, which is subject to change over time. Currently the default will be based on the installed Alibaba Cloud Region.

7.1.99. .status.storage.oss.encryption

Description

Encryption specifies whether you would like your data encrypted on the server side. More details, you can look cat the [official documentation](https://www.alibabacloud.com/help/doc-detail/117914.htm)

Type

object

Property	Type	Description
kms	**object**	KMS (key management service) is an encryption type that holds the struct for KMS KeyID
method	**string**	Method defines the different encrytion modes available Empty value means no opinion and the platform chooses the a default, which is subject to change over time. Currently the default is **AES256**.

7.1.100. .status.storage.oss.encryption.kms

Description

KMS (key management service) is an encryption type that holds the struct for KMS KeyID

Type

object

Required

- **keyID**

Property	Type	Description
keyID	**string**	KeyID holds the KMS encryption key ID

7.1.101. .status.storage.pvc

Description

pvc represents configuration that uses a PersistentVolumeClaim.

Type

object

Property	Type	Description
claim	**string**	claim defines the Persisent Volume Claim's name to be used.

7.1.102. .status.storage.s3

Description

s3 represents configuration that uses Amazon Simple Storage Service.

Type

object

Property	Type	Description
bucket	**string**	bucket is the bucket name in which you want to store the registry's data. Optional, will be generated if not provided.
chunkSizeMiB	**integer**	chunkSizeMiB defines the size of the multipart upload chunks of the S3 API. The S3 API requires multipart upload chunks to be at least 5MiB. When omitted, this means no opinion and the platform is left to choose a reasonable default, which is subject to change over time. The current default value is 10 MiB. The value is an integer number of MiB. The minimum value is 5 and the maximum value is 5120 (5 GiB).
cloudFront	**object**	cloudFront configures Amazon Cloudfront as the storage middleware in a registry.
encrypt	**boolean**	encrypt specifies whether the registry stores the image in encrypted format or not. Optional, defaults to false.
keyID	**string**	keyID is the KMS key ID to use for encryption. Optional, Encrypt must be true, or this parameter is ignored.

Property	Type	Description
region	**string**	region is the AWS region in which your bucket exists. Optional, will be set based on the installed AWS Region.
regionEndpoint	**string**	regionEndpoint is the endpoint for S3 compatible storage services. It should be a valid URL with scheme, e.g. https://s3.example.com. Optional, defaults based on the Region that is provided.
trustedCA	**object**	trustedCA is a reference to a config map containing a CA bundle. The image registry and its operator use certificates from this bundle to verify S3 server certificates. The namespace for the config map referenced by trustedCA is "openshift-config". The key for the bundle in the config map is "ca-bundle.crt".
virtualHostedStyle	**boolean**	virtualHostedStyle enables using S3 virtual hosted style bucket paths with a custom RegionEndpoint Optional, defaults to false.

7.1.103. .status.storage.s3.cloudFront

Description

cloudFront configures Amazon Cloudfront as the storage middleware in a registry.

Type

object

Required

- **baseURL**
- **keypairID**
- **privateKey**

Property	Type	Description
baseURL	**string**	baseURL contains the SCHEME://HOST[/PATH] at which Cloudfront is served.
duration	**string**	duration is the duration of the Cloudfront session.
keypairID	**string**	keypairID is key pair ID provided by AWS.
privateKey	**object**	privateKey points to secret containing the private key, provided by AWS.

7.1.104. .status.storage.s3.cloudFront.privateKey

Description

privateKey points to secret containing the private key, provided by AWS.

Type

object

Required

- **key**

Property	Type	Description
key	**string**	The key of the secret to select from. Must be a valid secret key.
name	**string**	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names
optional	**boolean**	Specify whether the Secret or its key must be defined

7.1.105. .status.storage.s3.trustedCA

Description

trustedCA is a reference to a config map containing a CA bundle. The image registry and its operator use certificates from this bundle to verify S3 server certificates.
The namespace for the config map referenced by trustedCA is "openshift-config". The key for the bundle in the config map is "ca-bundle.crt".

Type

object

Property	Type	Description
name	**string**	name is the metadata.name of the referenced config map. This field must adhere to standard config map naming restrictions. The name must consist solely of alphanumeric characters, hyphens (-) and periods (.). It has a maximum length of 253 characters. If this field is not specified or is empty string, the default trust bundle will be used.

7.1.106. .status.storage.swift

Description

swift represents configuration that uses OpenStack Object Storage.

Type

object

Property	Type	Description
authURL	**string**	authURL defines the URL for obtaining an authentication token.
authVersion	**string**	authVersion specifies the OpenStack Auth's version.
container	**string**	container defines the name of Swift container where to store the registry's data.
domain	**string**	domain specifies Openstack's domain name for Identity v3 API.
domainID	**string**	domainID specifies Openstack's domain id for Identity v3 API.
regionName	**string**	regionName defines Openstack's region in which container exists.

Property	Type	Description
tenant	**string**	tenant defines Openstack tenant name to be used by registry.
tenantID	**string**	tenant defines Openstack tenant id to be used by registry.

7.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/imageregistry.operator.openshift.io/v1/configs**
  - **DELETE**: delete collection of Config
  - **GET**: list objects of kind Config
  - **POST**: create a Config
- **/apis/imageregistry.operator.openshift.io/v1/configs/{name}**
  - **DELETE**: delete a Config
  - **GET**: read the specified Config
  - **PATCH**: partially update the specified Config
  - **PUT**: replace the specified Config
- **/apis/imageregistry.operator.openshift.io/v1/configs/{name}/status**
  - **GET**: read status of the specified Config
  - **PATCH**: partially update status of the specified Config
  - **PUT**: replace status of the specified Config

7.2.1. /apis/imageregistry.operator.openshift.io/v1/configs

HTTP method

DELETE

Description

delete collection of Config

Table 7.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema

HTTP code	Reponse body
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind Config

Table 7.2. HTTP responses

HTTP code	Reponse body
200 - OK	**ConfigList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a Config

Table 7.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 7.4. Body parameters

Parameter	Type	Description
body	**Config** schema	

Table 7.5. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
201 - Created	**Config** schema
202 - Accepted	**Config** schema
401 - Unauthorized	Empty

7.2.2. /apis/imageregistry.operator.openshift.io/v1/configs/{name}

Table 7.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the Config

HTTP method

DELETE

Description

delete a Config

Table 7.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 7.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified Config

Table 7.9. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified Config

Table 7.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 7.11. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified Config

Table 7.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 7.13. Body parameters

Parameter	Type	Description
body	**Config** schema	

Table 7.14. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
201 - Created	**Config** schema
401 - Unauthorized	Empty

7.2.3. /apis/imageregistry.operator.openshift.io/v1/configs/{name}/status

Table 7.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the Config

HTTP method

GET

Description

read status of the specified Config

Table 7.16. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified Config

Table 7.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 7.18. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified Config

Table 7.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 7.20. Body parameters

Parameter	Type	Description
body	**Config** schema	

Table 7.21. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
201 - Created	**Config** schema
401 - Unauthorized	Empty

CHAPTER 8. CONFIG [SAMPLES.OPERATOR.OPENSHIFT.IO/V1]

Description

Config contains the configuration and detailed condition status for the Samples Operator. Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **metadata**
- **spec**

8.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	ConfigSpec contains the desired configuration and state for the Samples Operator, controlling various behavior around the imagestreams and templates it creates/updates in the openshift namespace.
status	**object**	ConfigStatus contains the actual configuration in effect, as well as various details that describe the state of the Samples Operator.

8.1.1. .spec

Description

ConfigSpec contains the desired configuration and state for the Samples Operator, controlling various behavior around the imagestreams and templates it creates/updates in the openshift namespace.

Type

object

Property	Type	Description
architectures	**array (string)**	architectures determine which hardware architecture(s) to install, where x86_64, ppc64le, and s390x are the only supported choices currently.

Property	Type	Description
managementState	**string**	managementState is top level on/off type of switch for all operators. When "Managed", this operator processes config and manipulates the samples accordingly. When "Unmanaged", this operator ignores any updates to the resources it watches. When "Removed", it reacts that same wasy as it does if the Config object is deleted, meaning any ImageStreams or Templates it manages (i.e. it honors the skipped lists) and the registry secret are deleted, along with the ConfigMap in the operator's namespace that represents the last config used to manipulate the samples,
samplesRegistry	**string**	samplesRegistry allows for the specification of which registry is accessed by the ImageStreams for their image content. Defaults on the content in https://github.com/openshift/library that are pulled into this github repository, but based on our pulling only ocp content it typically defaults to registry.redhat.io.
skippedImagestreams	**array (string)**	skippedImagestreams specifies names of image streams that should NOT be created/updated. Admins can use this to allow them to delete content they don't want. They will still have to manually delete the content but the operator will not recreate(or update) anything listed here.

Property	Type	Description
skippedTemplates	**array (string)**	skippedTemplates specifies names of templates that should NOT be created/updated. Admins can use this to allow them to delete content they don't want. They will still have to manually delete the content but the operator will not recreate(or update) anything listed here.

8.1.2. .status

Description

ConfigStatus contains the actual configuration in effect, as well as various details that describe the state of the Samples Operator.

Type

object

Property	Type	Description
architectures	**array (string)**	architectures determine which hardware architecture(s) to install, where x86_64 and ppc64le are the supported choices.
conditions	**array**	conditions represents the available maintenance status of the sample imagestreams and templates.
conditions[]	**object**	ConfigCondition captures various conditions of the Config as entries are processed.
managementState	**string**	managementState reflects the current operational status of the on/off switch for the operator. This operator compares the ManagementState as part of determining that we are turning the operator back on (i.e. "Managed") when it was previously "Unmanaged".

Property	Type	Description
samplesRegistry	**string**	samplesRegistry allows for the specification of which registry is accessed by the ImageStreams for their image content. Defaults on the content in https://github.com/openshift/library that are pulled into this github repository, but based on our pulling only ocp content it typically defaults to registry.redhat.io.
skippedImagestreams	**array (string)**	skippedImagestreams specifies names of image streams that should NOT be created/updated. Admins can use this to allow them to delete content they don't want. They will still have to manually delete the content but the operator will not recreate(or update) anything listed here.
skippedTemplates	**array (string)**	skippedTemplates specifies names of templates that should NOT be created/updated. Admins can use this to allow them to delete content they don't want. They will still have to manually delete the content but the operator will not recreate(or update) anything listed here.
version	**string**	version is the value of the operator's payload based version indicator when it was last successfully processed

8.1.3. .status.conditions

Description

conditions represents the available maintenance status of the sample imagestreams and templates.

Type

array

8.1.4. .status.conditions[]

Description

ConfigCondition captures various conditions of the Config as entries are processed.

Type

object

Required

- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another.
lastUpdateTime	**string**	lastUpdateTime is the last time this condition was updated.
message	**string**	message is a human readable message indicating details about the transition.
reason	**string**	reason is what caused the condition's last transition.
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition.

8.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/samples.operator.openshift.io/v1/configs**
  - **DELETE**: delete collection of Config
  - **GET**: list objects of kind Config
  - **POST**: create a Config
- **/apis/samples.operator.openshift.io/v1/configs/{name}**
  - **DELETE**: delete a Config
  - **GET**: read the specified Config
  - **PATCH**: partially update the specified Config
  - **PUT**: replace the specified Config
- **/apis/samples.operator.openshift.io/v1/configs/{name}/status**

- **GET**: read status of the specified Config
- **PATCH**: partially update status of the specified Config
- **PUT**: replace status of the specified Config

8.2.1. /apis/samples.operator.openshift.io/v1/configs

HTTP method

DELETE

Description

delete collection of Config

Table 8.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind Config

Table 8.2. HTTP responses

HTTP code	Reponse body
200 - OK	**ConfigList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a Config

Table 8.3. Query parameters

Parameter	Type	Description

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 8.4. Body parameters

Parameter	Type	Description
body	**Config** schema	

Table 8.5. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
201 - Created	**Config** schema
202 - Accepted	**Config** schema
401 - Unauthorized	Empty

8.2.2. /apis/samples.operator.openshift.io/v1/configs/{name}

Table 8.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the Config

HTTP method

DELETE

Description

delete a Config

Table 8.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 8.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified Config

Table 8.9. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified Config

Table 8.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 8.11. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified Config

Table 8.12. Query parameters

Parameter	Type	Description

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 8.13. Body parameters

Parameter	Type	Description
body	**Config** schema	

Table 8.14. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
201 - Created	**Config** schema
401 - Unauthorized	Empty

8.2.3. /apis/samples.operator.openshift.io/v1/configs/{name}/status

Table 8.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the Config

HTTP method

GET

Description

read status of the specified Config

Table 8.16. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified Config

Table 8.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 8.18. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified Config

Table 8.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 8.20. Body parameters

Parameter	Type	Description
body	**Config** schema	

Table 8.21. HTTP responses

HTTP code	Reponse body
200 - OK	**Config** schema
201 - Created	**Config** schema
401 - Unauthorized	Empty

CHAPTER 9. CSISNAPSHOTCONTROLLER [OPERATOR.OPENSHIFT.IO/V1]

Description

CSISnapshotController provides a means to configure an operator to manage the CSI snapshots. **cluster** is the canonical name.

Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

9.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	spec holds user settable values for configuration
status	**object**	status holds observed values from the cluster. They may not be overridden.

9.1.1. .spec

Description

spec holds user settable values for configuration

Type

object

Property	Type	Description
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
managementState	**string**	managementState indicates whether and how the operator should manage the component
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator

Property	Type	Description
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
unsupportedConfigOverrides	``	unsupportedConfigOverrides overrides the final configuration that was computed by the operator. Red Hat does not support the use of this field. Misuse of this field could lead to unexpected behavior or conflict with other configuration options. Seek guidance from the Red Hat support before using this field. Use of this property blocks cluster upgrades, it must be removed before upgrading your cluster.

9.1.2. .status

Description

status holds observed values from the cluster. They may not be overridden.

Type

object

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status
conditions[]	**object**	OperatorCondition is just the standard condition fields.
generations	**array**	generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Property	Type	Description
generations[]	**object**	GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.
latestAvailableRevision	**integer**	latestAvailableRevision is the deploymentID of the most recent deployment
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with
readyReplicas	**integer**	readyReplicas indicates how many replicas are ready and at the desired state
version	**string**	version is the level this availability applies to

9.1.3. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

9.1.4. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	
reason	**string**	
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

9.1.5. .status.generations

Description

generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Type

array

9.1.6. .status.generations[]

Description

GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.

Type

object

Required

- **group**
- **name**
- **namespace**
- **resource**

Property	Type	Description
group	**string**	group is the group of the thing you're tracking
hash	**string**	hash is an optional field set for resources without generation that are content sensitive like secrets and configmaps
lastGeneration	**integer**	lastGeneration is the last generation of the workload controller involved
name	**string**	name is the name of the thing you're tracking
namespace	**string**	namespace is where the thing you're tracking is
resource	**string**	resource is the resource type of the thing you're tracking

9.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/csisnapshotcontrollers**
  - **DELETE**: delete collection of CSISnapshotController
  - **GET**: list objects of kind CSISnapshotController
  - **POST**: create a CSISnapshotController
- **/apis/operator.openshift.io/v1/csisnapshotcontrollers/{name}**
  - **DELETE**: delete a CSISnapshotController
  - **GET**: read the specified CSISnapshotController
  - **PATCH**: partially update the specified CSISnapshotController
  - **PUT**: replace the specified CSISnapshotController
- **/apis/operator.openshift.io/v1/csisnapshotcontrollers/{name}/status**
  - **GET**: read status of the specified CSISnapshotController
  - **PATCH**: partially update status of the specified CSISnapshotController

- **PUT**: replace status of the specified CSISnapshotController

9.2.1. /apis/operator.openshift.io/v1/csisnapshotcontrollers

HTTP method

DELETE

Description

delete collection of CSISnapshotController

Table 9.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind CSISnapshotController

Table 9.2. HTTP responses

HTTP code	Reponse body
200 - OK	**CSISnapshotControllerList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a CSISnapshotController

Table 9.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 9.4. Body parameters

Parameter	Type	Description
body	**CSISnapshotController** schema	

Table 9.5. HTTP responses

HTTP code	Reponse body
200 - OK	**CSISnapshotController** schema
201 - Created	**CSISnapshotController** schema
202 - Accepted	**CSISnapshotController** schema
401 - Unauthorized	Empty

9.2.2. /apis/operator.openshift.io/v1/csisnapshotcontrollers/{name}

Table 9.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the CSISnapshotController

HTTP method

DELETE

Description

delete a CSISnapshotController

Table 9.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 9.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified CSISnapshotController

Table 9.9. HTTP responses

HTTP code	Reponse body
200 - OK	**CSISnapshotController** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified CSISnapshotController

Table 9.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 9.11. HTTP responses

HTTP code	Reponse body
200 - OK	**CSISnapshotController** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified CSISnapshotController

Table 9.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 9.13. Body parameters

Parameter	Type	Description
body	**CSISnapshotController** schema	

Table 9.14. HTTP responses

HTTP code	Reponse body
200 - OK	**CSISnapshotController** schema
201 - Created	**CSISnapshotController** schema
401 - Unauthorized	Empty

9.2.3. /apis/operator.openshift.io/v1/csisnapshotcontrollers/{name}/status

Table 9.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the CSISnapshotController

HTTP method

GET

Description

read status of the specified CSISnapshotController

Table 9.16. HTTP responses

HTTP code	Reponse body
200 - OK	**CSISnapshotController** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified CSISnapshotController

Table 9.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 9.18. HTTP responses

HTTP code	Reponse body
200 - OK	**CSISnapshotController** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified CSISnapshotController

Table 9.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 9.20. Body parameters

Parameter	Type	Description
body	**CSISnapshotController** schema	

Table 9.21. HTTP responses

HTTP code	Reponse body
200 - OK	**CSISnapshotController** schema
201 - Created	**CSISnapshotController** schema
401 - Unauthorized	Empty

CHAPTER 10. DNS [OPERATOR.OPENSHIFT.IO/V1]

Description

DNS manages the CoreDNS component to provide a name resolution service for pods and services in the cluster. This supports the DNS-based service discovery specification: https://github.com/kubernetes/dns/blob/master/docs/specification.md More details: https://kubernetes.io/docs/tasks/administer-cluster/coredns Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

10.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	spec is the specification of the desired behavior of the DNS.
status	**object**	status is the most recently observed status of the DNS.

10.1.1. .spec

Description

spec is the specification of the desired behavior of the DNS.

Type

object

Property	Type	Description
cache	**object**	cache describes the caching configuration that applies to all server blocks listed in the Corefile. This field allows a cluster admin to optionally configure: * positiveTTL which is a duration for which positive responses should be cached. * negativeTTL which is a duration for which negative responses should be cached. If this is not configured, OpenShift will configure positive and negative caching with a default value that is subject to change. At the time of writing, the default positiveTTL is 900 seconds and the default negativeTTL is 30 seconds or as noted in the respective Corefile for your version of OpenShift.
logLevel	**string**	logLevel describes the desired logging verbosity for CoreDNS. Any one of the following values may be specified: * Normal logs errors from upstream resolvers. * Debug logs errors, NXDOMAIN responses, and NODATA responses. * Trace logs errors and all responses. Setting logLevel: Trace will produce extremely verbose logs. Valid values are: "Normal", "Debug", "Trace". Defaults to "Normal".
managementState	**string**	managementState indicates whether the DNS operator should manage cluster DNS

Property	Type	Description
nodePlacement	**object**	nodePlacement provides explicit control over the scheduling of DNS pods. Generally, it is useful to run a DNS pod on every node so that DNS queries are always handled by a local DNS pod instead of going over the network to a DNS pod on another node. However, security policies may require restricting the placement of DNS pods to specific nodes. For example, if a security policy prohibits pods on arbitrary nodes from communicating with the API, a node selector can be specified to restrict DNS pods to nodes that are permitted to communicate with the API. Conversely, if running DNS pods on nodes with a particular taint is desired, a toleration can be specified for that taint. If unset, defaults are used. See nodePlacement for more details.
operatorLogLevel	**string**	operatorLogLevel controls the logging level of the DNS Operator. Valid values are: "Normal", "Debug", "Trace". Defaults to "Normal". setting operatorLogLevel: Trace will produce extremely verbose logs.
servers	**array**	servers is a list of DNS resolvers that provide name query delegation for one or more subdomains outside the scope of the cluster domain. If servers consists of more than one Server, longest suffix match will be used to determine the Server. For example, if there are two Servers, one for "foo.com" and another for "a.foo.com", and the name query is for "www.a.foo.com", it will be routed to the Server with Zone "a.foo.com". If this field is nil, no servers are created.

Property	Type	Description
servers[]	**object**	Server defines the schema for a server that runs per instance of CoreDNS.
upstreamResolvers	**object**	upstreamResolvers defines a schema for configuring CoreDNS to proxy DNS messages to upstream resolvers for the case of the default (".") server If this field is not specified, the upstream used will default to /etc/resolv.conf, with policy "sequential"

10.1.2. .spec.cache

Description

cache describes the caching configuration that applies to all server blocks listed in the Corefile. This field allows a cluster admin to optionally configure: * positiveTTL which is a duration for which positive responses should be cached. * negativeTTL which is a duration for which negative responses should be cached. If this is not configured, OpenShift will configure positive and negative caching with a default value that is subject to change. At the time of writing, the default positiveTTL is 900 seconds and the default negativeTTL is 30 seconds or as noted in the respective Corefile for your version of OpenShift.

Type

object

Property	Type	Description

Property	Type	Description
negativeTTL	**string**	negativeTTL is optional and specifies the amount of time that a negative response should be cached. If configured, it must be a value of 1s (1 second) or greater up to a theoretical maximum of several years. This field expects an unsigned duration string of decimal numbers, each with optional fraction and a unit suffix, e.g. "100s", "1m30s", "12h30m10s". Values that are fractions of a second are rounded down to the nearest second. If the configured value is less than 1s, the default value will be used. If not configured, the value will be 0s and OpenShift will use a default value of 30 seconds unless noted otherwise in the respective Corefile for your version of OpenShift. The default value of 30 seconds is subject to change.
positiveTTL	**string**	positiveTTL is optional and specifies the amount of time that a positive response should be cached. If configured, it must be a value of 1s (1 second) or greater up to a theoretical maximum of several years. This field expects an unsigned duration string of decimal numbers, each with optional fraction and a unit suffix, e.g. "100s", "1m30s", "12h30m10s". Values that are fractions of a second are rounded down to the nearest second. If the configured value is less than 1s, the default value will be used. If not configured, the value will be 0s and OpenShift will use a default value of 900 seconds unless noted otherwise in the respective Corefile for your version of OpenShift. The default value of 900 seconds is subject to change.

10.1.3. .spec.nodePlacement

Description

nodePlacement provides explicit control over the scheduling of DNS pods. Generally, it is useful to run a DNS pod on every node so that DNS queries are always handled by a local DNS pod instead of going over the network to a DNS pod on another node. However, security policies may require restricting the placement of DNS pods to specific nodes. For example, if a security policy prohibits pods on arbitrary nodes from communicating with the API, a node selector can be specified to restrict DNS pods to nodes that are permitted to communicate with the API. Conversely, if running DNS pods on nodes with a particular taint is desired, a toleration can be specified for that taint. If unset, defaults are used. See nodePlacement for more details.

Type

object

Property	Type	Description
nodeSelector	**object (string)**	nodeSelector is the node selector applied to DNS pods. If empty, the default is used, which is currently the following: kubernetes.io/os: linux This default is subject to change. If set, the specified selector is used and replaces the default.
tolerations	**array**	tolerations is a list of tolerations applied to DNS pods. If empty, the DNS operator sets a toleration for the "node-role.kubernetes.io/master" taint. This default is subject to change. Specifying tolerations without including a toleration for the "node-role.kubernetes.io/master" taint may be risky as it could lead to an outage if all worker nodes become unavailable. Note that the daemon controller adds some tolerations as well. See https://kubernetes.io/docs/concepts/scheduling-eviction/taint-and-toleration/
tolerations[]	**object**	The pod this Toleration is attached to tolerates any taint that matches the triple <key,value,effect> using the matching operator <operator>.

10.1.4. .spec.nodePlacement.tolerations

Description

tolerations is a list of tolerations applied to DNS pods. If empty, the DNS operator sets a toleration for the "node-role.kubernetes.io/master" taint. This default is subject to change. Specifying tolerations without including a toleration for the "node-role.kubernetes.io/master" taint may be risky

as it could lead to an outage if all worker nodes become unavailable. Note that the daemon controller adds some tolerations as well. See https://kubernetes.io/docs/concepts/scheduling-eviction/taint-and-toleration/

Type

array

10.1.5. .spec.nodePlacement.tolerations[]

Description

The pod this Toleration is attached to tolerates any taint that matches the triple <key,value,effect> using the matching operator <operator>.

Type

object

Property	Type	Description
effect	**string**	Effect indicates the taint effect to match. Empty means match all taint effects. When specified, allowed values are NoSchedule, PreferNoSchedule and NoExecute.
key	**string**	Key is the taint key that the toleration applies to. Empty means match all taint keys. If the key is empty, operator must be Exists; this combination means to match all values and all keys.
operator	**string**	Operator represents a key's relationship to the value. Valid operators are Exists and Equal. Defaults to Equal. Exists is equivalent to wildcard for value, so that a pod can tolerate all taints of a particular category.
tolerationSeconds	**integer**	TolerationSeconds represents the period of time the toleration (which must be of effect NoExecute, otherwise this field is ignored) tolerates the taint. By default, it is not set, which means tolerate the taint forever (do not evict). Zero and negative values will be treated as 0 (evict immediately) by the system.

Property	Type	Description
value	**string**	Value is the taint value the toleration matches to. If the operator is Exists, the value should be empty, otherwise just a regular string.

10.1.6. .spec.servers

Description

servers is a list of DNS resolvers that provide name query delegation for one or more subdomains outside the scope of the cluster domain. If servers consists of more than one Server, longest suffix match will be used to determine the Server. For example, if there are two Servers, one for "foo.com" and another for "a.foo.com", and the name query is for "www.a.foo.com", it will be routed to the Server with Zone "a.foo.com". If this field is nil, no servers are created.

Type

array

10.1.7. .spec.servers[]

Description

Server defines the schema for a server that runs per instance of CoreDNS.

Type

object

Property	Type	Description
forwardPlugin	**object**	forwardPlugin defines a schema for configuring CoreDNS to proxy DNS messages to upstream resolvers.
name	**string**	name is required and specifies a unique name for the server. Name must comply with the Service Name Syntax of rfc6335.
zones	**array (string)**	zones is required and specifies the subdomains that Server is authoritative for. Zones must conform to the rfc1123 definition of a subdomain. Specifying the cluster domain (i.e., "cluster.local") is invalid.

10.1.8. .spec.servers[].forwardPlugin

Description

forwardPlugin defines a schema for configuring CoreDNS to proxy DNS messages to upstream resolvers.

Type

object

Property	Type	Description
policy	**string**	policy is used to determine the order in which upstream servers are selected for querying. Any one of the following values may be specified: * "Random" picks a random upstream server for each query. * "RoundRobin" picks upstream servers in a round-robin order, moving to the next server for each new query. * "Sequential" tries querying upstream servers in a sequential order until one responds, starting with the first server for each new query. The default value is "Random"
protocolStrategy	**string**	protocolStrategy specifies the protocol to use for upstream DNS requests. Valid values for protocolStrategy are "TCP" and omitted. When omitted, this means no opinion and the platform is left to choose a reasonable default, which is subject to change over time. The current default is to use the protocol of the original client request. "TCP" specifies that the platform should use TCP for all upstream DNS requests, even if the client request uses UDP. "TCP" is useful for UDP-specific issues such as those created by non-compliant upstream resolvers, but may consume more bandwidth or increase DNS response time. Note that protocolStrategy only affects the protocol of DNS requests that CoreDNS makes to upstream resolvers. It does not affect the protocol of DNS requests between clients and CoreDNS.

Property	Type	Description
transportConfig	**object**	transportConfig is used to configure the transport type, server name, and optional custom CA or CA bundle to use when forwarding DNS requests to an upstream resolver. The default value is "" (empty) which results in a standard cleartext connection being used when forwarding DNS requests to an upstream resolver.
upstreams	**array (string)**	upstreams is a list of resolvers to forward name queries for subdomains of Zones. Each instance of CoreDNS performs health checking of Upstreams. When a healthy upstream returns an error during the exchange, another resolver is tried from Upstreams. The Upstreams are selected in the order specified in Policy. Each upstream is represented by an IP address or IP:port if the upstream listens on a port other than 53. A maximum of 15 upstreams is allowed per ForwardPlugin.

10.1.9. .spec.servers[].forwardPlugin.transportConfig

Description

transportConfig is used to configure the transport type, server name, and optional custom CA or CA bundle to use when forwarding DNS requests to an upstream resolver. The default value is "" (empty) which results in a standard cleartext connection being used when forwarding DNS requests to an upstream resolver.

Type

object

Property	Type	Description
tls	**object**	tls contains the additional configuration options to use when Transport is set to "TLS".

Property	Type	Description
transport	**string**	transport allows cluster administrators to opt-in to using a DNS-over-TLS connection between cluster DNS and an upstream resolver(s). Configuring TLS as the transport at this level without configuring a CABundle will result in the system certificates being used to verify the serving certificate of the upstream resolver(s). Possible values: "" (empty) - This means no explicit choice has been made and the platform chooses the default which is subject to change over time. The current default is "Cleartext". "Cleartext" - Cluster admin specified cleartext option. This results in the same functionality as an empty value but may be useful when a cluster admin wants to be more explicit about the transport, or wants to switch from "TLS" to "Cleartext" explicitly. "TLS" - This indicates that DNS queries should be sent over a TLS connection. If Transport is set to TLS, you MUST also set ServerName. If a port is not included with the upstream IP, port 853 will be tried by default per RFC 7858 section 3.1; https://datatracker.ietf.org/doc/html/rfc7858#section-3.1.

10.1.10. .spec.servers[].forwardPlugin.transportConfig.tls

Description

tls contains the additional configuration options to use when Transport is set to "TLS".

Type

object

Required

- **serverName**

Property	Type	Description
caBundle	**object**	caBundle references a ConfigMap that must contain either a single CA Certificate or a CA Bundle. This allows cluster administrators to provide their own CA or CA bundle for validating the certificate of upstream resolvers. 1. The configmap must contain a **ca-bundle.crt** key. 2. The value must be a PEM encoded CA certificate or CA bundle. 3. The administrator must create this configmap in the openshift-config namespace. 4. The upstream server certificate must contain a Subject Alternative Name (SAN) that matches ServerName.
serverName	**string**	serverName is the upstream server to connect to when forwarding DNS queries. This is required when Transport is set to "TLS". ServerName will be validated against the DNS naming conventions in RFC 1123 and should match the TLS certificate installed in the upstream resolver(s).

10.1.11. .spec.servers[].forwardPlugin.transportConfig.tls.caBundle

Description

caBundle references a ConfigMap that must contain either a single CA Certificate or a CA Bundle. This allows cluster administrators to provide their own CA or CA bundle for validating the certificate of upstream resolvers. 1. The configmap must contain a **ca-bundle.crt** key. 2. The value must be a PEM encoded CA certificate or CA bundle. 3. The administrator must create this configmap in the openshift-config namespace. 4. The upstream server certificate must contain a Subject Alternative Name (SAN) that matches ServerName.

Type

object

Required

- **name**

Property	Type	Description
name	**string**	name is the metadata.name of the referenced config map

10.1.12. .spec.upstreamResolvers

Description

upstreamResolvers defines a schema for configuring CoreDNS to proxy DNS messages to upstream resolvers for the case of the default (".") server If this field is not specified, the upstream used will default to /etc/resolv.conf, with policy "sequential"

Type

object

Property	Type	Description
policy	**string**	Policy is used to determine the order in which upstream servers are selected for querying. Any one of the following values may be specified: * "Random" picks a random upstream server for each query. * "RoundRobin" picks upstream servers in a round-robin order, moving to the next server for each new query. * "Sequential" tries querying upstream servers in a sequential order until one responds, starting with the first server for each new query. The default value is "Sequential"

Property	Type	Description
protocolStrategy	**string**	protocolStrategy specifies the protocol to use for upstream DNS requests. Valid values for protocolStrategy are "TCP" and omitted. When omitted, this means no opinion and the platform is left to choose a reasonable default, which is subject to change over time. The current default is to use the protocol of the original client request. "TCP" specifies that the platform should use TCP for all upstream DNS requests, even if the client request uses UDP. "TCP" is useful for UDP-specific issues such as those created by non-compliant upstream resolvers, but may consume more bandwidth or increase DNS response time. Note that protocolStrategy only affects the protocol of DNS requests that CoreDNS makes to upstream resolvers. It does not affect the protocol of DNS requests between clients and CoreDNS.
transportConfig	**object**	transportConfig is used to configure the transport type, server name, and optional custom CA or CA bundle to use when forwarding DNS requests to an upstream resolver. The default value is "" (empty) which results in a standard cleartext connection being used when forwarding DNS requests to an upstream resolver.

Property	Type	Description
upstreams	**array**	Upstreams is a list of resolvers to forward name queries for the "." domain. Each instance of CoreDNS performs health checking of Upstreams. When a healthy upstream returns an error during the exchange, another resolver is tried from Upstreams. The Upstreams are selected in the order specified in Policy. A maximum of 15 upstreams is allowed per ForwardPlugin. If no Upstreams are specified, /etc/resolv.conf is used by default
upstreams[]	**object**	Upstream can either be of type SystemResolvConf, or of type Network. - For an Upstream of type SystemResolvConf, no further fields are necessary: The upstream will be configured to use /etc/resolv.conf. - For an Upstream of type Network, a NetworkResolver field needs to be defined with an IP address or IP:port if the upstream listens on a port other than 53.

10.1.13. .spec.upstreamResolvers.transportConfig

Description

transportConfig is used to configure the transport type, server name, and optional custom CA or CA bundle to use when forwarding DNS requests to an upstream resolver. The default value is "" (empty) which results in a standard cleartext connection being used when forwarding DNS requests to an upstream resolver.

Type

object

Property	Type	Description
tls	**object**	tls contains the additional configuration options to use when Transport is set to "TLS".

Property	Type	Description
transport	**string**	transport allows cluster administrators to opt-in to using a DNS-over-TLS connection between cluster DNS and an upstream resolver(s). Configuring TLS as the transport at this level without configuring a CABundle will result in the system certificates being used to verify the serving certificate of the upstream resolver(s). Possible values: "" (empty) - This means no explicit choice has been made and the platform chooses the default which is subject to change over time. The current default is "Cleartext". "Cleartext" - Cluster admin specified cleartext option. This results in the same functionality as an empty value but may be useful when a cluster admin wants to be more explicit about the transport, or wants to switch from "TLS" to "Cleartext" explicitly. "TLS" - This indicates that DNS queries should be sent over a TLS connection. If Transport is set to TLS, you MUST also set ServerName. If a port is not included with the upstream IP, port 853 will be tried by default per RFC 7858 section 3.1; https://datatracker.ietf.org/doc/html/rfc7858#section-3.1.

10.1.14. .spec.upstreamResolvers.transportConfig.tls

Description

tls contains the additional configuration options to use when Transport is set to "TLS".

Type

object

Required

- **serverName**

Property	Type	Description
caBundle	**object**	caBundle references a ConfigMap that must contain either a single CA Certificate or a CA Bundle. This allows cluster administrators to provide their own CA or CA bundle for validating the certificate of upstream resolvers. 1. The configmap must contain a **ca-bundle.crt** key. 2. The value must be a PEM encoded CA certificate or CA bundle. 3. The administrator must create this configmap in the openshift-config namespace. 4. The upstream server certificate must contain a Subject Alternative Name (SAN) that matches ServerName.
serverName	**string**	serverName is the upstream server to connect to when forwarding DNS queries. This is required when Transport is set to "TLS". ServerName will be validated against the DNS naming conventions in RFC 1123 and should match the TLS certificate installed in the upstream resolver(s).

10.1.15. .spec.upstreamResolvers.transportConfig.tls.caBundle

Description

caBundle references a ConfigMap that must contain either a single CA Certificate or a CA Bundle. This allows cluster administrators to provide their own CA or CA bundle for validating the certificate of upstream resolvers. 1. The configmap must contain a **ca-bundle.crt** key. 2. The value must be a PEM encoded CA certificate or CA bundle. 3. The administrator must create this configmap in the openshift-config namespace. 4. The upstream server certificate must contain a Subject Alternative Name (SAN) that matches ServerName.

Type

object

Required

- **name**

Property	Type	Description

Property	Type	Description
name	**string**	name is the metadata.name of the referenced config map

10.1.16. .spec.upstreamResolvers.upstreams

Description

Upstreams is a list of resolvers to forward name queries for the "." domain. Each instance of CoreDNS performs health checking of Upstreams. When a healthy upstream returns an error during the exchange, another resolver is tried from Upstreams. The Upstreams are selected in the order specified in Policy. A maximum of 15 upstreams is allowed per ForwardPlugin. If no Upstreams are specified, /etc/resolv.conf is used by default

Type

array

10.1.17. .spec.upstreamResolvers.upstreams[]

Description

Upstream can either be of type SystemResolvConf, or of type Network. - For an Upstream of type SystemResolvConf, no further fields are necessary: The upstream will be configured to use /etc/resolv.conf. - For an Upstream of type Network, a NetworkResolver field needs to be defined with an IP address or IP:port if the upstream listens on a port other than 53.

Type

object

Required

- **type**

Property	Type	Description
address	**string**	Address must be defined when Type is set to Network. It will be ignored otherwise. It must be a valid ipv4 or ipv6 address.
port	**integer**	Port may be defined when Type is set to Network. It will be ignored otherwise. Port must be between 65535

Property	Type	Description
type	**string**	Type defines whether this upstream contains an IP/IP:port resolver or the local /etc/resolv.conf. Type accepts 2 possible values: SystemResolvConf or Network. * When SystemResolvConf is used, the Upstream structure does not require any further fields to be defined: /etc/resolv.conf will be used * When Network is used, the Upstream structure must contain at least an Address

10.1.18. .status

Description

status is the most recently observed status of the DNS.

Type

object

Required

- **clusterDomain**
- **clusterIP**

Property	Type	Description
clusterDomain	**string**	clusterDomain is the local cluster DNS domain suffix for DNS services. This will be a subdomain as defined in RFC 1034, section 3.5: https://tools.ietf.org/html/rfc1034#section-3.5 Example: "cluster.local" More info: https://kubernetes.io/docs/concepts/services-networking/dns-pod-service

Property	Type	Description
clusterIP	**string**	clusterIP is the service IP through which this DNS is made available. In the case of the default DNS, this will be a well known IP that is used as the default nameserver for pods that are using the default ClusterFirst DNS policy. In general, this IP can be specified in a pod's spec.dnsConfig.nameservers list or used explicitly when performing name resolution from within the cluster. Example: dig foo.com @<service IP> More info: https://kubernetes.io/docs/concepts/services-networking/service/#virtual-ips-and-service-proxies
conditions	**array**	conditions provide information about the state of the DNS on the cluster. These are the supported DNS conditions: * Available - True if the following conditions are met: * DNS controller daemonset is available. - False if any of those conditions are unsatisfied.
conditions[]	**object**	OperatorCondition is just the standard condition fields.

10.1.19. .status.conditions

Description

conditions provide information about the state of the DNS on the cluster. These are the supported DNS conditions: * Available - True if the following conditions are met: * DNS controller daemonset is available. - False if any of those conditions are unsatisfied.

Type

array

10.1.20. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **type**

- type

Property	Type	Description
lastTransitionTime	**string**	
message	**string**	
reason	**string**	
status	**string**	
type	**string**	

10.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/dnses**
  - **DELETE**: delete collection of DNS
  - **GET**: list objects of kind DNS
  - **POST**: create a DNS
- **/apis/operator.openshift.io/v1/dnses/{name}**
  - **DELETE**: delete a DNS
  - **GET**: read the specified DNS
  - **PATCH**: partially update the specified DNS
  - **PUT**: replace the specified DNS
- **/apis/operator.openshift.io/v1/dnses/{name}/status**
  - **GET**: read status of the specified DNS
  - **PATCH**: partially update status of the specified DNS
  - **PUT**: replace status of the specified DNS

10.2.1. /apis/operator.openshift.io/v1/dnses

HTTP method

DELETE

Description

delete collection of DNS

Table 10.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind DNS

Table 10.2. HTTP responses

HTTP code	Reponse body
200 - OK	**DNSList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a DNS

Table 10.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 10.4. Body parameters

Parameter	Type	Description
body	**DNS** schema	

Table 10.5. HTTP responses

HTTP code	Reponse body
200 - OK	**DNS** schema
201 - Created	**DNS** schema
202 - Accepted	**DNS** schema
401 - Unauthorized	Empty

10.2.2. /apis/operator.openshift.io/v1/dnses/{name}

Table 10.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the DNS

HTTP method

DELETE

Description

delete a DNS

Table 10.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 10.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified DNS

Table 10.9. HTTP responses

HTTP code	Reponse body
200 - OK	**DNS** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified DNS

Table 10.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 10.11. HTTP responses

HTTP code	Reponse body
200 - OK	**DNS** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified DNS

Table 10.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 10.13. Body parameters

Parameter	Type	Description
body	**DNS** schema	

Table 10.14. HTTP responses

HTTP code	Reponse body
200 - OK	**DNS** schema
201 - Created	**DNS** schema
401 - Unauthorized	Empty

10.2.3. /apis/operator.openshift.io/v1/dnses/{name}/status

Table 10.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the DNS

HTTP method

GET

Description

read status of the specified DNS

Table 10.16. HTTP responses

HTTP code	Reponse body
200 - OK	**DNS** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified DNS

Table 10.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 10.18. HTTP responses

HTTP code	Reponse body
200 - OK	**DNS** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified DNS

Table 10.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 10.20. Body parameters

Parameter	Type	Description
body	**DNS** schema	

Table 10.21. HTTP responses

HTTP code	Reponse body
200 - OK	**DNS** schema
201 - Created	**DNS** schema
401 - Unauthorized	Empty

CHAPTER 11. DNSRECORD [INGRESS.OPERATOR.OPENSHIFT.IO/V1]

Description

DNSRecord is a DNS record managed in the zones defined by dns.config.openshift.io/cluster .spec.publicZone and .spec.privateZone.
Cluster admin manipulation of this resource is not supported. This resource is only for internal communication of OpenShift operators.

If DNSManagementPolicy is "Unmanaged", the operator will not be responsible for managing the DNS records on the cloud provider.

Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

11.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	spec is the specification of the desired behavior of the dnsRecord.
status	**object**	status is the most recently observed status of the dnsRecord.

11.1.1. .spec

Description

spec is the specification of the desired behavior of the dnsRecord.

Type

object

Required

- **dnsManagementPolicy**
- **dnsName**
- **recordTTL**
- **recordType**
- **targets**

Property	Type	Description
dnsManagementPolicy	**string**	dnsManagementPolicy denotes the current policy applied on the DNS record. Records that have policy set as "Unmanaged" are ignored by the ingress operator. This means that the DNS record on the cloud provider is not managed by the operator, and the "Published" status condition will be updated to "Unknown" status, since it is externally managed. Any existing record on the cloud provider can be deleted at the discretion of the cluster admin. This field defaults to Managed. Valid values are "Managed" and "Unmanaged".

Property	Type	Description
dnsName	**string**	dnsName is the hostname of the DNS record
recordTTL	**integer**	recordTTL is the record TTL in seconds. If zero, the default is 30. RecordTTL will not be used in AWS regions Alias targets, but will be used in CNAME targets, per AWS API contract.
recordType	**string**	recordType is the DNS record type. For example, "A" or "CNAME".
targets	**array (string)**	targets are record targets.

11.1.2. .status

Description

status is the most recently observed status of the dnsRecord.

Type

object

Property	Type	Description
observedGeneration	**integer**	observedGeneration is the most recently observed generation of the DNSRecord. When the DNSRecord is updated, the controller updates the corresponding record in each managed zone. If an update for a particular zone fails, that failure is recorded in the status condition for the zone so that the controller can determine that it needs to retry the update for that specific zone.
zones	**array**	zones are the status of the record in each zone.
zones[]	**object**	DNSZoneStatus is the status of a record within a specific zone.

11.1.3. .status.zones

Description

zones are the status of the record in each zone.

Type

array

11.1.4. .status.zones[]

Description

DNSZoneStatus is the status of a record within a specific zone.

Type

object

Property	Type	Description
conditions	**array**	conditions are any conditions associated with the record in the zone. If publishing the record succeeds, the "Published" condition will be set with status "True" and upon failure it will be set to "False" along with the reason and message describing the cause of the failure.
conditions[]	**object**	DNSZoneCondition is just the standard condition fields.
dnsZone	**object**	dnsZone is the zone where the record is published.

11.1.5. .status.zones[].conditions

Description

conditions are any conditions associated with the record in the zone.
If publishing the record succeeds, the "Published" condition will be set with status "True" and upon failure it will be set to "False" along with the reason and message describing the cause of the failure.

Type

array

11.1.6. .status.zones[].conditions[]

Description

DNSZoneCondition is just the standard condition fields.

Type

object

Required

- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	
message	**string**	
reason	**string**	
status	**string**	
type	**string**	

11.1.7. .status.zones[].dnsZone

Description

dnsZone is the zone where the record is published.

Type

object

Property	Type	Description

Property	Type	Description
id	**string**	id is the identifier that can be used to find the DNS hosted zone. on AWS zone can be fetched using **ID** as id in [1] on Azure zone can be fetched using **ID** as a pre-determined name in [2], on GCP zone can be fetched using **ID** as a pre-determined name in [3]. [1]: https://docs.aws.amazon.com/cli/latest/reference/route53/get-hosted-zone.html#options [2]: https://docs.microsoft.com/en-us/cli/azure/network/dns/zone?view=azure-cli-latest#az-network-dns-zone-show [3]: https://cloud.google.com/dns/docs/reference/v1/managedZones/get
tags	**object (string)**	tags can be used to query the DNS hosted zone. on AWS, resourcegroupstaggingapi [1] can be used to fetch a zone using **Tags** as tag-filters, [1]: https://docs.aws.amazon.com/cli/latest/reference/resourcegroupstaggingapi/get-resources.html#options

11.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/ingress.operator.openshift.io/v1/dnsrecords**
  - **GET**: list objects of kind DNSRecord
- **/apis/ingress.operator.openshift.io/v1/namespaces/{namespace}/dnsrecords**
  - **DELETE**: delete collection of DNSRecord
  - **GET**: list objects of kind DNSRecord
  - **POST**: create a DNSRecord

- **/apis/ingress.operator.openshift.io/v1/namespaces/{namespace}/dnsrecords/{name}**
  - **DELETE**: delete a DNSRecord
  - **GET**: read the specified DNSRecord
  - **PATCH**: partially update the specified DNSRecord
  - **PUT**: replace the specified DNSRecord
- **/apis/ingress.operator.openshift.io/v1/namespaces/{namespace}/dnsrecords/{name}/status**
  - **GET**: read status of the specified DNSRecord
  - **PATCH**: partially update status of the specified DNSRecord
  - **PUT**: replace status of the specified DNSRecord

11.2.1. /apis/ingress.operator.openshift.io/v1/dnsrecords

HTTP method

GET

Description

list objects of kind DNSRecord

Table 11.1. HTTP responses

HTTP code	Reponse body
200 - OK	**DNSRecordList** schema
401 - Unauthorized	Empty

11.2.2. /apis/ingress.operator.openshift.io/v1/namespaces/{namespace}/dnsrecords

HTTP method

DELETE

Description

delete collection of DNSRecord

Table 11.2. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind DNSRecord

Table 11.3. HTTP responses

HTTP code	Reponse body
200 - OK	**DNSRecordList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a DNSRecord

Table 11.4. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 11.5. Body parameters

Parameter	Type	Description
body	**DNSRecord** schema	

Table 11.6. HTTP responses

HTTP code	Reponse body
200 - OK	**DNSRecord** schema
201 - Created	**DNSRecord** schema
202 - Accepted	**DNSRecord** schema
401 - Unauthorized	Empty

11.2.3. /apis/ingress.operator.openshift.io/v1/namespaces/{namespace}/dnsrecords/{

Table 11.7. Global path parameters

Parameter	Type	Description
name	**string**	name of the DNSRecord

HTTP method

DELETE

Description

delete a DNSRecord

Table 11.8. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 11.9. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema

HTTP code	Reponse body
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified DNSRecord

Table 11.10. HTTP responses

HTTP code	Reponse body
200 - OK	**DNSRecord** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified DNSRecord

Table 11.11. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 11.12. HTTP responses

HTTP code	Reponse body
200 - OK	**DNSRecord** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified DNSRecord

Table 11.13. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 11.14. Body parameters

Parameter	Type	Description
body	**DNSRecord** schema	

Table 11.15. HTTP responses

HTTP code	Reponse body
200 - OK	**DNSRecord** schema
201 - Created	**DNSRecord** schema
401 - Unauthorized	Empty

11.2.4. /apis/ingress.operator.openshift.io/v1/namespaces/{namespace}/dnsrecords/{

Table 11.16. Global path parameters

Parameter	Type	Description
name	**string**	name of the DNSRecord

HTTP method

GET

Description

read status of the specified DNSRecord

Table 11.17. HTTP responses

HTTP code	Reponse body
200 - OK	**DNSRecord** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified DNSRecord

Table 11.18. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 11.19. HTTP responses

HTTP code	Reponse body
200 - OK	**DNSRecord** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified DNSRecord

Table 11.20. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 11.21. Body parameters

Parameter	Type	Description
body	**DNSRecord** schema	

Table 11.22. HTTP responses

HTTP code	Reponse body
200 - OK	**DNSRecord** schema
201 - Created	**DNSRecord** schema
401 - Unauthorized	Empty

CHAPTER 12. ETCD [OPERATOR.OPENSHIFT.IO/V1]

Description

Etcd provides information to configure an operator to manage etcd.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

12.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	
status	**object**	

12.1.1. .spec

Description

Type

object

Property	Type	Description
controlPlaneHardwareSpeed	**string**	HardwareSpeed allows user to change the etcd tuning profile which configures the latency parameters for heartbeat interval and leader election timeouts allowing the cluster to tolerate longer round-trip-times between etcd members. Valid values are "", "Standard" and "Slower". "" means no opinion and the platform is left to choose a reasonable default which is subject to change without notice.
failedRevisionLimit	**integer**	failedRevisionLimit is the number of failed static pod installer revisions to keep on disk and in the api -1 = unlimited, 0 or unset = 5 (default)
forceRedeploymentReason	**string**	forceRedeploymentReason can be used to force the redeployment of the operand by providing a unique string. This provides a mechanism to kick a previously failed deployment and provide a reason why you think it will work this time instead of failing again on the same config.
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".

Property	Type	Description
managementState	**string**	managementState indicates whether and how the operator should manage the component
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
succeededRevisionLimit	**integer**	succeededRevisionLimit is the number of successful static pod installer revisions to keep on disk and in the api -1 = unlimited, 0 or unset = 5 (default)
unsupportedConfigOverrides	``	unsupportedConfigOverrides overrides the final configuration that was computed by the operator. Red Hat does not support the use of this field. Misuse of this field could lead to unexpected behavior or conflict with other configuration options. Seek guidance from the Red Hat support before using this field. Use of this property blocks cluster upgrades, it must be removed before upgrading your cluster.

12.1.2. .status

Description

Type

object

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status
conditions[]	**object**	OperatorCondition is just the standard condition fields.
controlPlaneHardwareSpeed	**string**	ControlPlaneHardwareSpeed declares valid hardware speed tolerance levels
generations	**array**	generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.
generations[]	**object**	GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.
latestAvailableRevision	**integer**	latestAvailableRevision is the deploymentID of the most recent deployment
latestAvailableRevisionReason	**string**	latestAvailableRevisionReason describe the detailed reason for the most recent deployment
nodeStatuses	**array**	nodeStatuses track the deployment values and errors across individual nodes
nodeStatuses[]	**object**	NodeStatus provides information about the current state of a particular node managed by this operator.
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with
readyReplicas	**integer**	readyReplicas indicates how many replicas are ready and at the desired state
version	**string**	version is the level this availability applies to

12.1.3. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

12.1.4. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	
reason	**string**	
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

12.1.5. .status.generations

Description

generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Type

array

12.1.6. .status.generations[]

Description

GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.

Type

object

Required

- **group**
- **name**
- **namespace**
- **resource**

Property	Type	Description
group	**string**	group is the group of the thing you're tracking
hash	**string**	hash is an optional field set for resources without generation that are content sensitive like secrets and configmaps
lastGeneration	**integer**	lastGeneration is the last generation of the workload controller involved
name	**string**	name is the name of the thing you're tracking
namespace	**string**	namespace is where the thing you're tracking is
resource	**string**	resource is the resource type of the thing you're tracking

12.1.7. .status.nodeStatuses

Description

nodeStatuses track the deployment values and errors across individual nodes

Type

array

12.1.8. .status.nodeStatuses[]

Description

NodeStatus provides information about the current state of a particular node managed by this operator.

Type

object

Required

- **nodeName**

Property	Type	Description
currentRevision	**integer**	currentRevision is the generation of the most recently successful deployment. Can not be set on creation of a nodeStatus. Updates must only increase the value.
lastFailedCount	**integer**	lastFailedCount is how often the installer pod of the last failed revision failed.
lastFailedReason	**string**	lastFailedReason is a machine readable failure reason string.
lastFailedRevision	**integer**	lastFailedRevision is the generation of the deployment we tried and failed to deploy.
lastFailedRevisionErrors	**array (string)**	lastFailedRevisionErrors is a list of human readable errors during the failed deployment referenced in lastFailedRevision.
lastFailedTime	**string**	lastFailedTime is the time the last failed revision failed the last time.
lastFallbackCount	**integer**	lastFallbackCount is how often a fallback to a previous revision happened.
nodeName	**string**	nodeName is the name of the node
targetRevision	**integer**	targetRevision is the generation of the deployment we're trying to apply. Can not be set on creation of a nodeStatus.

12.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/etcds**
  - **DELETE**: delete collection of Etcd
  - **GET**: list objects of kind Etcd
  - **POST**: create an Etcd
- **/apis/operator.openshift.io/v1/etcds/{name}**
  - **DELETE**: delete an Etcd
  - **GET**: read the specified Etcd
  - **PATCH**: partially update the specified Etcd
  - **PUT**: replace the specified Etcd
- **/apis/operator.openshift.io/v1/etcds/{name}/status**
  - **GET**: read status of the specified Etcd
  - **PATCH**: partially update status of the specified Etcd
  - **PUT**: replace status of the specified Etcd

12.2.1. /apis/operator.openshift.io/v1/etcds

HTTP method

DELETE

Description

delete collection of Etcd

Table 12.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind Etcd

Table 12.2. HTTP responses

HTTP code	Reponse body
200 - OK	**EtcdList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create an Etcd

Table 12.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 12.4. Body parameters

Parameter	Type	Description
body	**Etcd** schema	

Table 12.5. HTTP responses

HTTP code	Reponse body
200 - OK	**Etcd** schema
201 - Created	**Etcd** schema
202 - Accepted	**Etcd** schema
401 - Unauthorized	Empty

12.2.2. /apis/operator.openshift.io/v1/etcds/{name}

Table 12.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the Etcd

HTTP method

DELETE

Description

delete an Etcd

Table 12.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 12.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified Etcd

Table 12.9. HTTP responses

HTTP code	Reponse body
200 - OK	**Etcd** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified Etcd

Table 12.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 12.11. HTTP responses

HTTP code	Reponse body
200 - OK	**Etcd** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified Etcd

Table 12.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 12.13. Body parameters

Parameter	Type	Description
body	**Etcd** schema	

Table 12.14. HTTP responses

HTTP code	Reponse body
200 - OK	**Etcd** schema
201 - Created	**Etcd** schema
401 - Unauthorized	Empty

12.2.3. /apis/operator.openshift.io/v1/etcds/{name}/status

Table 12.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the Etcd

HTTP method

GET

Description

read status of the specified Etcd

Table 12.16. HTTP responses

HTTP code	Reponse body
200 - OK	**Etcd** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified Etcd

Table 12.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 12.18. HTTP responses

HTTP code	Reponse body
200 - OK	**Etcd** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified Etcd

Table 12.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 12.20. Body parameters

Parameter	Type	Description
body	**Etcd** schema	

Table 12.21. HTTP responses

HTTP code	Reponse body
200 - OK	**Etcd** schema
201 - Created	**Etcd** schema
401 - Unauthorized	Empty

CHAPTER 13. IMAGECONTENTSOURCEPOLICY [OPERATOR.OPENSHIFT.IO/V1ALPHA1]

Description

ImageContentSourcePolicy holds cluster-wide information about how to handle registry mirror rules. When multiple policies are defined, the outcome of the behavior is defined on each field. Compatibility level 4: No compatibility is provided, the API can change at any point for any reason. These capabilities should not be used by applications needing long term support.

Type

object

Required

- **spec**

13.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	spec holds user settable values for configuration

13.1.1. .spec

Description

spec holds user settable values for configuration

Type

object

Property	Type	Description

Property	Type	Description
repositoryDigestMirrors	**array**	repositoryDigestMirrors allows images referenced by image digests in pods to be pulled from alternative mirrored repository locations. The image pull specification provided to the pod will be compared to the source locations described in RepositoryDigestMirrors and the image may be pulled down from any of the mirrors in the list instead of the specified repository allowing administrators to choose a potentially faster mirror. Only image pull specifications that have an image digest will have this behavior applied to them - tags will continue to be pulled from the specified repository in the pull spec. Each “source” repository is treated independently; configurations for different “source” repositories don’t interact. When multiple policies are defined for the same “source” repository, the sets of defined mirrors will be merged together, preserving the relative order of the mirrors, if possible. For example, if policy A has mirrors **a, b, c** and policy B has mirrors **c, d, e**, the mirrors will be used in the order **a, b, c, d, e**. If the orders of mirror entries conflict (e.g. **a, b** vs. **b, a**) the configuration is not rejected but the resulting order is unspecified.
repositoryDigestMirrors[]	**object**	RepositoryDigestMirrors holds cluster-wide information about how to handle mirros in the registries config. Note: the mirrors only work when pulling the images that are referenced by their digests.

13.1.2. .spec.repositoryDigestMirrors

Description

repositoryDigestMirrors allows images referenced by image digests in pods to be pulled from alternative mirrored repository locations. The image pull specification provided to the pod will be compared to the source locations described in RepositoryDigestMirrors and the image may be pulled down from any of the mirrors in the list instead of the specified repository allowing administrators to choose a potentially faster mirror. Only image pull specifications that have an image digest will have this behavior applied to them - tags will continue to be pulled from the specified repository in the pull spec.
Each “source” repository is treated independently; configurations for different “source” repositories don’t interact.

When multiple policies are defined for the same “source” repository, the sets of defined mirrors will be merged together, preserving the relative order of the mirrors, if possible. For example, if policy A has mirrors **a, b, c** and policy B has mirrors **c, d, e**, the mirrors will be used in the order **a, b, c, d, e**. If the orders of mirror entries conflict (e.g. **a, b** vs. **b, a**) the configuration is not rejected but the resulting order is unspecified.

Type

array

13.1.3. .spec.repositoryDigestMirrors[]

Description

RepositoryDigestMirrors holds cluster-wide information about how to handle mirros in the registries config. Note: the mirrors only work when pulling the images that are referenced by their digests.

Type

object

Required

- **source**

Property	Type	Description
mirrors	**array (string)**	mirrors is one or more repositories that may also contain the same images. The order of mirrors in this list is treated as the user’s desired priority, while source is by default considered lower priority than all mirrors. Other cluster configuration, including (but not limited to) other repositoryDigestMirrors objects, may impact the exact order mirrors are contacted in, or some mirrors may be contacted in parallel, so this should be considered a preference rather than a guarantee of ordering.

Property	Type	Description
source	**string**	source is the repository that users refer to, e.g. in image pull specifications.

13.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1alpha1/imagecontentsourcepolicies**
  - **DELETE**: delete collection of ImageContentSourcePolicy
  - **GET**: list objects of kind ImageContentSourcePolicy
  - **POST**: create an ImageContentSourcePolicy
- **/apis/operator.openshift.io/v1alpha1/imagecontentsourcepolicies/{name}**
  - **DELETE**: delete an ImageContentSourcePolicy
  - **GET**: read the specified ImageContentSourcePolicy
  - **PATCH**: partially update the specified ImageContentSourcePolicy
  - **PUT**: replace the specified ImageContentSourcePolicy
- **/apis/operator.openshift.io/v1alpha1/imagecontentsourcepolicies/{name}/status**
  - **GET**: read status of the specified ImageContentSourcePolicy
  - **PATCH**: partially update status of the specified ImageContentSourcePolicy
  - **PUT**: replace status of the specified ImageContentSourcePolicy

13.2.1. /apis/operator.openshift.io/v1alpha1/imagecontentsourcepolicies

HTTP method

DELETE

Description

delete collection of ImageContentSourcePolicy

Table 13.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind ImageContentSourcePolicy

Table 13.2. HTTP responses

HTTP code	Reponse body
200 - OK	**ImageContentSourcePolicyList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create an ImageContentSourcePolicy

Table 13.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 13.4. Body parameters

Parameter	Type	Description
body	**ImageContentSourcePolicy** schema	

Table 13.5. HTTP responses

HTTP code	Reponse body
200 - OK	**ImageContentSourcePolicy** schema
201 - Created	**ImageContentSourcePolicy** schema
202 - Accepted	**ImageContentSourcePolicy** schema
401 - Unauthorized	Empty

13.2.2. /apis/operator.openshift.io/v1alpha1/imagecontentsourcepolicies/{name}

Table 13.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the ImageContentSourcePolicy

HTTP method

DELETE

Description

delete an ImageContentSourcePolicy

Table 13.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 13.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema

HTTP code	Reponse body
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified ImageContentSourcePolicy

Table 13.9. HTTP responses

HTTP code	Reponse body
200 - OK	**ImageContentSourcePolicy** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified ImageContentSourcePolicy

Table 13.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 13.11. HTTP responses

HTTP code	Reponse body
200 - OK	**ImageContentSourcePolicy** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified ImageContentSourcePolicy

Table 13.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 13.13. Body parameters

Parameter	Type	Description
body	**ImageContentSourcePolicy** schema	

Table 13.14. HTTP responses

HTTP code	Reponse body
200 - OK	**ImageContentSourcePolicy** schema
201 - Created	**ImageContentSourcePolicy** schema
401 - Unauthorized	Empty

13.2.3. /apis/operator.openshift.io/v1alpha1/imagecontentsourcepolicies/{name}/stat

Table 13.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the ImageContentSourcePolicy

HTTP method

GET

Description

read status of the specified ImageContentSourcePolicy

Table 13.16. HTTP responses

HTTP code	Reponse body
200 - OK	**ImageContentSourcePolicy** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified ImageContentSourcePolicy

Table 13.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 13.18. HTTP responses

HTTP code	Reponse body
200 - OK	**ImageContentSourcePolicy** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified ImageContentSourcePolicy

Table 13.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 13.20. Body parameters

Parameter	Type	Description
body	**ImageContentSourcePolicy** schema	

Table 13.21. HTTP responses

HTTP code	Reponse body
200 - OK	**ImageContentSourcePolicy** schema
201 - Created	**ImageContentSourcePolicy** schema
401 - Unauthorized	Empty

CHAPTER 14. IMAGEPRUNER [IMAGEREGISTRY.OPERATOR.OPENSHIFT.IO/V1]

Description

ImagePruner is the configuration object for an image registry pruner managed by the registry operator.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **metadata**
- **spec**

14.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	ImagePrunerSpec defines the specs for the running image pruner.
status	**object**	ImagePrunerStatus reports image pruner operational status.

14.1.1. .spec

Description

ImagePrunerSpec defines the specs for the running image pruner.

Type

object

Property	Type	Description
affinity	**object**	affinity is a group of node affinity scheduling rules for the image pruner pod.
failedJobsHistoryLimit	**integer**	failedJobsHistoryLimit specifies how many failed image pruner jobs to retain. Defaults to 3 if not set.
ignoreInvalidImageReferences	**boolean**	ignoreInvalidImageReferences indicates whether the pruner can ignore errors while parsing image references.
keepTagRevisions	**integer**	keepTagRevisions specifies the number of image revisions for a tag in an image stream that will be preserved. Defaults to 3.
keepYoungerThan	**integer**	keepYoungerThan specifies the minimum age in nanoseconds of an image and its referrers for it to be considered a candidate for pruning. DEPRECATED: This field is deprecated in favor of keepYoungerThanDuration. If both are set, this field is ignored and keepYoungerThanDuration takes precedence.

Property	Type	Description
keepYoungerThanDuration	**string**	keepYoungerThanDuration specifies the minimum age of an image and its referrers for it to be considered a candidate for pruning. Defaults to 60m (60 minutes).
logLevel	**string**	logLevel sets the level of log output for the pruner job. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
nodeSelector	**object (string)**	nodeSelector defines the node selection constraints for the image pruner pod.
resources	**object**	resources defines the resource requests and limits for the image pruner pod.
schedule	**string**	schedule specifies when to execute the job using standard cronjob syntax: https://wikipedia.org/wiki/Cron. Defaults to **0 0 * * ***.
successfulJobsHistoryLimit	**integer**	successfulJobsHistoryLimit specifies how many successful image pruner jobs to retain. Defaults to 3 if not set.
suspend	**boolean**	suspend specifies whether or not to suspend subsequent executions of this cronjob. Defaults to false.
tolerations	**array**	tolerations defines the node tolerations for the image pruner pod.
tolerations[]	**object**	The pod this Toleration is attached to tolerates any taint that matches the triple <key,value,effect> using the matching operator <operator>.

14.1.2. .spec.affinity

Description

affinity is a group of node affinity scheduling rules for the image pruner pod.

Type

object

Property	Type	Description
nodeAffinity	**object**	Describes node affinity scheduling rules for the pod.
podAffinity	**object**	Describes pod affinity scheduling rules (e.g. co-locate this pod in the same node, zone, etc. as some other pod(s)).
podAntiAffinity	**object**	Describes pod anti-affinity scheduling rules (e.g. avoid putting this pod in the same node, zone, etc. as some other pod(s)).

14.1.3. .spec.affinity.nodeAffinity

Description

Describes node affinity scheduling rules for the pod.

Type

object

Property	Type	Description
preferredDuringSchedulingIgnoredDuringExecution	**array**	The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node matches the corresponding matchExpressions; the node(s) with the highest sum are the most preferred.

Property	Type	Description
preferredDuringSchedulingIgnoredDuringExecution[]	**object**	An empty preferred scheduling term matches all objects with implicit weight 0 (i.e. it's a no-op). A null preferred scheduling term matches no objects (i.e. is also a no-op).
requiredDuringSchedulingIgnoredDuringExecution	**object**	If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to an update), the system may or may not try to eventually evict the pod from its node.

14.1.4. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution

Description

The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node matches the corresponding matchExpressions; the node(s) with the highest sum are the most preferred.

Type

array

14.1.5. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[

Description

An empty preferred scheduling term matches all objects with implicit weight 0 (i.e. it's a no-op). A null preferred scheduling term matches no objects (i.e. is also a no-op).

Type

object

Required

- **preference**
- **weight**

Property	Type	Description
preference	**object**	A node selector term, associated with the corresponding weight.
weight	**integer**	Weight associated with matching the corresponding nodeSelectorTerm, in the range 1-100.

14.1.6. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[

Description

A node selector term, associated with the corresponding weight.

Type

object

Property	Type	Description
matchExpressions	**array**	A list of node selector requirements by node's labels.
matchExpressions[]	**object**	A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchFields	**array**	A list of node selector requirements by node's fields.

Property	Type	Description
matchFields[]	**object**	A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

14.1.7. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[

Description

A list of node selector requirements by node's labels.

Type

array

14.1.8. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	The label key that the selector applies to.
operator	**string**	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.

Property	Type	Description
values	**array (string)**	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

14.1.9. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution[

Description

A list of node selector requirements by node's fields.

Type

array

14.1.10. .spec.affinity.nodeAffinity.preferredDuringSchedulingIgnoredDuringExecution

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	The label key that the selector applies to.
operator	**string**	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.

Property	Type	Description
values	**array (string)**	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

14.1.11. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution

Description

If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to an update), the system may or may not try to eventually evict the pod from its node.

Type

object

Required

- **nodeSelectorTerms**

Property	Type	Description
nodeSelectorTerms	**array**	Required. A list of node selector terms. The terms are ORed.
nodeSelectorTerms[]	**object**	A null or empty node selector term matches no objects. The requirements of them are ANDed. The TopologySelectorTerm type implements a subset of the NodeSelectorTerm.

14.1.12. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.r

Description

Required. A list of node selector terms. The terms are ORed.

Type

array

14.1.13. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.r

Description

A null or empty node selector term matches no objects. The requirements of them are ANDed. The TopologySelectorTerm type implements a subset of the NodeSelectorTerm.

Type

object

Property	Type	Description
matchExpressions	**array**	A list of node selector requirements by node's labels.
matchExpressions[]	**object**	A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchFields	**array**	A list of node selector requirements by node's fields.
matchFields[]	**object**	A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

14.1.14. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.n

Description

A list of node selector requirements by node's labels.

Type

array

14.1.15. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.n

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	The label key that the selector applies to.
operator	**string**	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.
values	**array (string)**	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

14.1.16. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.r

Description

A list of node selector requirements by node's fields.

Type

array

14.1.17. .spec.affinity.nodeAffinity.requiredDuringSchedulingIgnoredDuringExecution.r

Description

A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	The label key that the selector applies to.

Property	Type	Description
operator	**string**	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt.
values	**array (string)**	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

14.1.18. .spec.affinity.podAffinity

Description

Describes pod affinity scheduling rules (e.g. co-locate this pod in the same node, zone, etc. as some other pod(s)).

Type

object

Property	Type	Description
preferredDuringSchedulingIgnoredDuringExecution	**array**	The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.

Property	Type	Description
preferredDuringSchedulingIgnoredDuringExecution[]	**object**	The weights of all of the matched WeightedPodAffinityTerm fields are added per-node to find the most preferred node(s)
requiredDuringSchedulingIgnoredDuringExecution	**array**	If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.
requiredDuringSchedulingIgnoredDuringExecution[]	**object**	Defines a set of pods (namely those matching the labelSelector relative to the given namespace(s)) that this pod should be co-located (affinity) or not co-located (anti-affinity) with, where co-located is defined as running on a node whose value of the label with key <topologyKey> matches that of any node on which a pod of the set of pods is running

14.1.19. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution

Description

The scheduler will prefer to schedule pods to nodes that satisfy the affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.

Type

array

14.1.20. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[

Description

The weights of all of the matched WeightedPodAffinityTerm fields are added per-node to find the most preferred node(s)

Type

object

Required

- **podAffinityTerm**
- **weight**

Property	Type	Description
podAffinityTerm	**object**	Required. A pod affinity term, associated with the corresponding weight.
weight	**integer**	weight associated with matching the corresponding podAffinityTerm, in the range 1-100.

14.1.21. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[]

Description

Required. A pod affinity term, associated with the corresponding weight.

Type

object

Required

- **topologyKey**

Property	Type	Description
labelSelector	**object**	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Property	Type	Description
matchLabelKeys	**array (string)**	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with **labelSelector** as **key in (value)** to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).
mismatchLabelKeys	**array (string)**	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with **labelSelector** as **key notin (value)** to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).

Property	Type	Description
namespaceSelector	**object**	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
namespaces	**array (string)**	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".
topologyKey	**string**	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

14.1.22. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.

Property	Type	Description
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

14.1.23. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

14.1.24. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

14.1.25. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

14.1.26. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

14.1.27. .spec.affinity.podAffinity.preferredDuringSchedulingIgnoredDuringExecution[

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

14.1.28. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution

Description

If the affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.

Type

array

14.1.29. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[]

Description

Defines a set of pods (namely those matching the labelSelector relative to the given namespace(s)) that this pod should be co-located (affinity) or not co-located (anti-affinity) with, where co-located is defined as running on a node whose value of the label with key <topologyKey> matches that of any

node on which a pod of the set of pods is running

Type

object

Required

- **topologyKey**

Property	Type	Description
labelSelector	**object**	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.
matchLabelKeys	**array (string)**	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with **labelSelector** as **key in (value)** to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).

Property	Type	Description
mismatchLabelKeys	**array (string)**	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with **labelSelector** as **key notin (value)** to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).
namespaceSelector	**object**	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
namespaces	**array (string)**	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Property	Type	Description
topologyKey	**string**	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

14.1.30. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[]

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

14.1.31. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[].

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

14.1.32. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

14.1.33. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[]

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.

Property	Type	Description
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

14.1.34. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[]

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

14.1.35. .spec.affinity.podAffinity.requiredDuringSchedulingIgnoredDuringExecution[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

14.1.36. .spec.affinity.podAntiAffinity

Description

Describes pod anti-affinity scheduling rules (e.g. avoid putting this pod in the same node, zone, etc. as some other pod(s)).

Type

object

Property	Type	Description
preferredDuringSchedulingIgnoredDuringExecution	**array**	The scheduler will prefer to schedule pods to nodes that satisfy the anti-affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling anti-affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.
preferredDuringSchedulingIgnoredDuringExecution[]	**object**	The weights of all of the matched WeightedPodAffinityTerm fields are added per-node to find the most preferred node(s)

Property	Type	Description
requiredDuringSchedulingIgnoredDuringExecution	**array**	If the anti-affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the anti-affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.
requiredDuringSchedulingIgnoredDuringExecution[]	**object**	Defines a set of pods (namely those matching the labelSelector relative to the given namespace(s)) that this pod should be co-located (affinity) or not co-located (anti-affinity) with, where co-located is defined as running on a node whose value of the label with key <topologyKey> matches that of any node on which a pod of the set of pods is running

14.1.37. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecut

Description

The scheduler will prefer to schedule pods to nodes that satisfy the anti-affinity expressions specified by this field, but it may choose a node that violates one or more of the expressions. The node that is most preferred is the one with the greatest sum of weights, i.e. for each node that meets all of the scheduling requirements (resource request, requiredDuringScheduling anti-affinity expressions, etc.), compute a sum by iterating through the elements of this field and adding "weight" to the sum if the node has pods which matches the corresponding podAffinityTerm; the node(s) with the highest sum are the most preferred.

Type

array

14.1.38. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecu

Description

The weights of all of the matched WeightedPodAffinityTerm fields are added per-node to find the most preferred node(s)

Type

object

Required

- **podAffinityTerm**
- **weight**

Property	Type	Description
podAffinityTerm	**object**	Required. A pod affinity term, associated with the corresponding weight.
weight	**integer**	weight associated with matching the corresponding podAffinityTerm, in the range 1-100.

14.1.39. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecu

Description

Required. A pod affinity term, associated with the corresponding weight.

Type

object

Required

- **topologyKey**

Property	Type	Description
labelSelector	**object**	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Property	Type	Description
matchLabelKeys	**array (string)**	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with **labelSelector** as **key in (value)** to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).
mismatchLabelKeys	**array (string)**	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with **labelSelector** as **key notin (value)** to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).

Property	Type	Description
namespaceSelector	**object**	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
namespaces	**array (string)**	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".
topologyKey	**string**	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

14.1.40. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecu

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.

Property	Type	Description
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

14.1.41. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecut

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

14.1.42. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecu

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

14.1.43. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecu

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

14.1.44. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecu

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

14.1.45. .spec.affinity.podAntiAffinity.preferredDuringSchedulingIgnoredDuringExecu

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

14.1.46. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecuti

Description

If the anti-affinity requirements specified by this field are not met at scheduling time, the pod will not be scheduled onto the node. If the anti-affinity requirements specified by this field cease to be met at some point during pod execution (e.g. due to a pod label update), the system may or may not try to eventually evict the pod from its node. When there are multiple elements, the lists of nodes corresponding to each podAffinityTerm are intersected, i.e. all terms must be satisfied.

Type

array

14.1.47. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecuti

Description

Defines a set of pods (namely those matching the labelSelector relative to the given namespace(s))

that this pod should be co-located (affinity) or not co-located (anti-affinity) with, where co-located is defined as running on a node whose value of the label with key <topologyKey> matches that of any node on which a pod of the set of pods is running

Type

object

Required

- **topologyKey**

Property	Type	Description
labelSelector	**object**	A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.
matchLabelKeys	**array (string)**	MatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with **labelSelector** as **key in (value)** to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both matchLabelKeys and labelSelector. Also, matchLabelKeys cannot be set when labelSelector isn't set. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).

Property	Type	Description
mismatchLabelKeys	**array (string)**	MismatchLabelKeys is a set of pod label keys to select which pods will be taken into consideration. The keys are used to lookup values from the incoming pod labels, those key-value labels are merged with **labelSelector** as **key notin (value)** to select the group of existing pods which pods will be taken into consideration for the incoming pod's pod (anti) affinity. Keys that don't exist in the incoming pod labels will be ignored. The default value is empty. The same key is forbidden to exist in both mismatchLabelKeys and labelSelector. Also, mismatchLabelKeys cannot be set when labelSelector isn't set. This is a beta field and requires enabling MatchLabelKeysInPodAffinity feature gate (enabled by default).
namespaceSelector	**object**	A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.
namespaces	**array (string)**	namespaces specifies a static list of namespace names that the term applies to. The term is applied to the union of the namespaces listed in this field and the ones selected by namespaceSelector. null or empty namespaces list and null namespaceSelector means "this pod's namespace".

Property	Type	Description
topologyKey	**string**	This pod should be co-located (affinity) or not co-located (anti-affinity) with the pods matching the labelSelector in the specified namespaces, where co-located is defined as running on a node whose value of the label with key topologyKey matches that of any node on which any of the selected pods is running. Empty topologyKey is not allowed.

14.1.48. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecuti

Description

A label query over a set of resources, in this case pods. If it's null, this PodAffinityTerm matches with no Pods.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

14.1.49. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecuti

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

14.1.50. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecuti

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

14.1.51. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecutic

Description

A label query over the set of namespaces that the term applies to. The term is applied to the union of the namespaces selected by this field and the ones listed in the namespaces field. null selector and null or empty namespaces list means "this pod's namespace". An empty selector ({}) matches all namespaces.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.

Property	Type	Description
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

14.1.52. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecuti

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

14.1.53. .spec.affinity.podAntiAffinity.requiredDuringSchedulingIgnoredDuringExecuti

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

14.1.54. .spec.resources

Description

resources defines the resource requests and limits for the image pruner pod.

Type

object

Property	Type	Description
claims	**array**	Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container. This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.
claims[]	**object**	ResourceClaim references one entry in PodSpec.ResourceClaims.
limits	**integer-or-string**	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Property	Type	Description
requests	**integer-or-string**	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

14.1.55. .spec.resources.claims

Description

Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container. This is an alpha field and requires enabling the DynamicResourceAllocation feature gate.

This field is immutable. It can only be set for containers.

Type

array

14.1.56. .spec.resources.claims[]

Description

ResourceClaim references one entry in PodSpec.ResourceClaims.

Type

object

Required

- **name**

Property	Type	Description
name	**string**	Name must match the name of one entry in pod.spec.resourceClaims of the Pod where this field is used. It makes that resource available inside a container.

Property	Type	Description
request	**string**	Request is the name chosen for a request in the referenced claim. If empty, everything from the claim is made available, otherwise only the result of this request.

14.1.57. .spec.tolerations

Description

tolerations defines the node tolerations for the image pruner pod.

Type

array

14.1.58. .spec.tolerations[]

Description

The pod this Toleration is attached to tolerates any taint that matches the triple <key,value,effect> using the matching operator <operator>.

Type

object

Property	Type	Description
effect	**string**	Effect indicates the taint effect to match. Empty means match all taint effects. When specified, allowed values are NoSchedule, PreferNoSchedule and NoExecute.
key	**string**	Key is the taint key that the toleration applies to. Empty means match all taint keys. If the key is empty, operator must be Exists; this combination means to match all values and all keys.
operator	**string**	Operator represents a key's relationship to the value. Valid operators are Exists and Equal. Defaults to Equal. Exists is equivalent to wildcard for value, so that a pod can tolerate all taints of a particular category.

Property	Type	Description
tolerationSeconds	**integer**	TolerationSeconds represents the period of time the toleration (which must be of effect NoExecute, otherwise this field is ignored) tolerates the taint. By default, it is not set, which means tolerate the taint forever (do not evict). Zero and negative values will be treated as 0 (evict immediately) by the system.
value	**string**	Value is the taint value the toleration matches to. If the operator is Exists, the value should be empty, otherwise just a regular string.

14.1.59. .status

Description

ImagePrunerStatus reports image pruner operational status.

Type

object

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status.
conditions[]	**object**	OperatorCondition is just the standard condition fields.
observedGeneration	**integer**	observedGeneration is the last generation change that has been applied.

14.1.60. .status.conditions

Description

conditions is a list of conditions and their status.

Type

array

14.1.61. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	
reason	**string**	
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

14.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/imageregistry.operator.openshift.io/v1/imagepruners**
  - **DELETE**: delete collection of ImagePruner
  - **GET**: list objects of kind ImagePruner
  - **POST**: create an ImagePruner
- **/apis/imageregistry.operator.openshift.io/v1/imagepruners/{name}**
  - **DELETE**: delete an ImagePruner
  - **GET**: read the specified ImagePruner
  - **PATCH**: partially update the specified ImagePruner

- **PUT**: replace the specified ImagePruner

- **/apis/imageregistry.operator.openshift.io/v1/imagepruners/{name}/status**
  - **GET**: read status of the specified ImagePruner
  - **PATCH**: partially update status of the specified ImagePruner
  - **PUT**: replace status of the specified ImagePruner

14.2.1. /apis/imageregistry.operator.openshift.io/v1/imagepruners

HTTP method

DELETE

Description

delete collection of ImagePruner

Table 14.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind ImagePruner

Table 14.2. HTTP responses

HTTP code	Reponse body
200 - OK	**ImagePrunerList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create an ImagePruner

Table 14.3. Query parameters

Parameter	Type	Description

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 14.4. Body parameters

Parameter	Type	Description
body	**ImagePruner** schema	

Table 14.5. HTTP responses

HTTP code	Reponse body
200 - OK	**ImagePruner** schema
201 - Created	**ImagePruner** schema
202 - Accepted	**ImagePruner** schema
401 - Unauthorized	Empty

14.2.2. /apis/imageregistry.operator.openshift.io/v1/imagepruners/{name}

Table 14.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the ImagePruner

HTTP method

DELETE

Description

delete an ImagePruner

Table 14.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 14.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified ImagePruner

Table 14.9. HTTP responses

HTTP code	Reponse body
200 - OK	**ImagePruner** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified ImagePruner

Table 14.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 14.11. HTTP responses

HTTP code	Reponse body
200 - OK	**ImagePruner** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified ImagePruner

Table 14.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 14.13. Body parameters

Parameter	Type	Description
body	**ImagePruner** schema	

Table 14.14. HTTP responses

HTTP code	Reponse body
200 - OK	**ImagePruner** schema
201 - Created	**ImagePruner** schema
401 - Unauthorized	Empty

14.2.3. /apis/imageregistry.operator.openshift.io/v1/imagepruners/{name}/status

Table 14.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the ImagePruner

HTTP method

GET

Description

read status of the specified ImagePruner

Table 14.16. HTTP responses

HTTP code	Reponse body
200 - OK	**ImagePruner** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified ImagePruner

Table 14.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 14.18. HTTP responses

HTTP code	Reponse body
200 - OK	**ImagePruner** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified ImagePruner

Table 14.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 14.20. Body parameters

Parameter	Type	Description
body	**ImagePruner** schema	

Table 14.21. HTTP responses

HTTP code	Reponse body
200 - OK	**ImagePruner** schema
201 - Created	**ImagePruner** schema
401 - Unauthorized	Empty

CHAPTER 15. INGRESSCONTROLLER [OPERATOR.OPENSHIFT.IO/V1]

Description

IngressController describes a managed ingress controller for the cluster. The controller can service OpenShift Route and Kubernetes Ingress resources.
When an IngressController is created, a new ingress controller deployment is created to allow external traffic to reach the services that expose Ingress or Route resources. Updating this resource may lead to disruption for public facing network connections as a new ingress controller revision may be rolled out.

https://kubernetes.io/docs/concepts/services-networking/ingress-controllers

Whenever possible, sensible defaults for the platform are used. See each field for more details.

Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

15.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds

Property	Type	Description
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	spec is the specification of the desired behavior of the IngressController.
status	**object**	status is the most recently observed status of the IngressController.

15.1.1. .spec

Description

spec is the specification of the desired behavior of the IngressController.

Type

object

Property	Type	Description
clientTLS	**object**	clientTLS specifies settings for requesting and verifying client certificates, which can be used to enable mutual TLS for edge-terminated and reencrypt routes.

Property	Type	Description
defaultCertificate	**object**	defaultCertificate is a reference to a secret containing the default certificate served by the ingress controller. When Routes don't specify their own certificate, defaultCertificate is used. The secret must contain the following keys and data: tls.crt: certificate file contents tls.key: key file contents If unset, a wildcard certificate is automatically generated and used. The certificate is valid for the ingress controller domain (and subdomains) and the generated certificate's CA will be automatically integrated with the cluster's trust store. If a wildcard certificate is used and shared by multiple HTTP/2 enabled routes (which implies ALPN) then clients (i.e., notably browsers) are at liberty to reuse open connections. This means a client can reuse a connection to another route and that is likely to fail. This behaviour is generally known as connection coalescing. The in-use certificate (whether generated or user-specified) will be automatically integrated with OpenShift's built-in OAuth server.

Property	Type	Description
domain	**string**	domain is a DNS name serviced by the ingress controller and is used to configure multiple features: * For the LoadBalancerService endpoint publishing strategy, domain is used to configure DNS records. See endpointPublishingStrategy. * When using a generated default certificate, the certificate will be valid for domain and its subdomains. See defaultCertificate. * The value is published to individual Route statuses so that end-users know where to target external DNS records. domain must be unique among all IngressControllers, and cannot be updated. If empty, defaults to ingress.config.openshift.io/cluster .spec.domain.

Property	Type	Description
endpointPublishingStrategy	**object**	endpointPublishingStrategy is used to publish the ingress controller endpoints to other networks, enable load balancer integrations, etc. If unset, the default is based on infrastructure.config.openshift.io/cluster .status.platform: AWS: LoadBalancerService (with External scope) Azure: LoadBalancerService (with External scope) GCP: LoadBalancerService (with External scope) IBMCloud: LoadBalancerService (with External scope) AlibabaCloud: LoadBalancerService (with External scope) Libvirt: HostNetwork Any other platform types (including None) default to HostNetwork. endpointPublishingStrategy cannot be updated.
httpCompression	**object**	httpCompression defines a policy for HTTP traffic compression. By default, there is no HTTP compression.

Property	Type	Description
httpEmptyRequestsPolicy	**string**	httpEmptyRequestsPolicy describes how HTTP connections should be handled if the connection times out before a request is received. Allowed values for this field are "Respond" and "Ignore". If the field is set to "Respond", the ingress controller sends an HTTP 400 or 408 response, logs the connection (if access logging is enabled), and counts the connection in the appropriate metrics. If the field is set to "Ignore", the ingress controller closes the connection without sending a response, logging the connection, or incrementing metrics. The default value is "Respond". Typically, these connections come from load balancers' health probes or Web browsers' speculative connections ("preconnect") and can be safely ignored. However, these requests may also be caused by network errors, and so setting this field to "Ignore" may impede detection and diagnosis of problems. In addition, these requests may be caused by port scans, in which case logging empty requests may aid in detecting intrusion attempts.

Property	Type	Description
httpErrorCodePages	**object**	httpErrorCodePages specifies a configmap with custom error pages. The administrator must create this configmap in the openshift-config namespace. This configmap should have keys in the format "error-page-<error code>.http", where <error code> is an HTTP error code. For example, "error-page-503.http" defines an error page for HTTP 503 responses. Currently only error pages for 503 and 404 responses can be customized. Each value in the configmap should be the full response, including HTTP headers. Eg- https://raw.githubusercontent.com/openshift/router/fadab45747a9b30cc3f0a4b41ad2871f95827a93/images/router/haproxy/conf/error-page-503.http If this field is empty, the ingress controller uses the default error pages.
httpHeaders	**object**	httpHeaders defines policy for HTTP headers. If this field is empty, the default values are used.
idleConnectionTerminationPolicy	**string**	idleConnectionTerminationPolicy maps directly to HAProxy's idle-close-on-response option and controls whether HAProxy keeps idle frontend connections open during a soft stop (router reload). Allowed values for this field are "Immediate" and "Deferred". The default value is "Immediate". When set to "Immediate", idle connections are closed immediately during router reloads. This ensures immediate propagation of route changes but may impact clients sensitive to connection resets. When set to "Deferred", HAProxy will maintain idle connections during a soft reload instead of

Property	Type	Description
		closing them immediately. These connections remain open until any of the following occurs: - A new request is received on the connection, in which case HAProxy handles it in the old process and closes the connection after sending the response. - HAProxy's **timeout http-keep-alive** duration expires (300 seconds in OpenShift's configuration, not configurable). - The client's keep-alive timeout expires, causing the client to close the connection. Setting Deferred can help prevent errors in clients or load balancers that do not properly handle connection resets. Additionally, this option allows you to retain the pre-2.4 HAProxy behaviour: in HAProxy version 2.2 (OpenShift versions < 4.14), maintaining idle connections during a soft reload was the default behaviour, but starting with HAProxy 2.4, the default changed to closing idle connections immediately. Important Consideration: - Using Deferred will result in temporary inconsistencies for the first request on each persistent connection after a route update and router reload. This request will be processed by the old HAProxy process using its old configuration. Subsequent requests will use the updated configuration. Operational Considerations: - Keeping idle connections open during reloads may lead to an accumulation of old HAProxy processes if connections remain idle for extended periods, especially in environments where frequent reloads occur.

Property	Type	Description
		- Consider monitoring the number of HAProxy processes in the router pods when Deferred is set. - You may need to enable or adjust the **ingress.operator.openshift.io/hard-stop-after** duration (configured via an annotation on the IngressController resource) in environments with frequent reloads to prevent resource exhaustion.
logging	**object**	logging defines parameters for what should be logged where. If this field is empty, operational logs are enabled but access logs are disabled.
namespaceSelector	**object**	namespaceSelector is used to filter the set of namespaces serviced by the ingress controller. This is useful for implementing shards. If unset, the default is no filtering.
nodePlacement	**object**	nodePlacement enables explicit control over the scheduling of the ingress controller. If unset, defaults are used. See NodePlacement for more details.

Property	Type	Description
replicas	**integer**	replicas is the desired number of ingress controller replicas. If unset, the default depends on the value of the defaultPlacement field in the cluster config.openshift.io/v1/ingresses status. The value of replicas is set based on the value of a chosen field in the Infrastructure CR. If defaultPlacement is set to ControlPlane, the chosen field will be controlPlaneTopology. If it is set to Workers the chosen field will be infrastructureTopology. Replicas will then be set to 1 or 2 based whether the chosen field's value is SingleReplica or HighlyAvailable, respectively. These defaults are subject to change.
routeAdmission	**object**	routeAdmission defines a policy for handling new route claims (for example, to allow or deny claims across namespaces). If empty, defaults will be applied. See specific routeAdmission fields for details about their defaults.
routeSelector	**object**	routeSelector is used to filter the set of Routes serviced by the ingress controller. This is useful for implementing shards. If unset, the default is no filtering.

Property	Type	Description
tlsSecurityProfile	**object**	tlsSecurityProfile specifies settings for TLS connections for ingresscontrollers. If unset, the default is based on the apiservers.config.openshift.io/cluster resource. Note that when using the Old, Intermediate, and Modern profile types, the effective profile configuration is subject to change between releases. For example, given a specification to use the Intermediate profile deployed on release X.Y.Z, an upgrade to release X.Y.Z+1 may cause a new profile configuration to be applied to the ingress controller, resulting in a rollout.
tuningOptions	**object**	tuningOptions defines parameters for adjusting the performance of ingress controller pods. All fields are optional and will use their respective defaults if not set. See specific tuningOptions fields for more details. Setting fields within tuningOptions is generally not recommended. The default values are suitable for most configurations.
unsupportedConfigOverrides	``	unsupportedConfigOverrides allows specifying unsupported configuration options. Its use is unsupported.

15.1.2. .spec.clientTLS

Description

clientTLS specifies settings for requesting and verifying client certificates, which can be used to enable mutual TLS for edge-terminated and reencrypt routes.

Type

object

Required

- **clientCA**
- **clientCertificatePolicy**

Property	Type	Description
allowedSubjectPatterns	**array (string)**	allowedSubjectPatterns specifies a list of regular expressions that should be matched against the distinguished name on a valid client certificate to filter requests. The regular expressions must use PCRE syntax. If this list is empty, no filtering is performed. If the list is nonempty, then at least one pattern must match a client certificate's distinguished name or else the ingress controller rejects the certificate and denies the connection.
clientCA	**object**	clientCA specifies a configmap containing the PEM-encoded CA certificate bundle that should be used to verify a client's certificate. The administrator must create this configmap in the openshift-config namespace.
clientCertificatePolicy	**string**	clientCertificatePolicy specifies whether the ingress controller requires clients to provide certificates. This field accepts the values "Required" or "Optional". Note that the ingress controller only checks client certificates for edge-terminated and reencrypt TLS routes; it cannot check certificates for cleartext HTTP or passthrough TLS routes.

15.1.3. .spec.clientTLS.clientCA

Description

clientCA specifies a configmap containing the PEM-encoded CA certificate bundle that should be used to verify a client's certificate. The administrator must create this configmap in the openshift-config namespace.

Type

object

Required

- **name**

Property	Type	Description
name	**string**	name is the metadata.name of the referenced config map

15.1.4. .spec.defaultCertificate

Description

defaultCertificate is a reference to a secret containing the default certificate served by the ingress controller. When Routes don't specify their own certificate, defaultCertificate is used.
The secret must contain the following keys and data:

```
tls.crt: certificate file contents
tls.key: key file contents
```

If unset, a wildcard certificate is automatically generated and used. The certificate is valid for the ingress controller domain (and subdomains) and the generated certificate's CA will be automatically integrated with the cluster's trust store.

If a wildcard certificate is used and shared by multiple HTTP/2 enabled routes (which implies ALPN) then clients (i.e., notably browsers) are at liberty to reuse open connections. This means a client can reuse a connection to another route and that is likely to fail. This behaviour is generally known as connection coalescing.

The in-use certificate (whether generated or user-specified) will be automatically integrated with OpenShift's built-in OAuth server.

Type

object

Property	Type	Description
name	**string**	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names

15.1.5. .spec.endpointPublishingStrategy

Description

endpointPublishingStrategy is used to publish the ingress controller endpoints to other networks, enable load balancer integrations, etc.

If unset, the default is based on infrastructure.config.openshift.io/cluster .status.platform:

```
AWS:          LoadBalancerService (with External scope)
Azure:        LoadBalancerService (with External scope)
GCP:          LoadBalancerService (with External scope)
IBMCloud:     LoadBalancerService (with External scope)
AlibabaCloud: LoadBalancerService (with External scope)
Libvirt:      HostNetwork
```

Any other platform types (including None) default to HostNetwork.

endpointPublishingStrategy cannot be updated.

Type

object

Required

- **type**

Property	Type	Description
hostNetwork	**object**	hostNetwork holds parameters for the HostNetwork endpoint publishing strategy. Present only if type is HostNetwork.
loadBalancer	**object**	loadBalancer holds parameters for the load balancer. Present only if type is LoadBalancerService.
nodePort	**object**	nodePort holds parameters for the NodePortService endpoint publishing strategy. Present only if type is NodePortService.
private	**object**	private holds parameters for the Private endpoint publishing strategy. Present only if type is Private.
type	**string**	type is the publishing strategy to use. Valid values are: * LoadBalancerService Publishes the ingress controller using a Kubernetes LoadBalancer Service. In this configuration, the ingress controller deployment uses container networking. A

Property	Type	Description
		LoadBalancer Service is created to publish the deployment. See: https://kubernetes.io/docs/concepts/services-networking/service/#loadbalancer If domain is set, a wildcard DNS record will be managed to point at the LoadBalancer Service's external name. DNS records are managed only in DNS zones defined by dns.config.openshift.io/cluster .spec.publicZone and .spec.privateZone. Wildcard DNS management is currently supported only on the AWS, Azure, and GCP platforms. * HostNetwork Publishes the ingress controller on node ports where the ingress controller is deployed. In this configuration, the ingress controller deployment uses host networking, bound to node ports 80 and 443. The user is responsible for configuring an external load balancer to publish the ingress controller via the node ports. * Private Does not publish the ingress controller. In this configuration, the ingress controller deployment uses container networking, and is not explicitly published. The user must manually publish the ingress controller. * NodePortService Publishes the ingress controller using a Kubernetes NodePort Service.

Property	Type	Description
		In this configuration, the ingress controller deployment uses container networking. A NodePort Service is created to publish the deployment. The specific node ports are dynamically allocated by OpenShift; however, to support static port allocations, user changes to the node port field of the managed NodePort Service will preserved.

15.1.6. .spec.endpointPublishingStrategy.hostNetwork

Description

hostNetwork holds parameters for the HostNetwork endpoint publishing strategy. Present only if type is HostNetwork.

Type

object

Property	Type	Description
httpPort	**integer**	httpPort is the port on the host which should be used to listen for HTTP requests. This field should be set when port 80 is already in use. The value should not coincide with the NodePort range of the cluster. When the value is 0 or is not specified it defaults to 80.
httpsPort	**integer**	httpsPort is the port on the host which should be used to listen for HTTPS requests. This field should be set when port 443 is already in use. The value should not coincide with the NodePort range of the cluster. When the value is 0 or is not specified it defaults to 443.

Property	Type	Description
protocol	**string**	protocol specifies whether the IngressController expects incoming connections to use plain TCP or whether the IngressController expects PROXY protocol. PROXY protocol can be used with load balancers that support it to communicate the source addresses of client connections when forwarding those connections to the IngressController. Using PROXY protocol enables the IngressController to report those source addresses instead of reporting the load balancer's address in HTTP headers and logs. Note that enabling PROXY protocol on the IngressController will cause connections to fail if you are not using a load balancer that uses PROXY protocol to forward connections to the IngressController. See http://www.haproxy.org/download/2.2/doc/proxy-protocol.txt for information about PROXY protocol. The following values are valid for this field: * The empty string. * "TCP". * "PROXY". The empty string specifies the default, which is TCP without PROXY protocol. Note that the default is subject to change.

Property	Type	Description
statsPort	**integer**	statsPort is the port on the host where the stats from the router are published. The value should not coincide with the NodePort range of the cluster. If an external load balancer is configured to forward connections to this IngressController, the load balancer should use this port for health checks. The load balancer can send HTTP probes on this port on a given node, with the path /healthz/ready to determine if the ingress controller is ready to receive traffic on the node. For proper operation the load balancer must not forward traffic to a node until the health check reports ready. The load balancer should also stop forwarding requests within a maximum of 45 seconds after /healthz/ready starts reporting not-ready. Probing every 5 to 10 seconds, with a 5-second timeout and with a threshold of two successful or failed requests to become healthy or unhealthy respectively, are well-tested values. When the value is 0 or is not specified it defaults to 1936.

15.1.7. .spec.endpointPublishingStrategy.loadBalancer

Description

loadBalancer holds parameters for the load balancer. Present only if type is LoadBalancerService.

Type

object

Required

- **dnsManagementPolicy**
- **scope**

Property	Type	Description

Property	Type	Description
allowedSourceRanges	``	allowedSourceRanges specifies an allowlist of IP address ranges to which access to the load balancer should be restricted. Each range must be specified using CIDR notation (e.g. "10.0.0.0/8" or "fd00::/8"). If no range is specified, "0.0.0.0/0" for IPv4 and "::/0" for IPv6 are used by default, which allows all source addresses. To facilitate migration from earlier versions of OpenShift that did not have the allowedSourceRanges field, you may set the service.beta.kubernetes.io/load-balancer-source-ranges annotation on the "router-<ingresscontroller name>" service in the "openshift-ingress" namespace, and this annotation will take effect if allowedSourceRanges is empty on OpenShift 4.12.
dnsManagementPolicy	**string**	dnsManagementPolicy indicates if the lifecycle of the wildcard DNS record associated with the load balancer service will be managed by the ingress operator. It defaults to Managed. Valid values are: Managed and Unmanaged.
providerParameters	**object**	providerParameters holds desired load balancer information specific to the underlying infrastructure provider. If empty, defaults will be applied. See specific providerParameters fields for details about their defaults.
scope	**string**	scope indicates the scope at which the load balancer is exposed. Possible values are "External" and "Internal".

15.1.8. .spec.endpointPublishingStrategy.loadBalancer.providerParameters

Description

providerParameters holds desired load balancer information specific to the underlying infrastructure provider.
If empty, defaults will be applied. See specific providerParameters fields for details about their defaults.

Type

object

Required

- **type**

Property	Type	Description
aws	**object**	aws provides configuration settings that are specific to AWS load balancers. If empty, defaults will be applied. See specific aws fields for details about their defaults.
gcp	**object**	gcp provides configuration settings that are specific to GCP load balancers. If empty, defaults will be applied. See specific gcp fields for details about their defaults.
ibm	**object**	ibm provides configuration settings that are specific to IBM Cloud load balancers. If empty, defaults will be applied. See specific ibm fields for details about their defaults.
openstack	**object**	openstack provides configuration settings that are specific to OpenStack load balancers. If empty, defaults will be applied. See specific openstack fields for details about their defaults.

Property	Type	Description
type	**string**	type is the underlying infrastructure provider for the load balancer. Allowed values are "AWS", "Azure", "BareMetal", "GCP", "IBM", "Nutanix", "OpenStack", and "VSphere".

15.1.9. .spec.endpointPublishingStrategy.loadBalancer.providerParameters.aws

Description

aws provides configuration settings that are specific to AWS load balancers.
If empty, defaults will be applied. See specific aws fields for details about their defaults.

Type

object

Required

- **type**

Property	Type	Description
classicLoadBalancer	**object**	classicLoadBalancerParameters holds configuration parameters for an AWS classic load balancer. Present only if type is Classic.
networkLoadBalancer	**object**	networkLoadBalancerParameters holds configuration parameters for an AWS network load balancer. Present only if type is NLB.

Property	Type	Description
type	**string**	type is the type of AWS load balancer to instantiate for an ingresscontroller. Valid values are: * "Classic": A Classic Load Balancer that makes routing decisions at either the transport layer (TCP/SSL) or the application layer (HTTP/HTTPS). See the following for additional details: https://docs.aws.amazon.com/AmazonECS/latest/developerguide/load-balancer-types.html#clb * "NLB": A Network Load Balancer that makes routing decisions at the transport layer (TCP/SSL). See the following for additional details: https://docs.aws.amazon.com/AmazonECS/latest/developerguide/load-balancer-types.html#nlb

15.1.10. .spec.endpointPublishingStrategy.loadBalancer.providerParameters.aws.classi

Description

classicLoadBalancerParameters holds configuration parameters for an AWS classic load balancer. Present only if type is Classic.

Type

object

Property	Type	Description

Property	Type	Description
connectionIdleTimeout	**string**	connectionIdleTimeout specifies the maximum time period that a connection may be idle before the load balancer closes the connection. The value must be parseable as a time duration value; see https://pkg.go.dev/time#ParseDuration. A nil or zero value means no opinion, in which case a default value is used. The default value for this field is 60s. This default is subject to change.
subnets	**object**	subnets specifies the subnets to which the load balancer will attach. The subnets may be specified by either their ID or name. The total number of subnets is limited to 10. In order for the load balancer to be provisioned with subnets, each subnet must exist, each subnet must be from a different availability zone, and the load balancer service must be recreated to pick up new values. When omitted from the spec, the subnets will be auto-discovered for each availability zone. Auto-discovered subnets are not reported in the status of the IngressController object.

15.1.11. .spec.endpointPublishingStrategy.loadBalancer.providerParameters.aws.classic

Description

subnets specifies the subnets to which the load balancer will attach. The subnets may be specified by either their ID or name. The total number of subnets is limited to 10.
In order for the load balancer to be provisioned with subnets, each subnet must exist, each subnet must be from a different availability zone, and the load balancer service must be recreated to pick up new values.

When omitted from the spec, the subnets will be auto-discovered for each availability zone. Auto-discovered subnets are not reported in the status of the IngressController object.

Type

object

Property	Type	Description
ids	**array (string)**	ids specifies a list of AWS subnets by subnet ID. Subnet IDs must start with "subnet-", consist only of alphanumeric characters, must be exactly 24 characters long, must be unique, and the total number of subnets specified by ids and names must not exceed 10.
names	**array (string)**	names specifies a list of AWS subnets by subnet name. Subnet names must not start with "subnet-", must not include commas, must be under 256 characters in length, must be unique, and the total number of subnets specified by ids and names must not exceed 10.

15.1.12. .spec.endpointPublishingStrategy.loadBalancer.providerParameters.aws.netwo

Description

networkLoadBalancerParameters holds configuration parameters for an AWS network load balancer. Present only if type is NLB.

Type

object

Property	Type	Description

Property	Type	Description
eipAllocations	**array (string)**	eipAllocations is a list of IDs for Elastic IP (EIP) addresses that are assigned to the Network Load Balancer. The following restrictions apply: eipAllocations can only be used with external scope, not internal. An EIP can be allocated to only a single IngressController. The number of EIP allocations must match the number of subnets that are used for the load balancer. Each EIP allocation must be unique. A maximum of 10 EIP allocations are permitted. See https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/elastic-ip-addresses-eip.html for general information about configuration, characteristics, and limitations of Elastic IP addresses.
subnets	**object**	subnets specifies the subnets to which the load balancer will attach. The subnets may be specified by either their ID or name. The total number of subnets is limited to 10. In order for the load balancer to be provisioned with subnets, each subnet must exist, each subnet must be from a different availability zone, and the load balancer service must be recreated to pick up new values. When omitted from the spec, the subnets will be auto-discovered for each availability zone. Auto-discovered subnets are not reported in the status of the IngressController object.

15.1.13. .spec.endpointPublishingStrategy.loadBalancer.providerParameters.aws.netwc

Description

subnets specifies the subnets to which the load balancer will attach. The subnets may be specified by either their ID or name. The total number of subnets is limited to 10.
In order for the load balancer to be provisioned with subnets, each subnet must exist, each subnet must be from a different availability zone, and the load balancer service must be recreated to pick up new values.

When omitted from the spec, the subnets will be auto-discovered for each availability zone. Auto-discovered subnets are not reported in the status of the IngressController object.

Type

object

Property	Type	Description
ids	**array (string)**	ids specifies a list of AWS subnets by subnet ID. Subnet IDs must start with "subnet-", consist only of alphanumeric characters, must be exactly 24 characters long, must be unique, and the total number of subnets specified by ids and names must not exceed 10.
names	**array (string)**	names specifies a list of AWS subnets by subnet name. Subnet names must not start with "subnet-", must not include commas, must be under 256 characters in length, must be unique, and the total number of subnets specified by ids and names must not exceed 10.

15.1.14. .spec.endpointPublishingStrategy.loadBalancer.providerParameters.gcp

Description

gcp provides configuration settings that are specific to GCP load balancers.
If empty, defaults will be applied. See specific gcp fields for details about their defaults.

Type

object

Property	Type	Description

Property	Type	Description
clientAccess	**string**	clientAccess describes how client access is restricted for internal load balancers. Valid values are: * "Global": Specifying an internal load balancer with Global client access allows clients from any region within the VPC to communicate with the load balancer. https://cloud.google.com/kubernetes-engine/docs/how-to/internal-load-balancing#global_access * "Local": Specifying an internal load balancer with Local client access means only clients within the same region (and VPC) as the GCP load balancer can communicate with the load balancer. Note that this is the default behavior. https://cloud.google.com/load-balancing/docs/internal#client_access

15.1.15. .spec.endpointPublishingStrategy.loadBalancer.providerParameters.ibm

Description

ibm provides configuration settings that are specific to IBM Cloud load balancers.
If empty, defaults will be applied. See specific ibm fields for details about their defaults.

Type

object

Property	Type	Description

Property	Type	Description
protocol	**string**	protocol specifies whether the load balancer uses PROXY protocol to forward connections to the IngressController. See "service.kubernetes.io/ibm-load-balancer-cloud-provider-enable-features: "proxy-protocol"" at https://cloud.ibm.com/docs/containers?topic=containers-vpc-lbaas PROXY protocol can be used with load balancers that support it to communicate the source addresses of client connections when forwarding those connections to the IngressController. Using PROXY protocol enables the IngressController to report those source addresses instead of reporting the load balancer's address in HTTP headers and logs. Note that enabling PROXY protocol on the IngressController will cause connections to fail if you are not using a load balancer that uses PROXY protocol to forward connections to the IngressController. See http://www.haproxy.org/download/2.2/doc/proxy-protocol.txt for information about PROXY protocol. Valid values for protocol are TCP, PROXY and omitted. When omitted, this means no opinion and the platform is left to choose a reasonable default, which is subject to change over time. The current default is TCP, without the proxy protocol enabled.

15.1.16. .spec.endpointPublishingStrategy.loadBalancer.providerParameters.openstack

Description

openstack provides configuration settings that are specific to OpenStack load balancers.
If empty, defaults will be applied. See specific openstack fields for details about their defaults.

Type

object

Property	Type	Description
floatingIP	**string**	floatingIP specifies the IP address that the load balancer will use. When not specified, an IP address will be assigned randomly by the OpenStack cloud provider. When specified, the floating IP has to be pre-created. If the specified value is not a floating IP or is already claimed, the OpenStack cloud provider won't be able to provision the load balancer. This field may only be used if the IngressController has External scope. This value must be a valid IPv4 or IPv6 address.

15.1.17. .spec.endpointPublishingStrategy.nodePort

Description

nodePort holds parameters for the NodePortService endpoint publishing strategy. Present only if type is NodePortService.

Type

object

Property	Type	Description

Property	Type	Description
protocol	**string**	protocol specifies whether the IngressController expects incoming connections to use plain TCP or whether the IngressController expects PROXY protocol. PROXY protocol can be used with load balancers that support it to communicate the source addresses of client connections when forwarding those connections to the IngressController. Using PROXY protocol enables the IngressController to report those source addresses instead of reporting the load balancer's address in HTTP headers and logs. Note that enabling PROXY protocol on the IngressController will cause connections to fail if you are not using a load balancer that uses PROXY protocol to forward connections to the IngressController. See http://www.haproxy.org/download/2.2/doc/proxy-protocol.txt for information about PROXY protocol. The following values are valid for this field: * The empty string. * "TCP". * "PROXY". The empty string specifies the default, which is TCP without PROXY protocol. Note that the default is subject to change.

15.1.18. .spec.endpointPublishingStrategy.private

Description

private holds parameters for the Private endpoint publishing strategy. Present only if type is Private.

Type

object

Property	Type	Description
protocol	**string**	protocol specifies whether the IngressController expects incoming connections to use plain TCP or whether the IngressController expects PROXY protocol. PROXY protocol can be used with load balancers that support it to communicate the source addresses of client connections when forwarding those connections to the IngressController. Using PROXY protocol enables the IngressController to report those source addresses instead of reporting the load balancer's address in HTTP headers and logs. Note that enabling PROXY protocol on the IngressController will cause connections to fail if you are not using a load balancer that uses PROXY protocol to forward connections to the IngressController. See http://www.haproxy.org/download/2.2/doc/proxy-protocol.txt for information about PROXY protocol. The following values are valid for this field: * The empty string. * "TCP". * "PROXY". The empty string specifies the default, which is TCP without PROXY protocol. Note that the default is subject to change.

15.1.19. .spec.httpCompression

Description

httpCompression defines a policy for HTTP traffic compression. By default, there is no HTTP compression.

Type

object

Property	Type	Description
mimeTypes	**array (string)**	mimeTypes is a list of MIME types that should have compression applied. This list can be empty, in which case the ingress controller does not apply compression. Note: Not all MIME types benefit from compression, but HAProxy will still use resources to try to compress if instructed to. Generally speaking, text (html, css, js, etc.) formats benefit from compression, but formats that are already compressed (image, audio, video, etc.) benefit little in exchange for the time and cpu spent on compressing again. See https://joehonton.medium.com/the-gzip-penalty-d31bd697f1a2

15.1.20. .spec.httpErrorCodePages

Description

httpErrorCodePages specifies a configmap with custom error pages. The administrator must create this configmap in the openshift-config namespace. This configmap should have keys in the format "error-page-<error code>.http", where <error code> is an HTTP error code. For example, "error-page-503.http" defines an error page for HTTP 503 responses. Currently only error pages for 503 and 404 responses can be customized. Each value in the configmap should be the full response, including HTTP headers. Eg- https://raw.githubusercontent.com/openshift/router/fadab45747a9b30cc3f0a4b41ad2871f95827a93/images/customresourcedefinition/error-page-503.http If this field is empty, the ingress controller uses the default error pages.

Type

object

Required

- **name**

Property	Type	Description
name	**string**	name is the metadata.name of the referenced config map

15.1.21. .spec.httpHeaders

Description

httpHeaders defines policy for HTTP headers.
If this field is empty, the default values are used.

Type

object

Property	Type	Description
actions	**object**	actions specifies options for modifying headers and their values. Note that this option only applies to cleartext HTTP connections and to secure HTTP connections for which the ingress controller terminates encryption (that is, edge-terminated or reencrypt connections). Headers cannot be modified for TLS passthrough connections. Setting the HSTS (**Strict-Transport-Security**) header is not supported via actions. **Strict-Transport-Security** may only be configured using the "haproxy.router.openshift.io/hsts_header" route annotation, and only in accordance with the policy specified in Ingress.Spec.RequiredHSTSPolicies. Any actions defined here are applied after any actions related to the following other fields: cache-control, spec.clientTLS, spec.httpHeaders.forwardedHeaderPolicy, spec.httpHeaders.uniqueId, and spec.httpHeaders.headerNameCaseAdjustments. In case of HTTP request headers, the actions specified in spec.httpHeaders.actions on the Route will be executed after the actions specified in the IngressController's spec.httpHeaders.actions field. In case of HTTP response headers, the actions specified in spec.httpHeaders.actions on the IngressController will be executed after the actions specified in the Route's spec.httpHeaders.actions field. Headers set using this API cannot be captured for use in access logs. The following header names are reserved and may not be modified via this API: Strict-Transport-Security, Proxy, Host,

Property	Type	Description
		Cookie, Set-Cookie. Note that the total size of all net added headers **after** interpolating dynamic values must not exceed the value of spec.tuningOptions.headerBufferMaxRewriteBytes on the IngressController. Please refer to the documentation for that API field for more details.
forwardedHeaderPolicy	**string**	forwardedHeaderPolicy specifies when and how the IngressController sets the Forwarded, X-Forwarded-For, X-Forwarded-Host, X-Forwarded-Port, X-Forwarded-Proto, and X-Forwarded-Proto-Version HTTP headers. The value may be one of the following: * "Append", which specifies that the IngressController appends the headers, preserving existing headers. * "Replace", which specifies that the IngressController sets the headers, replacing any existing Forwarded or X-Forwarded-* headers. * "IfNone", which specifies that the IngressController sets the headers if they are not already set. * "Never", which specifies that the IngressController never sets the headers, preserving any existing headers. By default, the policy is "Append".

Property	Type	Description
headerNameCaseAdjustments	``	headerNameCaseAdjustments specifies case adjustments that can be applied to HTTP header names. Each adjustment is specified as an HTTP header name with the desired capitalization. For example, specifying "X-Forwarded-For" indicates that the "x-forwarded-for" HTTP header should be adjusted to have the specified capitalization. These adjustments are only applied to cleartext, edge-terminated, and re-encrypt routes, and only when using HTTP/1. For request headers, these adjustments are applied only for routes that have the haproxy.router.openshift.io/h1-adjust-case=true annotation. For response headers, these adjustments are applied to all HTTP responses. If this field is empty, no request headers are adjusted.
uniqueId	**object**	uniqueId describes configuration for a custom HTTP header that the ingress controller should inject into incoming HTTP requests. Typically, this header is configured to have a value that is unique to the HTTP request. The header can be used by applications or included in access logs to facilitate tracing individual HTTP requests. If this field is empty, no such header is injected into requests.

15.1.22. .spec.httpHeaders.actions

Description

actions specifies options for modifying headers and their values. Note that this option only applies to cleartext HTTP connections and to secure HTTP connections for which the ingress controller

terminates encryption (that is, edge-terminated or reencrypt connections). Headers cannot be modified for TLS passthrough connections. Setting the HSTS (**Strict-Transport-Security**) header is not supported via actions. **Strict-Transport-Security** may only be configured using the "haproxy.router.openshift.io/hsts_header" route annotation, and only in accordance with the policy specified in Ingress.Spec.RequiredHSTSPolicies. Any actions defined here are applied after any actions related to the following other fields: cache-control, spec.clientTLS, spec.httpHeaders.forwardedHeaderPolicy, spec.httpHeaders.uniqueId, and spec.httpHeaders.headerNameCaseAdjustments. In case of HTTP request headers, the actions specified in spec.httpHeaders.actions on the Route will be executed after the actions specified in the IngressController's spec.httpHeaders.actions field. In case of HTTP response headers, the actions specified in spec.httpHeaders.actions on the IngressController will be executed after the actions specified in the Route's spec.httpHeaders.actions field. Headers set using this API cannot be captured for use in access logs. The following header names are reserved and may not be modified via this API: Strict-Transport-Security, Proxy, Host, Cookie, Set-Cookie. Note that the total size of all net added headers **after** interpolating dynamic values must not exceed the value of spec.tuningOptions.headerBufferMaxRewriteBytes on the IngressController. Please refer to the documentation for that API field for more details.

Type

object

Property	Type	Description
request	**array**	request is a list of HTTP request headers to modify. Actions defined here will modify the request headers of all requests passing through an ingress controller. These actions are applied to all Routes i.e. for all connections handled by the ingress controller defined within a cluster. IngressController actions for request headers will be executed before Route actions. Currently, actions may define to either **Set** or **Delete** headers values. Actions are applied in sequence as defined in this list. A maximum of 20 request header actions may be configured. Sample fetchers allowed are "req.hdr" and "ssl_c_der". Converters allowed are "lower" and "base64". Example header values: "%[req.hdr(X-target),lower]", "%{+Q}[ssl_c_der,base64]".
request[]	**object**	IngressControllerHTTPHeader specifies configuration for setting or deleting an HTTP header.

Property	Type	Description
response	**array**	response is a list of HTTP response headers to modify. Actions defined here will modify the response headers of all requests passing through an ingress controller. These actions are applied to all Routes i.e. for all connections handled by the ingress controller defined within a cluster. IngressController actions for response headers will be executed after Route actions. Currently, actions may define to either **Set** or **Delete** headers values. Actions are applied in sequence as defined in this list. A maximum of 20 response header actions may be configured. Sample fetchers allowed are "res.hdr" and "ssl_c_der". Converters allowed are "lower" and "base64". Example header values: "%[res.hdr(X-target),lower]", "%{+Q}[ssl_c_der,base64]".
response[]	**object**	IngressControllerHTTPHeader specifies configuration for setting or deleting an HTTP header.

15.1.23. .spec.httpHeaders.actions.request

Description

request is a list of HTTP request headers to modify. Actions defined here will modify the request headers of all requests passing through an ingress controller. These actions are applied to all Routes i.e. for all connections handled by the ingress controller defined within a cluster. IngressController actions for request headers will be executed before Route actions. Currently, actions may define to either **Set** or **Delete** headers values. Actions are applied in sequence as defined in this list. A maximum of 20 request header actions may be configured. Sample fetchers allowed are "req.hdr" and "ssl_c_der". Converters allowed are "lower" and "base64". Example header values: "%[req.hdr(X-target),lower]", "%{+Q}[ssl_c_der,base64]".

Type

array

15.1.24. .spec.httpHeaders.actions.request[]

Description

IngressControllerHTTPHeader specifies configuration for setting or deleting an HTTP header.

Type

object

Required

- **action**
- **name**

Property	Type	Description
action	**object**	action specifies actions to perform on headers, such as setting or deleting headers.
name	**string**	name specifies the name of a header on which to perform an action. Its value must be a valid HTTP header name as defined in RFC 2616 section 4.2. The name must consist only of alphanumeric and the following special characters, "-!#$%&'*+.^_`". The following header names are reserved and may not be modified via this API: Strict-Transport-Security, Proxy, Host, Cookie, Set-Cookie. It must be no more than 255 characters in length. Header name must be unique.

15.1.25. .spec.httpHeaders.actions.request[].action

Description

action specifies actions to perform on headers, such as setting or deleting headers.

Type

object

Required

- **type**

Property	Type	Description
set	**object**	set specifies how the HTTP header should be set. This field is required when type is Set and forbidden otherwise.

Property	Type	Description
type	**string**	type defines the type of the action to be applied on the header. Possible values are Set or Delete. Set allows you to set HTTP request and response headers. Delete allows you to delete HTTP request and response headers.

15.1.26. .spec.httpHeaders.actions.request[].action.set

Description

set specifies how the HTTP header should be set. This field is required when type is Set and forbidden otherwise.

Type

object

Required

- **value**

Property	Type	Description
value	**string**	value specifies a header value. Dynamic values can be added. The value will be interpreted as an HAProxy format string as defined in http://cbonte.github.io/haproxy-dconv/2.6/configuration.html#8.2.6 and may use HAProxy's %[] syntax and otherwise must be a valid HTTP header value as defined in https://datatracker.ietf.org/doc/html/rfc7230#section-3.2. The value of this field must be no more than 16384 characters in length. Note that the total size of all net added headers **after** interpolating dynamic values must not exceed the value of spec.tuningOptions.headerBufferMaxRewriteBytes on the IngressController.

15.1.27. .spec.httpHeaders.actions.response

Description

response is a list of HTTP response headers to modify. Actions defined here will modify the response headers of all requests passing through an ingress controller. These actions are applied to all Routes i.e. for all connections handled by the ingress controller defined within a cluster. IngressController actions for response headers will be executed after Route actions. Currently, actions may define to either **Set** or **Delete** headers values. Actions are applied in sequence as defined in this list. A maximum of 20 response header actions may be configured. Sample fetchers allowed are "res.hdr" and "ssl_c_der". Converters allowed are "lower" and "base64". Example header values: "%[res.hdr(X-target),lower]", "%{+Q}[ssl_c_der,base64]".

Type

array

15.1.28. .spec.httpHeaders.actions.response[]

Description

IngressControllerHTTPHeader specifies configuration for setting or deleting an HTTP header.

Type

object

Required

- **action**
- **name**

Property	Type	Description
action	**object**	action specifies actions to perform on headers, such as setting or deleting headers.
name	**string**	name specifies the name of a header on which to perform an action. Its value must be a valid HTTP header name as defined in RFC 2616 section 4.2. The name must consist only of alphanumeric and the following special characters, "-!#$%&'*+.^_`". The following header names are reserved and may not be modified via this API: Strict-Transport-Security, Proxy, Host, Cookie, Set-Cookie. It must be no more than 255 characters in length. Header name must be unique.

15.1.29. .spec.httpHeaders.actions.response[].action

Description

action specifies actions to perform on headers, such as setting or deleting headers.

Type

object

Required

- **type**

Property	Type	Description
set	**object**	set specifies how the HTTP header should be set. This field is required when type is Set and forbidden otherwise.
type	**string**	type defines the type of the action to be applied on the header. Possible values are Set or Delete. Set allows you to set HTTP request and response headers. Delete allows you to delete HTTP request and response headers.

15.1.30. .spec.httpHeaders.actions.response[].action.set

Description

set specifies how the HTTP header should be set. This field is required when type is Set and forbidden otherwise.

Type

object

Required

- **value**

Property	Type	Description

Property	Type	Description
value	**string**	value specifies a header value. Dynamic values can be added. The value will be interpreted as an HAProxy format string as defined in http://cbonte.github.io/haproxy-dconv/2.6/configuration.html#8.2.6 and may use HAProxy's %[] syntax and otherwise must be a valid HTTP header value as defined in https://datatracker.ietf.org/doc/html/rfc7230#section-3.2. The value of this field must be no more than 16384 characters in length. Note that the total size of all net added headers **after** interpolating dynamic values must not exceed the value of spec.tuningOptions.headerBufferMaxRewriteBytes on the IngressController.

15.1.31. .spec.httpHeaders.uniqueId

Description

uniqueId describes configuration for a custom HTTP header that the ingress controller should inject into incoming HTTP requests. Typically, this header is configured to have a value that is unique to the HTTP request. The header can be used by applications or included in access logs to facilitate tracing individual HTTP requests.
If this field is empty, no such header is injected into requests.

Type

object

Property	Type	Description

Property	Type	Description
format	**string**	format specifies the format for the injected HTTP header's value. This field has no effect unless name is specified. For the HAProxy-based ingress controller implementation, this format uses the same syntax as the HTTP log format. If the field is empty, the default value is "%{+X}o\\ %ci:%cp_%fi:%fp_%Ts_%rt:%pid"; see the corresponding HAProxy documentation: http://cbonte.github.io/haproxy-dconv/2.0/configuration.html#8.2.3
name	**string**	name specifies the name of the HTTP header (for example, "unique-id") that the ingress controller should inject into HTTP requests. The field's value must be a valid HTTP header name as defined in RFC 2616 section 4.2. If the field is empty, no header is injected.

15.1.32. .spec.logging

Description

logging defines parameters for what should be logged where. If this field is empty, operational logs are enabled but access logs are disabled.

Type

object

Property	Type	Description
access	**object**	access describes how the client requests should be logged. If this field is empty, access logging is disabled.

15.1.33. .spec.logging.access

Description

access describes how the client requests should be logged.
If this field is empty, access logging is disabled.

Type

object

Required

- **destination**

Property	Type	Description
destination	**object**	destination is where access logs go.
httpCaptureCookies	``	httpCaptureCookies specifies HTTP cookies that should be captured in access logs. If this field is empty, no cookies are captured.
httpCaptureHeaders	**object**	httpCaptureHeaders defines HTTP headers that should be captured in access logs. If this field is empty, no headers are captured. Note that this option only applies to cleartext HTTP connections and to secure HTTP connections for which the ingress controller terminates encryption (that is, edge-terminated or reencrypt connections). Headers cannot be captured for TLS passthrough connections.

Property	Type	Description
httpLogFormat	**string**	httpLogFormat specifies the format of the log message for an HTTP request. If this field is empty, log messages use the implementation's default HTTP log format. For HAProxy's default HTTP log format, see the HAProxy documentation: http://cbonte.github.io/haproxy-dconv/2.0/configuration.html#8.2.3 Note that this format only applies to cleartext HTTP connections and to secure HTTP connections for which the ingress controller terminates encryption (that is, edge-terminated or reencrypt connections). It does not affect the log format for TLS passthrough connections.
logEmptyRequests	**string**	logEmptyRequests specifies how connections on which no request is received should be logged. Typically, these empty requests come from load balancers' health probes or Web browsers' speculative connections ("preconnect"), in which case logging these requests may be undesirable. However, these requests may also be caused by network errors, in which case logging empty requests may be useful for diagnosing the errors. In addition, these requests may be caused by port scans, in which case logging empty requests may aid in detecting intrusion attempts. Allowed values for this field are "Log" and "Ignore". The default value is "Log".

15.1.34. .spec.logging.access.destination

Description

destination is where access logs go.

Type

object

Required

- **type**

Property	Type	Description
container	**object**	container holds parameters for the Container logging destination. Present only if type is Container.
syslog	**object**	syslog holds parameters for a syslog endpoint. Present only if type is Syslog.
type	**string**	type is the type of destination for logs. It must be one of the following: * Container The ingress operator configures the sidecar container named "logs" on the ingress controller pod and configures the ingress controller to write logs to the sidecar. The logs are then available as container logs. The expectation is that the administrator configures a custom logging solution that reads logs from this sidecar. Note that using container logs means that logs may be dropped if the rate of logs exceeds the container runtime's or the custom logging solution's capacity. * Syslog Logs are sent to a syslog endpoint. The administrator must specify an endpoint that can receive syslog messages. The expectation is that the administrator has configured a custom syslog instance.

15.1.35. .spec.logging.access.destination.container

Description

container holds parameters for the Container logging destination. Present only if type is Container.

Type

object

Property	Type	Description
maxLength	**integer**	maxLength is the maximum length of the log message. Valid values are integers in the range 480 to 8192, inclusive. When omitted, the default value is 1024.

15.1.36. .spec.logging.access.destination.syslog

Description

syslog holds parameters for a syslog endpoint. Present only if type is Syslog.

Type

object

Required

- **address**
- **port**

Property	Type	Description
address	**string**	address is the IP address of the syslog endpoint that receives log messages.
facility	**string**	facility specifies the syslog facility of log messages. If this field is empty, the facility is "local1".
maxLength	**integer**	maxLength is the maximum length of the log message. Valid values are integers in the range 480 to 4096, inclusive. When omitted, the default value is 1024.

Property	Type	Description
port	**integer**	port is the UDP port number of the syslog endpoint that receives log messages.

15.1.37. .spec.logging.access.httpCaptureHeaders

Description

httpCaptureHeaders defines HTTP headers that should be captured in access logs. If this field is empty, no headers are captured.
Note that this option only applies to cleartext HTTP connections and to secure HTTP connections for which the ingress controller terminates encryption (that is, edge-terminated or reencrypt connections). Headers cannot be captured for TLS passthrough connections.

Type

object

Property	Type	Description
request	``	request specifies which HTTP request headers to capture. If this field is empty, no request headers are captured.
response	``	response specifies which HTTP response headers to capture. If this field is empty, no response headers are captured.

15.1.38. .spec.namespaceSelector

Description

namespaceSelector is used to filter the set of namespaces serviced by the ingress controller. This is useful for implementing shards.
If unset, the default is no filtering.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.

Property	Type	Description
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

15.1.39. .spec.namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

15.1.40. .spec.namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

15.1.41. .spec.nodePlacement

Description

nodePlacement enables explicit control over the scheduling of the ingress controller. If unset, defaults are used. See NodePlacement for more details.

Type

object

Property	Type	Description

Property	Type	Description
nodeSelector	**object**	nodeSelector is the node selector applied to ingress controller deployments. If set, the specified selector is used and replaces the default. If unset, the default depends on the value of the defaultPlacement field in the cluster config.openshift.io/v1/ingresses status. When defaultPlacement is Workers, the default is: kubernetes.io/os: linux node-role.kubernetes.io/worker: '' When defaultPlacement is ControlPlane, the default is: kubernetes.io/os: linux node-role.kubernetes.io/master: '' These defaults are subject to change. Note that using nodeSelector.matchExpressions is not supported. Only nodeSelector.matchLabels may be used. This is a limitation of the Kubernetes API: the pod spec does not allow complex expressions for node selectors.
tolerations	**array**	tolerations is a list of tolerations applied to ingress controller deployments. The default is an empty list. See https://kubernetes.io/docs/concepts/configuration/taint-and-toleration/

Property	Type	Description
tolerations[]	**object**	The pod this Toleration is attached to tolerates any taint that matches the triple <key,value,effect> using the matching operator <operator>.

15.1.42. .spec.nodePlacement.nodeSelector

Description

nodeSelector is the node selector applied to ingress controller deployments.
If set, the specified selector is used and replaces the default.

If unset, the default depends on the value of the defaultPlacement field in the cluster config.openshift.io/v1/ingresses status.

When defaultPlacement is Workers, the default is:

```
kubernetes.io/os: linux
node-role.kubernetes.io/worker: ''
```

When defaultPlacement is ControlPlane, the default is:

```
kubernetes.io/os: linux
node-role.kubernetes.io/master: ''
```

These defaults are subject to change.

Note that using nodeSelector.matchExpressions is not supported. Only nodeSelector.matchLabels may be used. This is a limitation of the Kubernetes API: the pod spec does not allow complex expressions for node selectors.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Property	Type	Description
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

15.1.43. .spec.nodePlacement.nodeSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

15.1.44. .spec.nodePlacement.nodeSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

Property	Type	Description
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

15.1.45. .spec.nodePlacement.tolerations

Description

tolerations is a list of tolerations applied to ingress controller deployments.
The default is an empty list.

See https://kubernetes.io/docs/concepts/configuration/taint-and-toleration/

Type

array

15.1.46. .spec.nodePlacement.tolerations[]

Description

The pod this Toleration is attached to tolerates any taint that matches the triple <key,value,effect> using the matching operator <operator>.

Type

object

Property	Type	Description
effect	**string**	Effect indicates the taint effect to match. Empty means match all taint effects. When specified, allowed values are NoSchedule, PreferNoSchedule and NoExecute.
key	**string**	Key is the taint key that the toleration applies to. Empty means match all taint keys. If the key is empty, operator must be Exists; this combination means to match all values and all keys.

Property	Type	Description
operator	**string**	Operator represents a key's relationship to the value. Valid operators are Exists and Equal. Defaults to Equal. Exists is equivalent to wildcard for value, so that a pod can tolerate all taints of a particular category.
tolerationSeconds	**integer**	TolerationSeconds represents the period of time the toleration (which must be of effect NoExecute, otherwise this field is ignored) tolerates the taint. By default, it is not set, which means tolerate the taint forever (do not evict). Zero and negative values will be treated as 0 (evict immediately) by the system.
value	**string**	Value is the taint value the toleration matches to. If the operator is Exists, the value should be empty, otherwise just a regular string.

15.1.47. .spec.routeAdmission

Description

routeAdmission defines a policy for handling new route claims (for example, to allow or deny claims across namespaces).
If empty, defaults will be applied. See specific routeAdmission fields for details about their defaults.

Type

object

Property	Type	Description

Property	Type	Description
namespaceOwnership	**string**	namespaceOwnership describes how host name claims across namespaces should be handled. Value must be one of: - Strict: Do not allow routes in different namespaces to claim the same host. - InterNamespaceAllowed: Allow routes to claim different paths of the same host name across namespaces. If empty, the default is Strict.
wildcardPolicy	**string**	wildcardPolicy describes how routes with wildcard policies should be handled for the ingress controller. WildcardPolicy controls use of routes [1] exposed by the ingress controller based on the route's wildcard policy. [1] https://github.com/openshift/api/blob/master/route/v1/types.go Note: Updating WildcardPolicy from WildcardsAllowed to WildcardsDisallowed will cause admitted routes with a wildcard policy of Subdomain to stop working. These routes must be updated to a wildcard policy of None to be readmitted by the ingress controller. WildcardPolicy supports WildcardsAllowed and WildcardsDisallowed values. If empty, defaults to "WildcardsDisallowed".

15.1.48. .spec.routeSelector

Description

routeSelector is used to filter the set of Routes serviced by the ingress controller. This is useful for implementing shards.
If unset, the default is no filtering.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

15.1.49. .spec.routeSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

15.1.50. .spec.routeSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.

Property	Type	Description
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

15.1.51. .spec.tlsSecurityProfile

Description

tlsSecurityProfile specifies settings for TLS connections for ingresscontrollers.
If unset, the default is based on the apiservers.config.openshift.io/cluster resource.

Note that when using the Old, Intermediate, and Modern profile types, the effective profile configuration is subject to change between releases. For example, given a specification to use the Intermediate profile deployed on release X.Y.Z, an upgrade to release X.Y.Z+1 may cause a new profile configuration to be applied to the ingress controller, resulting in a rollout.

Type

object

Property	Type	Description

Property	Type	Description
custom	``	custom is a user-defined TLS security profile. Be extremely careful using a custom profile as invalid configurations can be catastrophic. An example custom profile looks like this: ciphers: - ECDHE-ECDSA-CHACHA20-POLY1305 - ECDHE-RSA-CHACHA20-POLY1305 - ECDHE-RSA-AES128-GCM-SHA256 - ECDHE-ECDSA-AES128-GCM-SHA256 minTLSVersion: VersionTLS11

Property	Type	Description
intermediate	``	intermediate is a TLS security profile based on: https://wiki.mozilla.org/Security/Server_Side_TLS#Intermediate_compatibility_.28recommended.29 and looks like this (yaml): ciphers: - TLS_AES_128_GCM_SHA256 - TLS_AES_256_GCM_SHA384 - TLS_CHACHA20_POLY1305_SHA256 - ECDHE-ECDSA-AES128-GCM-SHA256 - ECDHE-RSA-AES128-GCM-SHA256 - ECDHE-ECDSA-AES256-GCM-SHA384 - ECDHE-RSA-AES256-GCM-SHA384 - ECDHE-ECDSA-CHACHA20-POLY1305 - ECDHE-RSA-CHACHA20-POLY1305 - DHE-RSA-AES128-GCM-SHA256 - DHE-RSA-AES256-GCM-SHA384 minTLSVersion: VersionTLS12

Property	Type	Description
modern	``	modern is a TLS security profile based on: https://wiki.mozilla.org/Security/Server_Side_TLS#Modern_compatibility and looks like this (yaml): ciphers: - TLS_AES_128_GCM_SHA256 - TLS_AES_256_GCM_SHA384 - TLS_CHACHA20_POLY1305_SHA256 minTLSVersion: VersionTLS13
old	``	old is a TLS security profile based on: https://wiki.mozilla.org/Security/Server_Side_TLS#Old_backward_compatibility and looks like this (yaml): ciphers: - TLS_AES_128_GCM_SHA256 - TLS_AES_256_GCM_SHA384 - TLS_CHACHA20_POLY1305_SHA256 - ECDHE-ECDSA-AES128-GCM-SHA256 - ECDHE-RSA-AES128-GCM-SHA256 - ECDHE-ECDSA-AES256-GCM-SHA384 - ECDHE-RSA-AES256-GCM-SHA384 - ECDHE-ECDSA-CHACHA20-POLY1305

Property	Type	Description
		- ECDHE-RSA-CHACHA20-POLY1305 - DHE-RSA-AES128-GCM-SHA256 - DHE-RSA-AES256-GCM-SHA384 - DHE-RSA-CHACHA20-POLY1305 - ECDHE-ECDSA-AES128-SHA256 - ECDHE-RSA-AES128-SHA256 - ECDHE-ECDSA-AES128-SHA - ECDHE-RSA-AES128-SHA - ECDHE-ECDSA-AES256-SHA384 - ECDHE-RSA-AES256-SHA384 - ECDHE-ECDSA-AES256-SHA - ECDHE-RSA-AES256-SHA - DHE-RSA-AES128-SHA256 - DHE-RSA-AES256-SHA256 - AES128-GCM-SHA256 - AES256-GCM-SHA384 - AES128-SHA256 - AES256-SHA256 - AES128-SHA - AES256-SHA - DES-CBC3-SHA minTLSVersion: VersionTLS10

Property	Type	Description
type	**string**	type is one of Old, Intermediate, Modern or Custom. Custom provides the ability to specify individual TLS security profile parameters. Old, Intermediate and Modern are TLS security profiles based on: https://wiki.mozilla.org/Security/Server_Side_TLS#Recommended_configurations The profiles are intent based, so they may change over time as new ciphers are developed and existing ciphers are found to be insecure. Depending on precisely which ciphers are available to a process, the list may be reduced. Note that the Modern profile is currently not supported because it is not yet well adopted by common software libraries.

15.1.52. .spec.tuningOptions

Description

tuningOptions defines parameters for adjusting the performance of ingress controller pods. All fields are optional and will use their respective defaults if not set. See specific tuningOptions fields for more details.
Setting fields within tuningOptions is generally not recommended. The default values are suitable for most configurations.

Type

object

Property	Type	Description
clientFinTimeout	**string**	clientFinTimeout defines how long a connection will be held open while waiting for the client response to the server/backend closing the connection. If unset, the default timeout is 1s

Property	Type	Description
clientTimeout	**string**	clientTimeout defines how long a connection will be held open while waiting for a client response. If unset, the default timeout is 30s
connectTimeout	**string**	connectTimeout defines the maximum time to wait for a connection attempt to a server/backend to succeed. This field expects an unsigned duration string of decimal numbers, each with optional fraction and a unit suffix, e.g. "300ms", "1.5h" or "2h45m". Valid time units are "ns", "us" (or "µs" U+00B5 or "μs" U+03BC), "ms", "s", "m", "h". When omitted, this means the user has no opinion and the platform is left to choose a reasonable default. This default is subject to change over time. The current default is 5s.
headerBufferBytes	**integer**	headerBufferBytes describes how much memory should be reserved (in bytes) for IngressController connection sessions. Note that this value must be at least 16384 if HTTP/2 is enabled for the IngressController (https://tools.ietf.org/html/rfc7540). If this field is empty, the IngressController will use a default value of 32768 bytes. Setting this field is generally not recommended as headerBufferBytes values that are too small may break the IngressController and headerBufferBytes values that are too large could cause the IngressController to use significantly more memory than necessary.

Property	Type	Description
headerBufferMaxRewriteBytes	**integer**	headerBufferMaxRewriteBytes describes how much memory should be reserved (in bytes) from headerBufferBytes for HTTP header rewriting and appending for IngressController connection sessions. Note that incoming HTTP requests will be limited to (headerBufferBytes - headerBufferMaxRewriteBytes) bytes, meaning headerBufferBytes must be greater than headerBufferMaxRewriteBytes. If this field is empty, the IngressController will use a default value of 8192 bytes. Setting this field is generally not recommended as headerBufferMaxRewriteBytes values that are too small may break the IngressController and headerBufferMaxRewriteBytes values that are too large could cause the IngressController to use significantly more memory than necessary.

Property	Type	Description
healthCheckInterval	**string**	healthCheckInterval defines how long the router waits between two consecutive health checks on its configured backends. This value is applied globally as a default for all routes, but may be overridden per-route by the route annotation "router.openshift.io/haproxy.health.check.interval". Expects an unsigned duration string of decimal numbers, each with optional fraction and a unit suffix, eg "300ms", "1.5h" or "2h45m". Valid time units are "ns", "us" (or "µs" U+00B5 or "μs" U+03BC), "ms", "s", "m", "h". Setting this to less than 5s can cause excess traffic due to too frequent TCP health checks and accompanying SYN packet storms. Alternatively, setting this too high can result in increased latency, due to backend servers that are no longer available, but haven't yet been detected as such. An empty or zero healthCheckInterval means no opinion and IngressController chooses a default, which is subject to change over time. Currently the default healthCheckInterval value is 5s. Currently the minimum allowed value is 1s and the maximum allowed value is 2147483647ms (24.85 days). Both are subject to change over time.
maxConnections	**integer**	maxConnections defines the maximum number of simultaneous connections that can be established per HAProxy process. Increasing this value allows each ingress controller pod to handle more connections but at the cost of additional system resources being consumed.

Property	Type	Description
		Permitted values are: empty, 0, -1, and the range 2000-2000000. If this field is empty or 0, the IngressController will use the default value of 50000, but the default is subject to change in future releases. If the value is -1 then HAProxy will dynamically compute a maximum value based on the available ulimits in the running container. Selecting -1 (i.e., auto) will result in a large value being computed (~520000 on OpenShift >=4.10 clusters) and therefore each HAProxy process will incur significant memory usage compared to the current default of 50000. Setting a value that is greater than the current operating system limit will prevent the HAProxy process from starting. If you choose a discrete value (e.g., 750000) and the router pod is migrated to a new node, there's no guarantee that that new node has identical ulimits configured. In such a scenario the pod would fail to start. If you have nodes with different ulimits configured (e.g., different tuned profiles) and you choose a discrete value then the guidance is to use -1 and let the value be computed dynamically at runtime. You can monitor memory usage for router containers with the following metric: 'container_memory_working_set_bytes{container="router",namespace="openshift-ingress"}'. You can monitor memory usage of individual HAProxy processes in router containers with the following metric: 'container_memory_working_set_bytes{container="router",namespace="openshift-

reloadInterval Property	string Type	Description
		reloadInterval defines the minimum interval at which the router is allowed to reload to accept new changes. Increasing this value can prevent the accumulation of HAProxy processes, depending on the scenario. Increasing this interval can also lessen load imbalance on a backend's servers when using the roundrobin balancing algorithm. Alternatively, decreasing this value may decrease latency since updates to HAProxy's configuration can take effect more quickly. The value must be a time duration value; see https://pkg.go.dev/time#ParseDuration. Currently, the minimum value allowed is 1s, and the maximum allowed value is 120s. Minimum and maximum allowed values may change in future versions of OpenShift. Note that if a duration outside of these bounds is provided, the value of reloadInterval will be capped/floored and not rejected (e.g. a duration of over 120s will be capped to 120s; the IngressController will not reject and replace this disallowed value with the default). A zero value for reloadInterval tells the IngressController to choose the default, which is currently 5s and subject to change without notice. This field expects an unsigned duration string of decimal numbers, each with optional fraction and a unit suffix, e.g. "300ms", "1.5h" or "2h45m". Valid time units are "ns", "us" (or "µs" U+00B5 or "μs" U+03BC), "ms", "s", "m", "h". Note: Setting a value significantly larger than the default of 5s can cause latency in observing updates to routes and their endpoints. HAProxy's configuration will be reloaded less

Property	Type	Description
		frequently, and newly created routes will not be served until the subsequent reload.
serverFinTimeout	**string**	serverFinTimeout defines how long a connection will be held open while waiting for the server/backend response to the client closing the connection. If unset, the default timeout is 1s
serverTimeout	**string**	serverTimeout defines how long a connection will be held open while waiting for a server/backend response. If unset, the default timeout is 30s
threadCount	**integer**	threadCount defines the number of threads created per HAProxy process. Creating more threads allows each ingress controller pod to handle more connections, at the cost of more system resources being used. HAProxy currently supports up to 64 threads. If this field is empty, the IngressController will use the default value. The current default is 4 threads, but this may change in future releases. Setting this field is generally not recommended. Increasing the number of HAProxy threads allows ingress controller pods to utilize more CPU time under load, potentially starving other pods if set too high. Reducing the number of threads may cause the ingress controller to perform poorly.

Property	Type	Description
tlsInspectDelay	**string**	tlsInspectDelay defines how long the router can hold data to find a matching route. Setting this too short can cause the router to fall back to the default certificate for edge-terminated or reencrypt routes even when a better matching certificate could be used. If unset, the default inspect delay is 5s
tunnelTimeout	**string**	tunnelTimeout defines how long a tunnel connection (including websockets) will be held open while the tunnel is idle. If unset, the default timeout is 1h

15.1.53. .status

Description

status is the most recently observed status of the IngressController.

Type

object

Property	Type	Description
availableReplicas	**integer**	availableReplicas is number of observed available replicas according to the ingress controller deployment.

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status. Available means the ingress controller deployment is available and servicing route and ingress resources (i.e, .status.availableReplicas equals .spec.replicas) There are additional conditions which indicate the status of other ingress controller features and capabilities. * LoadBalancerManaged - True if the following conditions are met: * The endpoint publishing strategy requires a service load balancer. - False if any of those conditions are unsatisfied. * LoadBalancerReady - True if the following conditions are met: * A load balancer is managed. * The load balancer is ready. - False if any of those conditions are unsatisfied. * DNSManaged - True if the following conditions are met: * The endpoint publishing strategy and platform support DNS. * The ingress controller domain is set. * dns.config.openshift.io/cluster configures DNS zones. - False if any of those conditions are unsatisfied. * DNSReady - True if the following conditions are met: * DNS is managed. * DNS records have been successfully created. - False if any of those conditions are unsatisfied.
conditions[]	**object**	OperatorCondition is just the standard condition fields.
domain	**string**	domain is the actual domain in use.

Property	Type	Description
endpointPublishingStrategy	**object**	endpointPublishingStrategy is the actual strategy in use.
namespaceSelector	**object**	namespaceSelector is the actual namespaceSelector in use.
observedGeneration	**integer**	observedGeneration is the most recent generation observed.
routeSelector	**object**	routeSelector is the actual routeSelector in use.
selector	**string**	selector is a label selector, in string format, for ingress controller pods corresponding to the IngressController. The number of matching pods should equal the value of availableReplicas.
tlsProfile	**object**	tlsProfile is the TLS connection configuration that is in effect.

15.1.54. .status.conditions

Description

conditions is a list of conditions and their status.
Available means the ingress controller deployment is available and servicing route and ingress resources (i.e, .status.availableReplicas equals .spec.replicas)

There are additional conditions which indicate the status of other ingress controller features and capabilities.

- LoadBalancerManaged
  - True if the following conditions are met:
- The endpoint publishing strategy requires a service load balancer.
  - False if any of those conditions are unsatisfied.
- LoadBalancerReady
  - True if the following conditions are met:
- A load balancer is managed.
- The load balancer is ready.

- False if any of those conditions are unsatisfied.

- DNSManaged
  - True if the following conditions are met:
- The endpoint publishing strategy and platform support DNS.
- The ingress controller domain is set.
- dns.config.openshift.io/cluster configures DNS zones.
  - False if any of those conditions are unsatisfied.
- DNSReady
  - True if the following conditions are met:
- DNS is managed.
- DNS records have been successfully created.
  - False if any of those conditions are unsatisfied.

Type

array

15.1.55. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	

Property	Type	Description
reason	**string**	
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

15.1.56. .status.endpointPublishingStrategy

Description

endpointPublishingStrategy is the actual strategy in use.

Type

object

Required

- **type**

Property	Type	Description
hostNetwork	**object**	hostNetwork holds parameters for the HostNetwork endpoint publishing strategy. Present only if type is HostNetwork.
loadBalancer	**object**	loadBalancer holds parameters for the load balancer. Present only if type is LoadBalancerService.
nodePort	**object**	nodePort holds parameters for the NodePortService endpoint publishing strategy. Present only if type is NodePortService.
private	**object**	private holds parameters for the Private endpoint publishing strategy. Present only if type is Private.
type	**string**	type is the publishing strategy to use. Valid values are: * LoadBalancerService

Property	Type	Description
		Publishes the ingress controller using a Kubernetes LoadBalancer Service. In this configuration, the ingress controller deployment uses container networking. A LoadBalancer Service is created to publish the deployment. See: https://kubernetes.io/docs/concepts/services-networking/service/#loadbalancer If domain is set, a wildcard DNS record will be managed to point at the LoadBalancer Service's external name. DNS records are managed only in DNS zones defined by dns.config.openshift.io/cluster .spec.publicZone and .spec.privateZone. Wildcard DNS management is currently supported only on the AWS, Azure, and GCP platforms. * HostNetwork Publishes the ingress controller on node ports where the ingress controller is deployed. In this configuration, the ingress controller deployment uses host networking, bound to node ports 80 and 443. The user is responsible for configuring an external load balancer to publish the ingress controller via the node ports. * Private Does not publish the ingress controller. In this configuration, the ingress controller deployment uses container networking, and is not explicitly published. The user must manually publish the ingress controller. * NodePortService

Property	Type	Description
		Publishes the ingress controller using a Kubernetes NodePort Service. In this configuration, the ingress controller deployment uses container networking. A NodePort Service is created to publish the deployment. The specific node ports are dynamically allocated by OpenShift; however, to support static port allocations, user changes to the node port field of the managed NodePort Service will preserved.

15.1.57. .status.endpointPublishingStrategy.hostNetwork

Description

hostNetwork holds parameters for the HostNetwork endpoint publishing strategy. Present only if type is HostNetwork.

Type

object

Property	Type	Description
httpPort	**integer**	httpPort is the port on the host which should be used to listen for HTTP requests. This field should be set when port 80 is already in use. The value should not coincide with the NodePort range of the cluster. When the value is 0 or is not specified it defaults to 80.
httpsPort	**integer**	httpsPort is the port on the host which should be used to listen for HTTPS requests. This field should be set when port 443 is already in use. The value should not coincide with the NodePort range of the cluster. When the value is 0 or is not specified it defaults to 443.

Property	Type	Description
protocol	**string**	protocol specifies whether the IngressController expects incoming connections to use plain TCP or whether the IngressController expects PROXY protocol. PROXY protocol can be used with load balancers that support it to communicate the source addresses of client connections when forwarding those connections to the IngressController. Using PROXY protocol enables the IngressController to report those source addresses instead of reporting the load balancer's address in HTTP headers and logs. Note that enabling PROXY protocol on the IngressController will cause connections to fail if you are not using a load balancer that uses PROXY protocol to forward connections to the IngressController. See http://www.haproxy.org/download/2.2/doc/proxy-protocol.txt for information about PROXY protocol. The following values are valid for this field: * The empty string. * "TCP". * "PROXY". The empty string specifies the default, which is TCP without PROXY protocol. Note that the default is subject to change.

Property	Type	Description
statsPort	**integer**	statsPort is the port on the host where the stats from the router are published. The value should not coincide with the NodePort range of the cluster. If an external load balancer is configured to forward connections to this IngressController, the load balancer should use this port for health checks. The load balancer can send HTTP probes on this port on a given node, with the path /healthz/ready to determine if the ingress controller is ready to receive traffic on the node. For proper operation the load balancer must not forward traffic to a node until the health check reports ready. The load balancer should also stop forwarding requests within a maximum of 45 seconds after /healthz/ready starts reporting not-ready. Probing every 5 to 10 seconds, with a 5-second timeout and with a threshold of two successful or failed requests to become healthy or unhealthy respectively, are well-tested values. When the value is 0 or is not specified it defaults to 1936.

15.1.58. .status.endpointPublishingStrategy.loadBalancer

Description

loadBalancer holds parameters for the load balancer. Present only if type is LoadBalancerService.

Type

object

Required

- **dnsManagementPolicy**
- **scope**

Property	Type	Description

Property	Type	Description
allowedSourceRanges	``	allowedSourceRanges specifies an allowlist of IP address ranges to which access to the load balancer should be restricted. Each range must be specified using CIDR notation (e.g. "10.0.0.0/8" or "fd00::/8"). If no range is specified, "0.0.0.0/0" for IPv4 and "::/0" for IPv6 are used by default, which allows all source addresses. To facilitate migration from earlier versions of OpenShift that did not have the allowedSourceRanges field, you may set the service.beta.kubernetes.io/load-balancer-source-ranges annotation on the "router-<ingresscontroller name>" service in the "openshift-ingress" namespace, and this annotation will take effect if allowedSourceRanges is empty on OpenShift 4.12.
dnsManagementPolicy	**string**	dnsManagementPolicy indicates if the lifecycle of the wildcard DNS record associated with the load balancer service will be managed by the ingress operator. It defaults to Managed. Valid values are: Managed and Unmanaged.
providerParameters	**object**	providerParameters holds desired load balancer information specific to the underlying infrastructure provider. If empty, defaults will be applied. See specific providerParameters fields for details about their defaults.
scope	**string**	scope indicates the scope at which the load balancer is exposed. Possible values are "External" and "Internal".

15.1.59. .status.endpointPublishingStrategy.loadBalancer.providerParameters

Description

providerParameters holds desired load balancer information specific to the underlying infrastructure provider.
If empty, defaults will be applied. See specific providerParameters fields for details about their defaults.

Type

object

Required

- **type**

Property	Type	Description
aws	**object**	aws provides configuration settings that are specific to AWS load balancers. If empty, defaults will be applied. See specific aws fields for details about their defaults.
gcp	**object**	gcp provides configuration settings that are specific to GCP load balancers. If empty, defaults will be applied. See specific gcp fields for details about their defaults.
ibm	**object**	ibm provides configuration settings that are specific to IBM Cloud load balancers. If empty, defaults will be applied. See specific ibm fields for details about their defaults.
openstack	**object**	openstack provides configuration settings that are specific to OpenStack load balancers. If empty, defaults will be applied. See specific openstack fields for details about their defaults.

Property	Type	Description
type	**string**	type is the underlying infrastructure provider for the load balancer. Allowed values are "AWS", "Azure", "BareMetal", "GCP", "IBM", "Nutanix", "OpenStack", and "VSphere".

15.1.60. .status.endpointPublishingStrategy.loadBalancer.providerParameters.aws

Description

aws provides configuration settings that are specific to AWS load balancers.
If empty, defaults will be applied. See specific aws fields for details about their defaults.

Type

object

Required

- **type**

Property	Type	Description
classicLoadBalancer	**object**	classicLoadBalancerParameters holds configuration parameters for an AWS classic load balancer. Present only if type is Classic.
networkLoadBalancer	**object**	networkLoadBalancerParameters holds configuration parameters for an AWS network load balancer. Present only if type is NLB.

Property	Type	Description
type	**string**	type is the type of AWS load balancer to instantiate for an ingresscontroller. Valid values are: * "Classic": A Classic Load Balancer that makes routing decisions at either the transport layer (TCP/SSL) or the application layer (HTTP/HTTPS). See the following for additional details: https://docs.aws.amazon.com/AmazonECS/latest/developerguide/load-balancer-types.html#clb * "NLB": A Network Load Balancer that makes routing decisions at the transport layer (TCP/SSL). See the following for additional details: https://docs.aws.amazon.com/AmazonECS/latest/developerguide/load-balancer-types.html#nlb

15.1.61. .status.endpointPublishingStrategy.loadBalancer.providerParameters.aws.clas

Description

classicLoadBalancerParameters holds configuration parameters for an AWS classic load balancer. Present only if type is Classic.

Type

object

Property	Type	Description

Property	Type	Description
connectionIdleTimeout	**string**	connectionIdleTimeout specifies the maximum time period that a connection may be idle before the load balancer closes the connection. The value must be parseable as a time duration value; see https://pkg.go.dev/time#ParseDuration. A nil or zero value means no opinion, in which case a default value is used. The default value for this field is 60s. This default is subject to change.
subnets	**object**	subnets specifies the subnets to which the load balancer will attach. The subnets may be specified by either their ID or name. The total number of subnets is limited to 10. In order for the load balancer to be provisioned with subnets, each subnet must exist, each subnet must be from a different availability zone, and the load balancer service must be recreated to pick up new values. When omitted from the spec, the subnets will be auto-discovered for each availability zone. Auto-discovered subnets are not reported in the status of the IngressController object.

15.1.62. .status.endpointPublishingStrategy.loadBalancer.providerParameters.aws.clas

Description

subnets specifies the subnets to which the load balancer will attach. The subnets may be specified by either their ID or name. The total number of subnets is limited to 10.
In order for the load balancer to be provisioned with subnets, each subnet must exist, each subnet must be from a different availability zone, and the load balancer service must be recreated to pick up new values.

When omitted from the spec, the subnets will be auto-discovered for each availability zone. Auto-discovered subnets are not reported in the status of the IngressController object.

Type

object

Property	Type	Description
ids	**array (string)**	ids specifies a list of AWS subnets by subnet ID. Subnet IDs must start with "subnet-", consist only of alphanumeric characters, must be exactly 24 characters long, must be unique, and the total number of subnets specified by ids and names must not exceed 10.
names	**array (string)**	names specifies a list of AWS subnets by subnet name. Subnet names must not start with "subnet-", must not include commas, must be under 256 characters in length, must be unique, and the total number of subnets specified by ids and names must not exceed 10.

15.1.63. .status.endpointPublishingStrategy.loadBalancer.providerParameters.aws.net

Description

networkLoadBalancerParameters holds configuration parameters for an AWS network load balancer. Present only if type is NLB.

Type

object

Property	Type	Description

Property	Type	Description
eipAllocations	**array (string)**	eipAllocations is a list of IDs for Elastic IP (EIP) addresses that are assigned to the Network Load Balancer. The following restrictions apply: eipAllocations can only be used with external scope, not internal. An EIP can be allocated to only a single IngressController. The number of EIP allocations must match the number of subnets that are used for the load balancer. Each EIP allocation must be unique. A maximum of 10 EIP allocations are permitted. See https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/elastic-ip-addresses-eip.html for general information about configuration, characteristics, and limitations of Elastic IP addresses.
subnets	**object**	subnets specifies the subnets to which the load balancer will attach. The subnets may be specified by either their ID or name. The total number of subnets is limited to 10. In order for the load balancer to be provisioned with subnets, each subnet must exist, each subnet must be from a different availability zone, and the load balancer service must be recreated to pick up new values. When omitted from the spec, the subnets will be auto-discovered for each availability zone. Auto-discovered subnets are not reported in the status of the IngressController object.

15.1.64. .status.endpointPublishingStrategy.loadBalancer.providerParameters.aws.net

Description

subnets specifies the subnets to which the load balancer will attach. The subnets may be specified by either their ID or name. The total number of subnets is limited to 10.
In order for the load balancer to be provisioned with subnets, each subnet must exist, each subnet must be from a different availability zone, and the load balancer service must be recreated to pick up new values.

When omitted from the spec, the subnets will be auto-discovered for each availability zone. Auto-discovered subnets are not reported in the status of the IngressController object.

Type

object

Property	Type	Description
ids	**array (string)**	ids specifies a list of AWS subnets by subnet ID. Subnet IDs must start with "subnet-", consist only of alphanumeric characters, must be exactly 24 characters long, must be unique, and the total number of subnets specified by ids and names must not exceed 10.
names	**array (string)**	names specifies a list of AWS subnets by subnet name. Subnet names must not start with "subnet-", must not include commas, must be under 256 characters in length, must be unique, and the total number of subnets specified by ids and names must not exceed 10.

15.1.65. .status.endpointPublishingStrategy.loadBalancer.providerParameters.gcp

Description

gcp provides configuration settings that are specific to GCP load balancers.
If empty, defaults will be applied. See specific gcp fields for details about their defaults.

Type

object

Property	Type	Description

Property	Type	Description
clientAccess	**string**	clientAccess describes how client access is restricted for internal load balancers. Valid values are: * "Global": Specifying an internal load balancer with Global client access allows clients from any region within the VPC to communicate with the load balancer. https://cloud.google.com/kubernetes-engine/docs/how-to/internal-load-balancing#global_access * "Local": Specifying an internal load balancer with Local client access means only clients within the same region (and VPC) as the GCP load balancer can communicate with the load balancer. Note that this is the default behavior. https://cloud.google.com/load-balancing/docs/internal#client_access

15.1.66. .status.endpointPublishingStrategy.loadBalancer.providerParameters.ibm

Description

ibm provides configuration settings that are specific to IBM Cloud load balancers.
If empty, defaults will be applied. See specific ibm fields for details about their defaults.

Type

object

Property	Type	Description

Property	Type	Description
protocol	**string**	protocol specifies whether the load balancer uses PROXY protocol to forward connections to the IngressController. See "service.kubernetes.io/ibm-load-balancer-cloud-provider-enable-features: "proxy-protocol"" at https://cloud.ibm.com/docs/containers?topic=containers-vpc-lbaas PROXY protocol can be used with load balancers that support it to communicate the source addresses of client connections when forwarding those connections to the IngressController. Using PROXY protocol enables the IngressController to report those source addresses instead of reporting the load balancer's address in HTTP headers and logs. Note that enabling PROXY protocol on the IngressController will cause connections to fail if you are not using a load balancer that uses PROXY protocol to forward connections to the IngressController. See http://www.haproxy.org/download/2.2/doc/proxy-protocol.txt for information about PROXY protocol. Valid values for protocol are TCP, PROXY and omitted. When omitted, this means no opinion and the platform is left to choose a reasonable default, which is subject to change over time. The current default is TCP, without the proxy protocol enabled.

15.1.67. .status.endpointPublishingStrategy.loadBalancer.providerParameters.opensta

Description

openstack provides configuration settings that are specific to OpenStack load balancers.
If empty, defaults will be applied. See specific openstack fields for details about their defaults.

Type

object

Property	Type	Description
floatingIP	**string**	floatingIP specifies the IP address that the load balancer will use. When not specified, an IP address will be assigned randomly by the OpenStack cloud provider. When specified, the floating IP has to be pre-created. If the specified value is not a floating IP or is already claimed, the OpenStack cloud provider won't be able to provision the load balancer. This field may only be used if the IngressController has External scope. This value must be a valid IPv4 or IPv6 address.

15.1.68. .status.endpointPublishingStrategy.nodePort

Description

nodePort holds parameters for the NodePortService endpoint publishing strategy. Present only if type is NodePortService.

Type

object

Property	Type	Description

Property	Type	Description
protocol	**string**	protocol specifies whether the IngressController expects incoming connections to use plain TCP or whether the IngressController expects PROXY protocol. PROXY protocol can be used with load balancers that support it to communicate the source addresses of client connections when forwarding those connections to the IngressController. Using PROXY protocol enables the IngressController to report those source addresses instead of reporting the load balancer's address in HTTP headers and logs. Note that enabling PROXY protocol on the IngressController will cause connections to fail if you are not using a load balancer that uses PROXY protocol to forward connections to the IngressController. See http://www.haproxy.org/download/2.2/doc/proxy-protocol.txt for information about PROXY protocol. The following values are valid for this field: * The empty string. * "TCP". * "PROXY". The empty string specifies the default, which is TCP without PROXY protocol. Note that the default is subject to change.

15.1.69. .status.endpointPublishingStrategy.private

Description

private holds parameters for the Private endpoint publishing strategy. Present only if type is Private.

Type

object

Property	Type	Description
protocol	**string**	protocol specifies whether the IngressController expects incoming connections to use plain TCP or whether the IngressController expects PROXY protocol. PROXY protocol can be used with load balancers that support it to communicate the source addresses of client connections when forwarding those connections to the IngressController. Using PROXY protocol enables the IngressController to report those source addresses instead of reporting the load balancer's address in HTTP headers and logs. Note that enabling PROXY protocol on the IngressController will cause connections to fail if you are not using a load balancer that uses PROXY protocol to forward connections to the IngressController. See http://www.haproxy.org/download/2.2/doc/proxy-protocol.txt for information about PROXY protocol. The following values are valid for this field: * The empty string. * "TCP". * "PROXY". The empty string specifies the default, which is TCP without PROXY protocol. Note that the default is subject to change.

15.1.70. .status.namespaceSelector

Description

namespaceSelector is the actual namespaceSelector in use.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

15.1.71. .status.namespaceSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

15.1.72. .status.namespaceSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.

Property	Type	Description
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

15.1.73. .status.routeSelector

Description

routeSelector is the actual routeSelector in use.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

15.1.74. .status.routeSelector.matchExpressions

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

15.1.75. .status.routeSelector.matchExpressions[]

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

15.1.76. .status.tlsProfile

Description

tlsProfile is the TLS connection configuration that is in effect.

Type

object

Property	Type	Description

Property	Type	Description
ciphers	**array (string)**	ciphers is used to specify the cipher algorithms that are negotiated during the TLS handshake. Operators may remove entries their operands do not support. For example, to use DES-CBC3-SHA (yaml): ciphers: - DES-CBC3-SHA
minTLSVersion	**string**	minTLSVersion is used to specify the minimal version of the TLS protocol that is negotiated during the TLS handshake. For example, to use TLS versions 1.1, 1.2 and 1.3 (yaml): minTLSVersion: VersionTLS11 NOTE: currently the highest minTLSVersion allowed is VersionTLS12

15.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/ingresscontrollers**
  - **GET**: list objects of kind IngressController
- **/apis/operator.openshift.io/v1/namespaces/{namespace}/ingresscontrollers**
  - **DELETE**: delete collection of IngressController
  - **GET**: list objects of kind IngressController
  - **POST**: create an IngressController
- **/apis/operator.openshift.io/v1/namespaces/{namespace}/ingresscontrollers/{name}**
  - **DELETE**: delete an IngressController
  - **GET**: read the specified IngressController
  - **PATCH**: partially update the specified IngressController
  - **PUT**: replace the specified IngressController
- **/apis/operator.openshift.io/v1/namespaces/{namespace}/ingresscontrollers/{name}/scale**

- **GET**: read scale of the specified IngressController
- **PATCH**: partially update scale of the specified IngressController
- **PUT**: replace scale of the specified IngressController

- **/apis/operator.openshift.io/v1/namespaces/{namespace}/ingresscontrollers/{name}/status**
  - **GET**: read status of the specified IngressController
  - **PATCH**: partially update status of the specified IngressController
  - **PUT**: replace status of the specified IngressController

15.2.1. /apis/operator.openshift.io/v1/ingresscontrollers

HTTP method

GET

Description

list objects of kind IngressController

Table 15.1. HTTP responses

HTTP code	Reponse body
200 - OK	**IngressControllerList** schema
401 - Unauthorized	Empty

15.2.2. /apis/operator.openshift.io/v1/namespaces/{namespace}/ingresscontrollers

HTTP method

DELETE

Description

delete collection of IngressController

Table 15.2. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind IngressController

Table 15.3. HTTP responses

HTTP code	Reponse body
200 - OK	**IngressControllerList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create an IngressController

Table 15.4. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 15.5. Body parameters

Parameter	Type	Description
body	**IngressController** schema	

Table 15.6. HTTP responses

HTTP code	Reponse body
200 - OK	**IngressController** schema
201 - Created	**IngressController** schema
202 - Accepted	**IngressController** schema
401 - Unauthorized	Empty

15.2.3. /apis/operator.openshift.io/v1/namespaces/{namespace}/ingresscontrollers/{

Table 15.7. Global path parameters

Parameter	Type	Description
name	**string**	name of the IngressController

HTTP method

DELETE

Description

delete an IngressController

Table 15.8. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 15.9. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified IngressController

Table 15.10. HTTP responses

HTTP code	Reponse body
200 - OK	**IngressController** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified IngressController

Table 15.11. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 15.12. HTTP responses

HTTP code	Reponse body
200 - OK	**IngressController** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified IngressController

Table 15.13. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 15.14. Body parameters

Parameter	Type	Description
body	**IngressController** schema	

Table 15.15. HTTP responses

HTTP code	Reponse body
200 - OK	**IngressController** schema
201 - Created	**IngressController** schema
401 - Unauthorized	Empty

15.2.4. /apis/operator.openshift.io/v1/namespaces/{namespace}/ingresscontrollers/{

Table 15.16. Global path parameters

Parameter	Type	Description
name	**string**	name of the IngressController

HTTP method

GET

Description

read scale of the specified IngressController

Table 15.17. HTTP responses

HTTP code	Reponse body
200 - OK	**Scale** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update scale of the specified IngressController

Table 15.18. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 15.19. HTTP responses

HTTP code	Reponse body
200 - OK	**Scale** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace scale of the specified IngressController

Table 15.20. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 15.21. Body parameters

Parameter	Type	Description
body	**Scale** schema	

Table 15.22. HTTP responses

HTTP code	Reponse body
200 - OK	**Scale** schema
201 - Created	**Scale** schema
401 - Unauthorized	Empty

15.2.5. /apis/operator.openshift.io/v1/namespaces/{namespace}/ingresscontrollers/{

Table 15.23. Global path parameters

Parameter	Type	Description
name	**string**	name of the IngressController

HTTP method

GET

Description

read status of the specified IngressController

Table 15.24. HTTP responses

HTTP code	Reponse body
200 - OK	**IngressController** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified IngressController

Table 15.25. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 15.26. HTTP responses

HTTP code	Reponse body
200 - OK	**IngressController** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified IngressController

Table 15.27. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 15.28. Body parameters

Parameter	Type	Description
body	**IngressController** schema	

Table 15.29. HTTP responses

HTTP code	Reponse body
200 - OK	**IngressController** schema
201 - Created	**IngressController** schema
401 - Unauthorized	Empty

CHAPTER 16. INSIGHTSOPERATOR [OPERATOR.OPENSHIFT.IO/V1]

Description

InsightsOperator holds cluster-wide information about the Insights Operator. Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

16.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	spec is the specification of the desired behavior of the Insights.

Property	Type	Description
status	**object**	status is the most recently observed status of the Insights operator.

16.1.1. .spec

Description

spec is the specification of the desired behavior of the Insights.

Type

object

Property	Type	Description
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
managementState	**string**	managementState indicates whether and how the operator should manage the component
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".

Property	Type	Description
unsupportedConfigOverrides	``	unsupportedConfigOverrides holds a sparse config that will override any previously set options. It only needs to be the fields to override it will end up overlaying in the following order: 1. hardcoded defaults 2. observedConfig 3. unsupportedConfigOverrides

16.1.2. .status

Description

status is the most recently observed status of the Insights operator.

Type

object

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status
conditions[]	**object**	OperatorCondition is just the standard condition fields.
gatherStatus	**object**	gatherStatus provides basic information about the last Insights data gathering. When omitted, this means no data gathering has taken place yet.
generations	**array**	generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.
generations[]	**object**	GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.
insightsReport	**object**	insightsReport provides general Insights analysis results. When omitted, this means no data gathering has taken place yet.

Property	Type	Description
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with
readyReplicas	**integer**	readyReplicas indicates how many replicas are ready and at the desired state
version	**string**	version is the level this availability applies to

16.1.3. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

16.1.4. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Property	Type	Description
lastTransitionTime	**string**	
message	**string**	
reason	**string**	
status	**string**	
type	**string**	

16.1.5. .status.gatherStatus

Description

gatherStatus provides basic information about the last Insights data gathering. When omitted, this means no data gathering has taken place yet.

Type

object

Property	Type	Description
gatherers	**array**	gatherers is a list of active gatherers (and their statuses) in the last gathering.
gatherers[]	**object**	gathererStatus represents information about a particular data gatherer.
lastGatherDuration	**string**	lastGatherDuration is the total time taken to process all gatherers during the last gather event.
lastGatherTime	**string**	lastGatherTime is the last time when Insights data gathering finished. An empty value means that no data has been gathered yet.

16.1.6. .status.gatherStatus.gatherers

Description

gatherers is a list of active gatherers (and their statuses) in the last gathering.

Type

array

16.1.7. .status.gatherStatus.gatherers[]

Description

gathererStatus represents information about a particular data gatherer.

Type

object

Required

- **conditions**
- **lastGatherDuration**
- **name**

Property	Type	Description
conditions	**array**	conditions provide details on the status of each gatherer.

Property	Type	Description
conditions[]	**object**	Condition contains details for one aspect of the current state of this API Resource. --- This struct is intended for direct use as an array at the field path .status.conditions. For example, type FooStatus struct{ // Represents the observations of a foo's current state. // Known .status.conditions.type are: "Available", "Progressing", and "Degraded" // +patchMergeKey=type // +patchStrategy=merge // +listType=map // +listMapKey=type Conditions []metav1.Condition **json:"conditions,omitempty" patchStrategy:"merge" patchMergeKey:"type" protobuf:"bytes,1,rep,name=conditions"** // other fields }
lastGatherDuration	**string**	lastGatherDuration represents the time spent gathering.
name	**string**	name is the name of the gatherer.

16.1.8. .status.gatherStatus.gatherers[].conditions

Description

conditions provide details on the status of each gatherer.

Type

array

16.1.9. .status.gatherStatus.gatherers[].conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource. --- This struct is intended for direct use as an array at the field path .status.conditions. For example, type FooStatus struct{ // Represents the observations of a foo's current state. // Known .status.conditions.type are: "Available", "Progressing", and "Degraded" // +patchMergeKey=type // +patchStrategy=merge // +listType=map // +listMapKey=type Conditions []metav1.Condition **json:"conditions,omitempty" patchStrategy:"merge" patchMergeKey:"type" protobuf:"bytes,1,rep,name=conditions"** // other fields }

Type

object

Required

- **lastTransitionTime**
- **message**
- **reason**
- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	message is a human readable message indicating details about the transition. This may be an empty string.
observedGeneration	**integer**	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.
reason	**string**	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.
status	**string**	status of the condition, one of True, False, Unknown.

Property	Type	Description
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase. --- Many .condition.type values are consistent across resources like Available, but because arbitrary conditions can be useful (see .node.status.conditions), the ability to deconflict is important. The regex it matches is (dns1123SubdomainFmt/)? (qualifiedNameFmt)

16.1.10. .status.generations

Description

generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Type

array

16.1.11. .status.generations[]

Description

GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.

Type

object

Property	Type	Description
group	**string**	group is the group of the thing you're tracking
hash	**string**	hash is an optional field set for resources without generation that are content sensitive like secrets and configmaps
lastGeneration	**integer**	lastGeneration is the last generation of the workload controller involved
name	**string**	name is the name of the thing you're tracking

Property	Type	Description
namespace	**string**	namespace is where the thing you're tracking is
resource	**string**	resource is the resource type of the thing you're tracking

16.1.12. .status.insightsReport

Description

insightsReport provides general Insights analysis results. When omitted, this means no data gathering has taken place yet.

Type

object

Property	Type	Description
downloadedAt	**string**	downloadedAt is the time when the last Insights report was downloaded. An empty value means that there has not been any Insights report downloaded yet and it usually appears in disconnected clusters (or clusters when the Insights data gathering is disabled).
healthChecks	**array**	healthChecks provides basic information about active Insights health checks in a cluster.
healthChecks[]	**object**	healthCheck represents an Insights health check attributes.

16.1.13. .status.insightsReport.healthChecks

Description

healthChecks provides basic information about active Insights health checks in a cluster.

Type

array

16.1.14. .status.insightsReport.healthChecks[]

Description

healthCheck represents an Insights health check attributes.

Type

object

Required

- **advisorURI**
- **description**
- **state**
- **totalRisk**

Property	Type	Description
advisorURI	**string**	advisorURI provides the URL link to the Insights Advisor.
description	**string**	description provides basic description of the healtcheck.
state	**string**	state determines what the current state of the health check is. Health check is enabled by default and can be disabled by the user in the Insights advisor user interface.
totalRisk	**integer**	totalRisk of the healthcheck. Indicator of the total risk posed by the detected issue; combination of impact and likelihood. The values can be from 1 to 4, and the higher the number, the more important the issue.

16.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/insightsoperators**
  - **DELETE**: delete collection of InsightsOperator
  - **GET**: list objects of kind InsightsOperator
  - **POST**: create an InsightsOperator
- **/apis/operator.openshift.io/v1/insightsoperators/{name}**
  - **DELETE**: delete an InsightsOperator
  - **GET**: read the specified InsightsOperator
  - **PATCH**: partially update the specified InsightsOperator

- **PUT**: replace the specified InsightsOperator

- **/apis/operator.openshift.io/v1/insightsoperators/{name}/scale**
  - **GET**: read scale of the specified InsightsOperator
  - **PATCH**: partially update scale of the specified InsightsOperator
  - **PUT**: replace scale of the specified InsightsOperator
- **/apis/operator.openshift.io/v1/insightsoperators/{name}/status**
  - **GET**: read status of the specified InsightsOperator
  - **PATCH**: partially update status of the specified InsightsOperator
  - **PUT**: replace status of the specified InsightsOperator

16.2.1. /apis/operator.openshift.io/v1/insightsoperators

HTTP method

DELETE

Description

delete collection of InsightsOperator

Table 16.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind InsightsOperator

Table 16.2. HTTP responses

HTTP code	Reponse body
200 - OK	**InsightsOperatorList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create an InsightsOperator

Table 16.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 16.4. Body parameters

Parameter	Type	Description
body	**InsightsOperator** schema	

Table 16.5. HTTP responses

HTTP code	Reponse body
200 - OK	**InsightsOperator** schema
201 - Created	**InsightsOperator** schema
202 - Accepted	**InsightsOperator** schema
401 - Unauthorized	Empty

16.2.2. /apis/operator.openshift.io/v1/insightsoperators/{name}

Table 16.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the InsightsOperator

HTTP method

DELETE

Description

delete an InsightsOperator

Table 16.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 16.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified InsightsOperator

Table 16.9. HTTP responses

HTTP code	Reponse body
200 - OK	**InsightsOperator** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified InsightsOperator

Table 16.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 16.11. HTTP responses

HTTP code	Reponse body
200 - OK	**InsightsOperator** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified InsightsOperator

Table 16.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 16.13. Body parameters

Parameter	Type	Description
body	**InsightsOperator** schema	

Table 16.14. HTTP responses

HTTP code	Reponse body
200 - OK	**InsightsOperator** schema
201 - Created	**InsightsOperator** schema
401 - Unauthorized	Empty

16.2.3. /apis/operator.openshift.io/v1/insightsoperators/{name}/scale

Table 16.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the InsightsOperator

HTTP method

GET

Description

read scale of the specified InsightsOperator

Table 16.16. HTTP responses

HTTP code	Reponse body
200 - OK	**Scale** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update scale of the specified InsightsOperator

Table 16.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 16.18. HTTP responses

HTTP code	Reponse body
200 - OK	**Scale** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace scale of the specified InsightsOperator

Table 16.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 16.20. Body parameters

Parameter	Type	Description
body	**Scale** schema	

Table 16.21. HTTP responses

HTTP code	Reponse body
200 - OK	**Scale** schema
201 - Created	**Scale** schema
401 - Unauthorized	Empty

16.2.4. /apis/operator.openshift.io/v1/insightsoperators/{name}/status

Table 16.22. Global path parameters

Parameter	Type	Description
name	**string**	name of the InsightsOperator

HTTP method

GET

Description

read status of the specified InsightsOperator

Table 16.23. HTTP responses

HTTP code	Reponse body
200 - OK	**InsightsOperator** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified InsightsOperator

Table 16.24. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 16.25. HTTP responses

HTTP code	Reponse body
200 - OK	**InsightsOperator** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified InsightsOperator

Table 16.26. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 16.27. Body parameters

Parameter	Type	Description
body	**InsightsOperator** schema	

Table 16.28. HTTP responses

HTTP code	Reponse body
200 - OK	**InsightsOperator** schema
201 - Created	**InsightsOperator** schema
401 - Unauthorized	Empty

CHAPTER 17. KUBEAPISERVER [OPERATOR.OPENSHIFT.IO/V1]

Description

KubeAPIServer provides information to configure an operator to manage kube-apiserver. Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

17.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	spec is the specification of the desired behavior of the Kubernetes API Server
status	**object**	status is the most recently observed status of the Kubernetes API Server

17.1.1. .spec

Description

spec is the specification of the desired behavior of the Kubernetes API Server

Type

object

Property	Type	Description
failedRevisionLimit	**integer**	failedRevisionLimit is the number of failed static pod installer revisions to keep on disk and in the api -1 = unlimited, 0 or unset = 5 (default)
forceRedeploymentReason	**string**	forceRedeploymentReason can be used to force the redeployment of the operand by providing a unique string. This provides a mechanism to kick a previously failed deployment and provide a reason why you think it will work this time instead of failing again on the same config.
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
managementState	**string**	managementState indicates whether and how the operator should manage the component

Property	Type	Description
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
succeededRevisionLimit	**integer**	succeededRevisionLimit is the number of successful static pod installer revisions to keep on disk and in the api -1 = unlimited, 0 or unset = 5 (default)
unsupportedConfigOverrides	``	unsupportedConfigOverrides overrides the final configuration that was computed by the operator. Red Hat does not support the use of this field. Misuse of this field could lead to unexpected behavior or conflict with other configuration options. Seek guidance from the Red Hat support before using this field. Use of this property blocks cluster upgrades, it must be removed before upgrading your cluster.

17.1.2. .status

Description

status is the most recently observed status of the Kubernetes API Server

Type

object

Property	Type	Description

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status
conditions[]	**object**	OperatorCondition is just the standard condition fields.
generations	**array**	generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.
generations[]	**object**	GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.
latestAvailableRevision	**integer**	latestAvailableRevision is the deploymentID of the most recent deployment
latestAvailableRevisionReason	**string**	latestAvailableRevisionReason describe the detailed reason for the most recent deployment
nodeStatuses	**array**	nodeStatuses track the deployment values and errors across individual nodes
nodeStatuses[]	**object**	NodeStatus provides information about the current state of a particular node managed by this operator.
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with
readyReplicas	**integer**	readyReplicas indicates how many replicas are ready and at the desired state

Property	Type	Description
serviceAccountIssuers	**array**	serviceAccountIssuers tracks history of used service account issuers. The item without expiration time represents the currently used service account issuer. The other items represents service account issuers that were used previously and are still being trusted. The default expiration for the items is set by the platform and it defaults to 24h. see: https://kubernetes.io/docs/tasks/configure-pod-container/configure-service-account/#service-account-token-volume-projection
serviceAccountIssuers[]	**object**	
version	**string**	version is the level this availability applies to

17.1.3. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

17.1.4. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	
reason	**string**	
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

17.1.5. .status.generations

Description

generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Type

array

17.1.6. .status.generations[]

Description

GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.

Type

object

Required

- **group**
- **name**
- **namespace**
- **resource**

Property	Type	Description
group	**string**	group is the group of the thing you're tracking
hash	**string**	hash is an optional field set for resources without generation that are content sensitive like secrets and configmaps
lastGeneration	**integer**	lastGeneration is the last generation of the workload controller involved
name	**string**	name is the name of the thing you're tracking
namespace	**string**	namespace is where the thing you're tracking is
resource	**string**	resource is the resource type of the thing you're tracking

17.1.7. .status.nodeStatuses

Description

nodeStatuses track the deployment values and errors across individual nodes

Type

array

17.1.8. .status.nodeStatuses[]

Description

NodeStatus provides information about the current state of a particular node managed by this operator.

Type

object

Required

- **nodeName**

Property	Type	Description

Property	Type	Description
currentRevision	**integer**	currentRevision is the generation of the most recently successful deployment. Can not be set on creation of a nodeStatus. Updates must only increase the value.
lastFailedCount	**integer**	lastFailedCount is how often the installer pod of the last failed revision failed.
lastFailedReason	**string**	lastFailedReason is a machine readable failure reason string.
lastFailedRevision	**integer**	lastFailedRevision is the generation of the deployment we tried and failed to deploy.
lastFailedRevisionErrors	**array (string)**	lastFailedRevisionErrors is a list of human readable errors during the failed deployment referenced in lastFailedRevision.
lastFailedTime	**string**	lastFailedTime is the time the last failed revision failed the last time.
lastFallbackCount	**integer**	lastFallbackCount is how often a fallback to a previous revision happened.
nodeName	**string**	nodeName is the name of the node
targetRevision	**integer**	targetRevision is the generation of the deployment we're trying to apply. Can not be set on creation of a nodeStatus.

17.1.9. .status.serviceAccountIssuers

Description

serviceAccountIssuers tracks history of used service account issuers. The item without expiration time represents the currently used service account issuer. The other items represents service account issuers that were used previously and are still being trusted. The default expiration for the items is set by the platform and it defaults to 24h. see: https://kubernetes.io/docs/tasks/configure-pod-container/configure-service-account/#service-account-token-volume-projection

Type

array

17.1.10. .status.serviceAccountIssuers[]

Description

Type

object

Property	Type	Description
expirationTime	**string**	expirationTime is the time after which this service account issuer will be pruned and removed from the trusted list of service account issuers.
name	**string**	name is the name of the service account issuer

17.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/kubeapiservers**
  - **DELETE**: delete collection of KubeAPIServer
  - **GET**: list objects of kind KubeAPIServer
  - **POST**: create a KubeAPIServer
- **/apis/operator.openshift.io/v1/kubeapiservers/{name}**
  - **DELETE**: delete a KubeAPIServer
  - **GET**: read the specified KubeAPIServer
  - **PATCH**: partially update the specified KubeAPIServer
  - **PUT**: replace the specified KubeAPIServer
- **/apis/operator.openshift.io/v1/kubeapiservers/{name}/status**
  - **GET**: read status of the specified KubeAPIServer
  - **PATCH**: partially update status of the specified KubeAPIServer
  - **PUT**: replace status of the specified KubeAPIServer

17.2.1. /apis/operator.openshift.io/v1/kubeapiservers

HTTP method

DELETE

Description

delete collection of KubeAPIServer

Table 17.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind KubeAPIServer

Table 17.2. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeAPIServerList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a KubeAPIServer

Table 17.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 17.4. Body parameters

Parameter	Type	Description
body	**KubeAPIServer** schema	

Table 17.5. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeAPIServer** schema
201 - Created	**KubeAPIServer** schema
202 - Accepted	**KubeAPIServer** schema
401 - Unauthorized	Empty

17.2.2. /apis/operator.openshift.io/v1/kubeapiservers/{name}

Table 17.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the KubeAPIServer

HTTP method

DELETE

Description

delete a KubeAPIServer

Table 17.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 17.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified KubeAPIServer

Table 17.9. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeAPIServer** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified KubeAPIServer

Table 17.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 17.11. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeAPIServer** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified KubeAPIServer

Table 17.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 17.13. Body parameters

Parameter	Type	Description
body	**KubeAPIServer** schema	

Table 17.14. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeAPIServer** schema
201 - Created	**KubeAPIServer** schema
401 - Unauthorized	Empty

17.2.3. /apis/operator.openshift.io/v1/kubeapiservers/{name}/status

Table 17.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the KubeAPIServer

HTTP method

GET

Description

read status of the specified KubeAPIServer

Table 17.16. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeAPIServer** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified KubeAPIServer

Table 17.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 17.18. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeAPIServer** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified KubeAPIServer

Table 17.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 17.20. Body parameters

Parameter	Type	Description
body	**KubeAPIServer** schema	

Table 17.21. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeAPIServer** schema
201 - Created	**KubeAPIServer** schema
401 - Unauthorized	Empty

CHAPTER 18. KUBECONTROLLERMANAGER [OPERATOR.OPENSHIFT.IO/V1]

Description

KubeControllerManager provides information to configure an operator to manage kube-controller-manager.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

18.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	spec is the specification of the desired behavior of the Kubernetes Controller Manager
status	**object**	status is the most recently observed status of the Kubernetes Controller Manager

18.1.1. .spec

Description

spec is the specification of the desired behavior of the Kubernetes Controller Manager

Type

object

Property	Type	Description
failedRevisionLimit	**integer**	failedRevisionLimit is the number of failed static pod installer revisions to keep on disk and in the api -1 = unlimited, 0 or unset = 5 (default)
forceRedeploymentReason	**string**	forceRedeploymentReason can be used to force the redeployment of the operand by providing a unique string. This provides a mechanism to kick a previously failed deployment and provide a reason why you think it will work this time instead of failing again on the same config.
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".

Property	Type	Description
managementState	**string**	managementState indicates whether and how the operator should manage the component
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
succeededRevisionLimit	**integer**	succeededRevisionLimit is the number of successful static pod installer revisions to keep on disk and in the api -1 = unlimited, 0 or unset = 5 (default)
unsupportedConfigOverrides	``	unsupportedConfigOverrides overrides the final configuration that was computed by the operator. Red Hat does not support the use of this field. Misuse of this field could lead to unexpected behavior or conflict with other configuration options. Seek guidance from the Red Hat support before using this field. Use of this property blocks cluster upgrades, it must be removed before upgrading your cluster.

Property	Type	Description
useMoreSecureServiceCA	**boolean**	useMoreSecureServiceCA indicates that the service-ca.crt provided in SA token volumes should include only enough certificates to validate service serving certificates. Once set to true, it cannot be set to false. Even if someone finds a way to set it back to false, the service-ca.crt files that previously existed will only have the more secure content.

18.1.2. .status

Description

status is the most recently observed status of the Kubernetes Controller Manager

Type

object

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status
conditions[]	**object**	OperatorCondition is just the standard condition fields.
generations	**array**	generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.
generations[]	**object**	GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.
latestAvailableRevision	**integer**	latestAvailableRevision is the deploymentID of the most recent deployment
latestAvailableRevisionReason	**string**	latestAvailableRevisionReason describe the detailed reason for the most recent deployment

Property	Type	Description
nodeStatuses	**array**	nodeStatuses track the deployment values and errors across individual nodes
nodeStatuses[]	**object**	NodeStatus provides information about the current state of a particular node managed by this operator.
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with
readyReplicas	**integer**	readyReplicas indicates how many replicas are ready and at the desired state
version	**string**	version is the level this availability applies to

18.1.3. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

18.1.4. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	
reason	**string**	
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

18.1.5. .status.generations

Description

generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Type

array

18.1.6. .status.generations[]

Description

GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.

Type

object

Required

- **group**
- **name**
- **namespace**
- **resource**

Property	Type	Description
group	**string**	group is the group of the thing you're tracking
hash	**string**	hash is an optional field set for resources without generation that are content sensitive like secrets and configmaps
lastGeneration	**integer**	lastGeneration is the last generation of the workload controller involved
name	**string**	name is the name of the thing you're tracking
namespace	**string**	namespace is where the thing you're tracking is
resource	**string**	resource is the resource type of the thing you're tracking

18.1.7. .status.nodeStatuses

Description

nodeStatuses track the deployment values and errors across individual nodes

Type

array

18.1.8. .status.nodeStatuses[]

Description

NodeStatus provides information about the current state of a particular node managed by this operator.

Type

object

Required

- **nodeName**

Property	Type	Description

Property	Type	Description
currentRevision	**integer**	currentRevision is the generation of the most recently successful deployment. Can not be set on creation of a nodeStatus. Updates must only increase the value.
lastFailedCount	**integer**	lastFailedCount is how often the installer pod of the last failed revision failed.
lastFailedReason	**string**	lastFailedReason is a machine readable failure reason string.
lastFailedRevision	**integer**	lastFailedRevision is the generation of the deployment we tried and failed to deploy.
lastFailedRevisionErrors	**array (string)**	lastFailedRevisionErrors is a list of human readable errors during the failed deployment referenced in lastFailedRevision.
lastFailedTime	**string**	lastFailedTime is the time the last failed revision failed the last time.
lastFallbackCount	**integer**	lastFallbackCount is how often a fallback to a previous revision happened.
nodeName	**string**	nodeName is the name of the node
targetRevision	**integer**	targetRevision is the generation of the deployment we're trying to apply. Can not be set on creation of a nodeStatus.

18.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/kubecontrollermanagers**
  - **DELETE**: delete collection of KubeControllerManager
  - **GET**: list objects of kind KubeControllerManager

- **POST**: create a KubeControllerManager

- **/apis/operator.openshift.io/v1/kubecontrollermanagers/{name}**
  - **DELETE**: delete a KubeControllerManager
  - **GET**: read the specified KubeControllerManager
  - **PATCH**: partially update the specified KubeControllerManager
  - **PUT**: replace the specified KubeControllerManager
- **/apis/operator.openshift.io/v1/kubecontrollermanagers/{name}/status**
  - **GET**: read status of the specified KubeControllerManager
  - **PATCH**: partially update status of the specified KubeControllerManager
  - **PUT**: replace status of the specified KubeControllerManager

18.2.1. /apis/operator.openshift.io/v1/kubecontrollermanagers

HTTP method

DELETE

Description

delete collection of KubeControllerManager

Table 18.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind KubeControllerManager

Table 18.2. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeControllerManagerList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a KubeControllerManager

Table 18.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 18.4. Body parameters

Parameter	Type	Description
body	**KubeControllerManager** schema	

Table 18.5. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeControllerManager** schema
201 - Created	**KubeControllerManager** schema
202 - Accepted	**KubeControllerManager** schema

HTTP code	Reponse body
401 - Unauthorized	Empty

18.2.2. /apis/operator.openshift.io/v1/kubecontrollermanagers/{name}

Table 18.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the KubeControllerManager

HTTP method

DELETE

Description

delete a KubeControllerManager

Table 18.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 18.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified KubeControllerManager

Table 18.9. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeControllerManager** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified KubeControllerManager

Table 18.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 18.11. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeControllerManager** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified KubeControllerManager

Table 18.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 18.13. Body parameters

Parameter	Type	Description
body	**KubeControllerManager** schema	

Table 18.14. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeControllerManager** schema
201 - Created	**KubeControllerManager** schema

HTTP code	Reponse body
401 - Unauthorized	Empty

18.2.3. /apis/operator.openshift.io/v1/kubecontrollermanagers/{name}/status

Table 18.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the KubeControllerManager

HTTP method

GET

Description

read status of the specified KubeControllerManager

Table 18.16. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeControllerManager** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified KubeControllerManager

Table 18.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 18.18. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeControllerManager** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified KubeControllerManager

Table 18.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 18.20. Body parameters

Parameter	Type	Description
body	**KubeControllerManager** schema	

Table 18.21. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeControllerManager** schema
201 - Created	**KubeControllerManager** schema
401 - Unauthorized	Empty

CHAPTER 19. KUBESCHEDULER [OPERATOR.OPENSHIFT.IO/V1]

Description

KubeScheduler provides information to configure an operator to manage scheduler. Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

19.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	spec is the specification of the desired behavior of the Kubernetes Scheduler
status	**object**	status is the most recently observed status of the Kubernetes Scheduler

19.1.1. .spec

Description

spec is the specification of the desired behavior of the Kubernetes Scheduler

Type

object

Property	Type	Description
failedRevisionLimit	**integer**	failedRevisionLimit is the number of failed static pod installer revisions to keep on disk and in the api -1 = unlimited, 0 or unset = 5 (default)
forceRedeploymentReason	**string**	forceRedeploymentReason can be used to force the redeployment of the operand by providing a unique string. This provides a mechanism to kick a previously failed deployment and provide a reason why you think it will work this time instead of failing again on the same config.
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
managementState	**string**	managementState indicates whether and how the operator should manage the component

Property	Type	Description
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
succeededRevisionLimit	**integer**	succeededRevisionLimit is the number of successful static pod installer revisions to keep on disk and in the api -1 = unlimited, 0 or unset = 5 (default)
unsupportedConfigOverrides	``	unsupportedConfigOverrides overrides the final configuration that was computed by the operator. Red Hat does not support the use of this field. Misuse of this field could lead to unexpected behavior or conflict with other configuration options. Seek guidance from the Red Hat support before using this field. Use of this property blocks cluster upgrades, it must be removed before upgrading your cluster.

19.1.2. .status

Description

status is the most recently observed status of the Kubernetes Scheduler

Type

object

Property	Type	Description

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status
conditions[]	**object**	OperatorCondition is just the standard condition fields.
generations	**array**	generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.
generations[]	**object**	GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.
latestAvailableRevision	**integer**	latestAvailableRevision is the deploymentID of the most recent deployment
latestAvailableRevisionReason	**string**	latestAvailableRevisionReason describe the detailed reason for the most recent deployment
nodeStatuses	**array**	nodeStatuses track the deployment values and errors across individual nodes
nodeStatuses[]	**object**	NodeStatus provides information about the current state of a particular node managed by this operator.
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with
readyReplicas	**integer**	readyReplicas indicates how many replicas are ready and at the desired state
version	**string**	version is the level this availability applies to

19.1.3. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

19.1.4. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	
reason	**string**	
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

19.1.5. .status.generations

Description

generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Type

array

19.1.6. .status.generations[]

Description

GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.

Type

object

Required

- **group**
- **name**
- **namespace**
- **resource**

Property	Type	Description
group	**string**	group is the group of the thing you're tracking
hash	**string**	hash is an optional field set for resources without generation that are content sensitive like secrets and configmaps
lastGeneration	**integer**	lastGeneration is the last generation of the workload controller involved
name	**string**	name is the name of the thing you're tracking
namespace	**string**	namespace is where the thing you're tracking is
resource	**string**	resource is the resource type of the thing you're tracking

19.1.7. .status.nodeStatuses

Description

nodeStatuses track the deployment values and errors across individual nodes

Type

array

19.1.8. .status.nodeStatuses[]

Description

NodeStatus provides information about the current state of a particular node managed by this operator.

Type

object

Required

- **nodeName**

Property	Type	Description
currentRevision	**integer**	currentRevision is the generation of the most recently successful deployment. Can not be set on creation of a nodeStatus. Updates must only increase the value.
lastFailedCount	**integer**	lastFailedCount is how often the installer pod of the last failed revision failed.
lastFailedReason	**string**	lastFailedReason is a machine readable failure reason string.
lastFailedRevision	**integer**	lastFailedRevision is the generation of the deployment we tried and failed to deploy.
lastFailedRevisionErrors	**array (string)**	lastFailedRevisionErrors is a list of human readable errors during the failed deployment referenced in lastFailedRevision.
lastFailedTime	**string**	lastFailedTime is the time the last failed revision failed the last time.
lastFallbackCount	**integer**	lastFallbackCount is how often a fallback to a previous revision happened.
nodeName	**string**	nodeName is the name of the node
targetRevision	**integer**	targetRevision is the generation of the deployment we're trying to apply. Can not be set on creation of a nodeStatus.

19.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/kubeschedulers**
  - **DELETE**: delete collection of KubeScheduler
  - **GET**: list objects of kind KubeScheduler
  - **POST**: create a KubeScheduler
- **/apis/operator.openshift.io/v1/kubeschedulers/{name}**
  - **DELETE**: delete a KubeScheduler
  - **GET**: read the specified KubeScheduler
  - **PATCH**: partially update the specified KubeScheduler
  - **PUT**: replace the specified KubeScheduler
- **/apis/operator.openshift.io/v1/kubeschedulers/{name}/status**
  - **GET**: read status of the specified KubeScheduler
  - **PATCH**: partially update status of the specified KubeScheduler
  - **PUT**: replace status of the specified KubeScheduler

19.2.1. /apis/operator.openshift.io/v1/kubeschedulers

HTTP method

DELETE

Description

delete collection of KubeScheduler

Table 19.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind KubeScheduler

Table 19.2. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeSchedulerList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a KubeScheduler

Table 19.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 19.4. Body parameters

Parameter	Type	Description
body	**KubeScheduler** schema	

Table 19.5. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeScheduler** schema
201 - Created	**KubeScheduler** schema
202 - Accepted	**KubeScheduler** schema
401 - Unauthorized	Empty

19.2.2. /apis/operator.openshift.io/v1/kubeschedulers/{name}

Table 19.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the KubeScheduler

HTTP method

DELETE

Description

delete a KubeScheduler

Table 19.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 19.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified KubeScheduler

Table 19.9. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeScheduler** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified KubeScheduler

Table 19.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 19.11. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeScheduler** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified KubeScheduler

Table 19.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 19.13. Body parameters

Parameter	Type	Description
body	**KubeScheduler** schema	

Table 19.14. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeScheduler** schema
201 - Created	**KubeScheduler** schema
401 - Unauthorized	Empty

19.2.3. /apis/operator.openshift.io/v1/kubeschedulers/{name}/status

Table 19.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the KubeScheduler

HTTP method

GET

Description

read status of the specified KubeScheduler

Table 19.16. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeScheduler** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified KubeScheduler

Table 19.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 19.18. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeScheduler** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified KubeScheduler

Table 19.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 19.20. Body parameters

Parameter	Type	Description
body	**KubeScheduler** schema	

Table 19.21. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeScheduler** schema
201 - Created	**KubeScheduler** schema
401 - Unauthorized	Empty

CHAPTER 20. KUBESTORAGEVERSIONMIGRATOR [OPERATOR.OPENSHIFT.IO/V1]

Description

KubeStorageVersionMigrator provides information to configure an operator to manage kube-storage-version-migrator.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

20.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	

Property	Type	Description
status	**object**	

20.1.1. .spec

Description

Type

object

Property	Type	Description
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
managementState	**string**	managementState indicates whether and how the operator should manage the component
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".

Property	Type	Description
unsupportedConfigOverrides	``	unsupportedConfigOverrides overrides the final configuration that was computed by the operator. Red Hat does not support the use of this field. Misuse of this field could lead to unexpected behavior or conflict with other configuration options. Seek guidance from the Red Hat support before using this field. Use of this property blocks cluster upgrades, it must be removed before upgrading your cluster.

20.1.2. .status

Description

Type

object

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status
conditions[]	**object**	OperatorCondition is just the standard condition fields.
generations	**array**	generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.
generations[]	**object**	GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.
latestAvailableRevision	**integer**	latestAvailableRevision is the deploymentID of the most recent deployment
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with

Property	Type	Description
readyReplicas	**integer**	readyReplicas indicates how many replicas are ready and at the desired state
version	**string**	version is the level this availability applies to

20.1.3. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

20.1.4. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	
reason	**string**	

Property	Type	Description
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

20.1.5. .status.generations

Description

generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Type

array

20.1.6. .status.generations[]

Description

GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.

Type

object

Required

- **group**
- **name**
- **namespace**
- **resource**

Property	Type	Description
group	**string**	group is the group of the thing you're tracking
hash	**string**	hash is an optional field set for resources without generation that are content sensitive like secrets and configmaps
lastGeneration	**integer**	lastGeneration is the last generation of the workload controller involved

Property	Type	Description
name	**string**	name is the name of the thing you're tracking
namespace	**string**	namespace is where the thing you're tracking is
resource	**string**	resource is the resource type of the thing you're tracking

20.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/kubestorageversionmigrators**
  - **DELETE**: delete collection of KubeStorageVersionMigrator
  - **GET**: list objects of kind KubeStorageVersionMigrator
  - **POST**: create a KubeStorageVersionMigrator
- **/apis/operator.openshift.io/v1/kubestorageversionmigrators/{name}**
  - **DELETE**: delete a KubeStorageVersionMigrator
  - **GET**: read the specified KubeStorageVersionMigrator
  - **PATCH**: partially update the specified KubeStorageVersionMigrator
  - **PUT**: replace the specified KubeStorageVersionMigrator
- **/apis/operator.openshift.io/v1/kubestorageversionmigrators/{name}/status**
  - **GET**: read status of the specified KubeStorageVersionMigrator
  - **PATCH**: partially update status of the specified KubeStorageVersionMigrator
  - **PUT**: replace status of the specified KubeStorageVersionMigrator

20.2.1. /apis/operator.openshift.io/v1/kubestorageversionmigrators

HTTP method

DELETE

Description

delete collection of KubeStorageVersionMigrator

Table 20.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind KubeStorageVersionMigrator

Table 20.2. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeStorageVersionMigratorList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a KubeStorageVersionMigrator

Table 20.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 20.4. Body parameters

Parameter	Type	Description
body	**KubeStorageVersionMigrator** schema	

Table 20.5. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeStorageVersionMigrator** schema
201 - Created	**KubeStorageVersionMigrator** schema
202 - Accepted	**KubeStorageVersionMigrator** schema
401 - Unauthorized	Empty

20.2.2. /apis/operator.openshift.io/v1/kubestorageversionmigrators/{name}

Table 20.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the KubeStorageVersionMigrator

HTTP method

DELETE

Description

delete a KubeStorageVersionMigrator

Table 20.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 20.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified KubeStorageVersionMigrator

Table 20.9. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeStorageVersionMigrator** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified KubeStorageVersionMigrator

Table 20.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 20.11. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeStorageVersionMigrator** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified KubeStorageVersionMigrator

Table 20.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 20.13. Body parameters

Parameter	Type	Description
body	**KubeStorageVersionMigrator** schema	

Table 20.14. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeStorageVersionMigrator** schema
201 - Created	**KubeStorageVersionMigrator** schema
401 - Unauthorized	Empty

20.2.3. /apis/operator.openshift.io/v1/kubestorageversionmigrators/{name}/status

Table 20.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the KubeStorageVersionMigrator

HTTP method

GET

Description

read status of the specified KubeStorageVersionMigrator

Table 20.16. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeStorageVersionMigrator** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified KubeStorageVersionMigrator

Table 20.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 20.18. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeStorageVersionMigrator** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified KubeStorageVersionMigrator

Table 20.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 20.20. Body parameters

Parameter	Type	Description
body	**KubeStorageVersionMigrator** schema	

Table 20.21. HTTP responses

HTTP code	Reponse body
200 - OK	**KubeStorageVersionMigrator** schema
201 - Created	**KubeStorageVersionMigrator** schema
401 - Unauthorized	Empty

CHAPTER 21. MACHINECONFIGURATION [OPERATOR.OPENSHIFT.IO/V1]

Description

MachineConfiguration provides information to configure an operator to manage Machine Configuration.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

21.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	spec is the specification of the desired behavior of the Machine Config Operator
status	**object**	status is the most recently observed status of the Machine Config Operator

21.1.1. .spec

Description

spec is the specification of the desired behavior of the Machine Config Operator

Type

object

Property	Type	Description
failedRevisionLimit	**integer**	failedRevisionLimit is the number of failed static pod installer revisions to keep on disk and in the api -1 = unlimited, 0 or unset = 5 (default)
forceRedeploymentReason	**string**	forceRedeploymentReason can be used to force the redeployment of the operand by providing a unique string. This provides a mechanism to kick a previously failed deployment and provide a reason why you think it will work this time instead of failing again on the same config.
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".

Property	Type	Description
managedBootImages	**object**	managedBootImages allows configuration for the management of boot images for machine resources within the cluster. This configuration allows users to select resources that should be updated to the latest boot images during cluster upgrades, ensuring that new machines always boot with the current cluster version's boot image. When omitted, this means no opinion and the platform is left to choose a reasonable default, which is subject to change over time. The default for each machine manager mode is All for GCP and AWS platforms, and None for all other platforms.
managementState	**string**	managementState indicates whether and how the operator should manage the component
nodeDisruptionPolicy	**object**	nodeDisruptionPolicy allows an admin to set granular node disruption actions for MachineConfig-based updates, such as drains, service reloads, etc. Specifying this will allow for less downtime when doing small configuration updates to the cluster. This configuration has no effect on cluster upgrades which will still incur node disruption where required.
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator

Property	Type	Description
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
succeededRevisionLimit	**integer**	succeededRevisionLimit is the number of successful static pod installer revisions to keep on disk and in the api -1 = unlimited, 0 or unset = 5 (default)
unsupportedConfigOverrides	``	unsupportedConfigOverrides overrides the final configuration that was computed by the operator. Red Hat does not support the use of this field. Misuse of this field could lead to unexpected behavior or conflict with other configuration options. Seek guidance from the Red Hat support before using this field. Use of this property blocks cluster upgrades, it must be removed before upgrading your cluster.

21.1.2. .spec.managedBootImages

Description

managedBootImages allows configuration for the management of boot images for machine resources within the cluster. This configuration allows users to select resources that should be updated to the latest boot images during cluster upgrades, ensuring that new machines always boot with the current cluster version’s boot image. When omitted, this means no opinion and the platform is left to choose a reasonable default, which is subject to change over time. The default for each machine manager mode is All for GCP and AWS platforms, and None for all other platforms.

Type

object

Property	Type	Description

Property	Type	Description
machineManagers	**array**	machineManagers can be used to register machine management resources for boot image updates. The Machine Config Operator will watch for changes to this list. Only one entry is permitted per type of machine management resource.
machineManagers[]	**object**	MachineManager describes a target machine resource that is registered for boot image updates. It stores identifying information such as the resource type and the API Group of the resource. It also provides granular control via the selection field.

21.1.3. .spec.managedBootImages.machineManagers

Description

machineManagers can be used to register machine management resources for boot image updates. The Machine Config Operator will watch for changes to this list. Only one entry is permitted per type of machine management resource.

Type

array

21.1.4. .spec.managedBootImages.machineManagers[]

Description

MachineManager describes a target machine resource that is registered for boot image updates. It stores identifying information such as the resource type and the API Group of the resource. It also provides granular control via the selection field.

Type

object

Required

- **apiGroup**
- **resource**
- **selection**

Property	Type	Description
apiGroup	**string**	apiGroup is name of the APIGroup that the machine management resource belongs to. The only current valid value is machine.openshift.io. machine.openshift.io means that the machine manager will only register resources that belong to OpenShift machine API group.
resource	**string**	resource is the machine management resource's type. The only current valid value is machinesets. machinesets means that the machine manager will only register resources of the kind MachineSet.
selection	**object**	selection allows granular control of the machine management resources that will be registered for boot image updates.

21.1.5. .spec.managedBootImages.machineManagers[].selection

Description

selection allows granular control of the machine management resources that will be registered for boot image updates.

Type

object

Required

- **mode**

Property	Type	Description

Property	Type	Description
mode	**string**	mode determines how machine managers will be selected for updates. Valid values are All and Partial. All means that every resource matched by the machine manager will be updated. Partial requires specified selector(s) and allows customisation of which resources matched by the machine manager will be updated. None means that every resource matched by the machine manager will not be updated.
partial	**object**	partial provides label selector(s) that can be used to match machine management resources. Only permitted when mode is set to "Partial".

21.1.6. .spec.managedBootImages.machineManagers[].selection.partial

Description

partial provides label selector(s) that can be used to match machine management resources. Only permitted when mode is set to "Partial".

Type

object

Required

- **machineResourceSelector**

Property	Type	Description
machineResourceSelector	**object**	machineResourceSelector is a label selector that can be used to select machine resources like MachineSets.

21.1.7. .spec.managedBootImages.machineManagers[].selection.partial.machineResou

Description

machineResourceSelector is a label selector that can be used to select machine resources like MachineSets.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

21.1.8. .spec.managedBootImages.machineManagers[].selection.partial.machineResou

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

21.1.9. .spec.managedBootImages.machineManagers[].selection.partial.machineResou

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.

Property	Type	Description
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

21.1.10. .spec.nodeDisruptionPolicy

Description

nodeDisruptionPolicy allows an admin to set granular node disruption actions for MachineConfig-based updates, such as drains, service reloads, etc. Specifying this will allow for less downtime when doing small configuration updates to the cluster. This configuration has no effect on cluster upgrades which will still incur node disruption where required.

Type

object

Property	Type	Description
files	**array**	files is a list of MachineConfig file definitions and actions to take to changes on those paths This list supports a maximum of 50 entries.
files[]	**object**	NodeDisruptionPolicySpecFile is a file entry and corresponding actions to take and is used in the NodeDisruptionPolicyConfig object
sshkey	**object**	sshkey maps to the ignition.sshkeys field in the MachineConfig object, definition an action for this will apply to all sshkey changes in the cluster

Property	Type	Description
units	**array**	units is a list MachineConfig unit definitions and actions to take on changes to those services This list supports a maximum of 50 entries.
units[]	**object**	NodeDisruptionPolicySpecUnit is a systemd unit name and corresponding actions to take and is used in the NodeDisruptionPolicyConfig object

21.1.11. .spec.nodeDisruptionPolicy.files

Description

files is a list of MachineConfig file definitions and actions to take to changes on those paths This list supports a maximum of 50 entries.

Type

array

21.1.12. .spec.nodeDisruptionPolicy.files[]

Description

NodeDisruptionPolicySpecFile is a file entry and corresponding actions to take and is used in the NodeDisruptionPolicyConfig object

Type

object

Required

- **actions**
- **path**

Property	Type	Description

Property	Type	Description
actions	**array**	actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.
actions[]	**object**	
path	**string**	path is the location of a file being managed through a MachineConfig. The Actions in the policy will apply to changes to the file at this path.

21.1.13. .spec.nodeDisruptionPolicy.files[].actions

Description

actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.

Type

array

21.1.14. .spec.nodeDisruptionPolicy.files[].actions[]

Description

Type

object

Required

- **type**

Property	Type	Description
reload	**object**	reload specifies the service to reload, only valid if type is reload
restart	**object**	restart specifies the service to restart, only valid if type is restart
type	**string**	type represents the commands that will be carried out if this NodeDisruptionPolicySpecActionType is executed Valid values are Reboot, Drain, Reload, Restart, DaemonReload and None. reload/restart requires a corresponding service target specified in the reload/restart field. Other values require no further configuration

21.1.15. .spec.nodeDisruptionPolicy.files[].actions[].reload

Description

reload specifies the service to reload, only valid if type is reload

Type

object

Required

- **serviceName**

Property	Type	Description
serviceName	**string**	serviceName is the full name (e.g. crio.service) of the service to be reloaded Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "\". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

21.1.16. .spec.nodeDisruptionPolicy.files[].actions[].restart

Description

restart specifies the service to restart, only valid if type is restart

Type

object

Required

- **serviceName**

Property	Type	Description
serviceName	**string**	serviceName is the full name (e.g. crio.service) of the service to be restarted Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "\". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

21.1.17. .spec.nodeDisruptionPolicy.sshkey

Description

sshkey maps to the ignition.sshkeys field in the MachineConfig object, definition an action for this will apply to all sshkey changes in the cluster

Type

object

Required

- **actions**

Property	Type	Description

Property	Type	Description
actions	**array**	actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.
actions[]	**object**	

21.1.18. .spec.nodeDisruptionPolicy.sshkey.actions

Description

actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.

Type

array

21.1.19. .spec.nodeDisruptionPolicy.sshkey.actions[]

Description

Type

object

Required

- **type**

Property	Type	Description
reload	**object**	reload specifies the service to reload, only valid if type is reload

Property	Type	Description
restart	**object**	restart specifies the service to restart, only valid if type is restart
type	**string**	type represents the commands that will be carried out if this NodeDisruptionPolicySpecActionType is executed Valid values are Reboot, Drain, Reload, Restart, DaemonReload and None. reload/restart requires a corresponding service target specified in the reload/restart field. Other values require no further configuration

21.1.20. .spec.nodeDisruptionPolicy.sshkey.actions[].reload

Description

reload specifies the service to reload, only valid if type is reload

Type

object

Required

- **serviceName**

Property	Type	Description
serviceName	**string**	serviceName is the full name (e.g. crio.service) of the service to be reloaded Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "\". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

21.1.21. .spec.nodeDisruptionPolicy.sshkey.actions[].restart

Description

restart specifies the service to restart, only valid if type is restart

Type

object

Required

- **serviceName**

Property	Type	Description
serviceName	**string**	serviceName is the full name (e.g. crio.service) of the service to be restarted Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "\". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

21.1.22. .spec.nodeDisruptionPolicy.units

Description

units is a list MachineConfig unit definitions and actions to take on changes to those services This list supports a maximum of 50 entries.

Type

array

21.1.23. .spec.nodeDisruptionPolicy.units[]

Description

NodeDisruptionPolicySpecUnit is a systemd unit name and corresponding actions to take and is used in the NodeDisruptionPolicyConfig object

Type

object

Required

- **actions**
- **name**

Property	Type	Description
actions	**array**	actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.
actions[]	**object**	
name	**string**	name represents the service name of a systemd service managed through a MachineConfig Actions specified will be applied for changes to the named service. Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "\". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

21.1.24. .spec.nodeDisruptionPolicy.units[].actions

Description

actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.

Type

array

21.1.25. .spec.nodeDisruptionPolicy.units[].actions[]

Description

Type

object

Required

- **type**

Property	Type	Description
reload	**object**	reload specifies the service to reload, only valid if type is reload
restart	**object**	restart specifies the service to restart, only valid if type is restart
type	**string**	type represents the commands that will be carried out if this NodeDisruptionPolicySpecActionType is executed Valid values are Reboot, Drain, Reload, Restart, DaemonReload and None. reload/restart requires a corresponding service target specified in the reload/restart field. Other values require no further configuration

21.1.26. .spec.nodeDisruptionPolicy.units[].actions[].reload

Description

reload specifies the service to reload, only valid if type is reload

Type

object

Required

- **serviceName**

Property	Type	Description

Property	Type	Description
serviceName	**string**	serviceName is the full name (e.g. crio.service) of the service to be reloaded Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "\". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

21.1.27. .spec.nodeDisruptionPolicy.units[].actions[].restart

Description

restart specifies the service to restart, only valid if type is restart

Type

object

Required

- **serviceName**

Property	Type	Description
serviceName	**string**	serviceName is the full name (e.g. crio.service) of the service to be restarted Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "\". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

21.1.28. .status

Description

status is the most recently observed status of the Machine Config Operator

Type

object

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status
conditions[]	**object**	Condition contains details for one aspect of the current state of this API Resource.
managedBootImagesStatus	**object**	managedBootImagesStatus reflects what the latest cluster-validated boot image configuration is and will be used by Machine Config Controller while performing boot image updates.
nodeDisruptionPolicyStatus	**object**	nodeDisruptionPolicyStatus status reflects what the latest cluster-validated policies are, and will be used by the Machine Config Daemon during future node updates.
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with

21.1.29. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

21.1.30. .status.conditions[]

Description

Condition contains details for one aspect of the current state of this API Resource.

Type

object

Required

- **lastTransitionTime**

- **message**
- **reason**
- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	message is a human readable message indicating details about the transition. This may be an empty string.
observedGeneration	**integer**	observedGeneration represents the .metadata.generation that the condition was set based upon. For instance, if .metadata.generation is currently 12, but the .status.conditions[x].observedGeneration is 9, the condition is out of date with respect to the current state of the instance.
reason	**string**	reason contains a programmatic identifier indicating the reason for the condition's last transition. Producers of specific condition types may define expected values and meanings for this field, and whether the values are considered a guaranteed API. The value should be a CamelCase string. This field may not be empty.
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

21.1.31. .status.managedBootImagesStatus

Description

managedBootImagesStatus reflects what the latest cluster-validated boot image configuration is and will be used by Machine Config Controller while performing boot image updates.

Type

object

Property	Type	Description
machineManagers	**array**	machineManagers can be used to register machine management resources for boot image updates. The Machine Config Operator will watch for changes to this list. Only one entry is permitted per type of machine management resource.
machineManagers[]	**object**	MachineManager describes a target machine resource that is registered for boot image updates. It stores identifying information such as the resource type and the API Group of the resource. It also provides granular control via the selection field.

21.1.32. .status.managedBootImagesStatus.machineManagers

Description

machineManagers can be used to register machine management resources for boot image updates. The Machine Config Operator will watch for changes to this list. Only one entry is permitted per type of machine management resource.

Type

array

21.1.33. .status.managedBootImagesStatus.machineManagers[]

Description

MachineManager describes a target machine resource that is registered for boot image updates. It stores identifying information such as the resource type and the API Group of the resource. It also provides granular control via the selection field.

Type

object

Required

- **apiGroup**
- **resource**
- **selection**

Property	Type	Description
apiGroup	**string**	apiGroup is name of the APIGroup that the machine management resource belongs to. The only current valid value is machine.openshift.io. machine.openshift.io means that the machine manager will only register resources that belong to OpenShift machine API group.
resource	**string**	resource is the machine management resource's type. The only current valid value is machinesets. machinesets means that the machine manager will only register resources of the kind MachineSet.
selection	**object**	selection allows granular control of the machine management resources that will be registered for boot image updates.

21.1.34. .status.managedBootImagesStatus.machineManagers[].selection

Description

selection allows granular control of the machine management resources that will be registered for boot image updates.

Type

object

Required

- **mode**

Property	Type	Description

Property	Type	Description
mode	**string**	mode determines how machine managers will be selected for updates. Valid values are All and Partial. All means that every resource matched by the machine manager will be updated. Partial requires specified selector(s) and allows customisation of which resources matched by the machine manager will be updated. None means that every resource matched by the machine manager will not be updated.
partial	**object**	partial provides label selector(s) that can be used to match machine management resources. Only permitted when mode is set to "Partial".

21.1.35. .status.managedBootImagesStatus.machineManagers[].selection.partial

Description

partial provides label selector(s) that can be used to match machine management resources. Only permitted when mode is set to "Partial".

Type

object

Required

- **machineResourceSelector**

Property	Type	Description
machineResourceSelector	**object**	machineResourceSelector is a label selector that can be used to select machine resources like MachineSets.

21.1.36. .status.managedBootImagesStatus.machineManagers[].selection.partial.mach

Description

machineResourceSelector is a label selector that can be used to select machine resources like MachineSets.

Type

object

Property	Type	Description
matchExpressions	**array**	matchExpressions is a list of label selector requirements. The requirements are ANDed.
matchExpressions[]	**object**	A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchLabels	**object (string)**	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

21.1.37. .status.managedBootImagesStatus.machineManagers[].selection.partial.mach

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

21.1.38. .status.managedBootImagesStatus.machineManagers[].selection.partial.mach

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

- **key**
- **operator**

Property	Type	Description
key	**string**	key is the label key that the selector applies to.

Property	Type	Description
operator	**string**	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	**array (string)**	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

21.1.39. .status.nodeDisruptionPolicyStatus

Description

nodeDisruptionPolicyStatus status reflects what the latest cluster-validated policies are, and will be used by the Machine Config Daemon during future node updates.

Type

object

Property	Type	Description
clusterPolicies	**object**	clusterPolicies is a merge of cluster default and user provided node disruption policies.

21.1.40. .status.nodeDisruptionPolicyStatus.clusterPolicies

Description

clusterPolicies is a merge of cluster default and user provided node disruption policies.

Type

object

Property	Type	Description
files	**array**	files is a list of MachineConfig file definitions and actions to take to changes on those paths

Property	Type	Description
files[]	**object**	NodeDisruptionPolicyStatusFile is a file entry and corresponding actions to take and is used in the NodeDisruptionPolicyClusterStatus object
sshkey	**object**	sshkey is the overall sshkey MachineConfig definition
units	**array**	units is a list MachineConfig unit definitions and actions to take on changes to those services
units[]	**object**	NodeDisruptionPolicyStatusUnit is a systemd unit name and corresponding actions to take and is used in the NodeDisruptionPolicyClusterStatus object

21.1.41. .status.nodeDisruptionPolicyStatus.clusterPolicies.files

Description

files is a list of MachineConfig file definitions and actions to take to changes on those paths

Type

array

21.1.42. .status.nodeDisruptionPolicyStatus.clusterPolicies.files[]

Description

NodeDisruptionPolicyStatusFile is a file entry and corresponding actions to take and is used in the NodeDisruptionPolicyClusterStatus object

Type

object

Required

- **actions**
- **path**

Property	Type	Description

Property	Type	Description
actions	**array**	actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.
actions[]	**object**	
path	**string**	path is the location of a file being managed through a MachineConfig. The Actions in the policy will apply to changes to the file at this path.

21.1.43. .status.nodeDisruptionPolicyStatus.clusterPolicies.files[].actions

Description

actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.

Type

array

21.1.44. .status.nodeDisruptionPolicyStatus.clusterPolicies.files[].actions[]

Description

Type

object

Required

- **type**

Property	Type	Description
reload	**object**	reload specifies the service to reload, only valid if type is reload
restart	**object**	restart specifies the service to restart, only valid if type is restart
type	**string**	type represents the commands that will be carried out if this NodeDisruptionPolicyStatusActionType is executed Valid values are Reboot, Drain, Reload, Restart, DaemonReload, None and Special. reload/restart requires a corresponding service target specified in the reload/restart field. Other values require no further configuration

21.1.45. .status.nodeDisruptionPolicyStatus.clusterPolicies.files[].actions[].reload

Description

reload specifies the service to reload, only valid if type is reload

Type

object

Required

- **serviceName**

Property	Type	Description
serviceName	**string**	serviceName is the full name (e.g. crio.service) of the service to be reloaded Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "\". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

21.1.46. .status.nodeDisruptionPolicyStatus.clusterPolicies.files[].actions[].restart

Description

restart specifies the service to restart, only valid if type is restart

Type

object

Required

- **serviceName**

Property	Type	Description
serviceName	**string**	serviceName is the full name (e.g. crio.service) of the service to be restarted Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "\". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

21.1.47. .status.nodeDisruptionPolicyStatus.clusterPolicies.sshkey

Description

sshkey is the overall sshkey MachineConfig definition

Type

object

Required

- **actions**

Property	Type	Description

Property	Type	Description
actions	**array**	actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.
actions[]	**object**	

21.1.48. .status.nodeDisruptionPolicyStatus.clusterPolicies.sshkey.actions

Description

actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.

Type

array

21.1.49. .status.nodeDisruptionPolicyStatus.clusterPolicies.sshkey.actions[]

Description

Type

object

Required

- **type**

Property	Type	Description
reload	**object**	reload specifies the service to reload, only valid if type is reload

Property	Type	Description
restart	**object**	restart specifies the service to restart, only valid if type is restart
type	**string**	type represents the commands that will be carried out if this NodeDisruptionPolicyStatusActionType is executed Valid values are Reboot, Drain, Reload, Restart, DaemonReload, None and Special. reload/restart requires a corresponding service target specified in the reload/restart field. Other values require no further configuration

21.1.50. .status.nodeDisruptionPolicyStatus.clusterPolicies.sshkey.actions[].reload

Description

reload specifies the service to reload, only valid if type is reload

Type

object

Required

- **serviceName**

Property	Type	Description
serviceName	**string**	serviceName is the full name (e.g. crio.service) of the service to be reloaded Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "\". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

21.1.51. .status.nodeDisruptionPolicyStatus.clusterPolicies.sshkey.actions[].restart

Description

restart specifies the service to restart, only valid if type is restart

Type

object

Required

- **serviceName**

Property	Type	Description
serviceName	**string**	serviceName is the full name (e.g. crio.service) of the service to be restarted Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "\". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

21.1.52. .status.nodeDisruptionPolicyStatus.clusterPolicies.units

Description

units is a list MachineConfig unit definitions and actions to take on changes to those services

Type

array

21.1.53. .status.nodeDisruptionPolicyStatus.clusterPolicies.units[]

Description

NodeDisruptionPolicyStatusUnit is a systemd unit name and corresponding actions to take and is used in the NodeDisruptionPolicyClusterStatus object

Type

object

Required

- **actions**
- **name**

Property	Type	Description

Property	Type	Description
actions	**array**	actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot, the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.
actions[]	**object**	
name	**string**	name represents the service name of a systemd service managed through a MachineConfig Actions specified will be applied for changes to the named service. Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "\". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

21.1.54. .status.nodeDisruptionPolicyStatus.clusterPolicies.units[].actions

Description

actions represents the series of commands to be executed on changes to the file at the corresponding file path. Actions will be applied in the order that they are set in this list. If there are other incoming changes to other MachineConfig entries in the same update that require a reboot,

the reboot will supercede these actions. Valid actions are Reboot, Drain, Reload, DaemonReload and None. The Reboot action and the None action cannot be used in conjunction with any of the other actions. This list supports a maximum of 10 entries.

Type

array

21.1.55. .status.nodeDisruptionPolicyStatus.clusterPolicies.units[].actions[]

Description

Type

object

Required

- **type**

Property	Type	Description
reload	**object**	reload specifies the service to reload, only valid if type is reload
restart	**object**	restart specifies the service to restart, only valid if type is restart
type	**string**	type represents the commands that will be carried out if this NodeDisruptionPolicyStatusActionType is executed Valid values are Reboot, Drain, Reload, Restart, DaemonReload, None and Special. reload/restart requires a corresponding service target specified in the reload/restart field. Other values require no further configuration

21.1.56. .status.nodeDisruptionPolicyStatus.clusterPolicies.units[].actions[].reload

Description

reload specifies the service to reload, only valid if type is reload

Type

object

Required

- **serviceName**

Property	Type	Description
serviceName	**string**	serviceName is the full name (e.g. crio.service) of the service to be reloaded Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "\". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

21.1.57. .status.nodeDisruptionPolicyStatus.clusterPolicies.units[].actions[].restart

Description

restart specifies the service to restart, only valid if type is restart

Type

object

Required

- **serviceName**

Property	Type	Description
serviceName	**string**	serviceName is the full name (e.g. crio.service) of the service to be restarted Service names should be of the format ${NAME}${SERVICETYPE} and can up to 255 characters long. ${NAME} must be atleast 1 character long and can only consist of alphabets, digits, ":", "-", "_", ".", and "\". ${SERVICETYPE} must be one of ".service", ".socket", ".device", ".mount", ".automount", ".swap", ".target", ".path", ".timer", ".snapshot", ".slice" or ".scope".

21.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/machineconfigurations**
  - **DELETE**: delete collection of MachineConfiguration
  - **GET**: list objects of kind MachineConfiguration
  - **POST**: create a MachineConfiguration
- **/apis/operator.openshift.io/v1/machineconfigurations/{name}**
  - **DELETE**: delete a MachineConfiguration
  - **GET**: read the specified MachineConfiguration
  - **PATCH**: partially update the specified MachineConfiguration
  - **PUT**: replace the specified MachineConfiguration
- **/apis/operator.openshift.io/v1/machineconfigurations/{name}/status**
  - **GET**: read status of the specified MachineConfiguration
  - **PATCH**: partially update status of the specified MachineConfiguration
  - **PUT**: replace status of the specified MachineConfiguration

21.2.1. /apis/operator.openshift.io/v1/machineconfigurations

HTTP method

DELETE

Description

delete collection of MachineConfiguration

Table 21.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind MachineConfiguration

Table 21.2. HTTP responses

HTTP code	Reponse body
200 - OK	**MachineConfigurationList** schema

HTTP code	Reponse body
401 - Unauthorized	Empty

HTTP method

POST

Description

create a MachineConfiguration

Table 21.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 21.4. Body parameters

Parameter	Type	Description
body	**MachineConfiguration** schema	

Table 21.5. HTTP responses

HTTP code	Reponse body
200 - OK	**MachineConfiguration** schema
201 - Created	**MachineConfiguration** schema
202 - Accepted	**MachineConfiguration** schema
401 - Unauthorized	Empty

21.2.2. /apis/operator.openshift.io/v1/machineconfigurations/{name}

Table 21.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the MachineConfiguration

HTTP method

DELETE

Description

delete a MachineConfiguration

Table 21.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 21.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified MachineConfiguration

Table 21.9. HTTP responses

HTTP code	Reponse body
200 - OK	**MachineConfiguration** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified MachineConfiguration

Table 21.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 21.11. HTTP responses

HTTP code	Reponse body
200 - OK	**MachineConfiguration** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified MachineConfiguration

Table 21.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 21.13. Body parameters

Parameter	Type	Description
body	**MachineConfiguration** schema	

Table 21.14. HTTP responses

HTTP code	Reponse body
200 - OK	**MachineConfiguration** schema
201 - Created	**MachineConfiguration** schema
401 - Unauthorized	Empty

21.2.3. /apis/operator.openshift.io/v1/machineconfigurations/{name}/status

Table 21.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the MachineConfiguration

HTTP method

GET

Description

read status of the specified MachineConfiguration

Table 21.16. HTTP responses

HTTP code	Reponse body
200 - OK	**MachineConfiguration** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified MachineConfiguration

Table 21.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 21.18. HTTP responses

HTTP code	Reponse body
200 - OK	**MachineConfiguration** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified MachineConfiguration

Table 21.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 21.20. Body parameters

Parameter	Type	Description
body	**MachineConfiguration** schema	

Table 21.21. HTTP responses

HTTP code	Reponse body
200 - OK	**MachineConfiguration** schema
201 - Created	**MachineConfiguration** schema
401 - Unauthorized	Empty

CHAPTER 22. NETWORK [OPERATOR.OPENSHIFT.IO/V1]

Description

Network describes the cluster's desired network configuration. It is consumed by the cluster-network-operator.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

22.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	NetworkSpec is the top-level network configuration object.

Property	Type	Description
status	**object**	NetworkStatus is detailed operator status, which is distilled up to the Network clusteroperator object.

22.1.1. .spec

Description

NetworkSpec is the top-level network configuration object.

Type

object

Property	Type	Description
additionalNetworks	**array**	additionalNetworks is a list of extra networks to make available to pods when multiple networks are enabled.
additionalNetworks[]	**object**	AdditionalNetworkDefinition configures an extra network that is available but not created by default. Instead, pods must request them by name. type must be specified, along with exactly one "Config" that matches the type.
additionalRoutingCapabilities	**object**	additionalRoutingCapabilities describes components and relevant configuration providing additional routing capabilities. When set, it enables such components and the usage of the routing capabilities they provide for the machine network. Upstream operators, like MetalLB operator, requiring these capabilities may rely on, or automatically set this attribute. Network plugins may leverage advanced routing capabilities acquired through the enablement of these components but may require specific configuration on their side to do so; refer to their respective documentation and configuration options.

Property	Type	Description
clusterNetwork	**array**	clusterNetwork is the IP address pool to use for pod IPs. Some network providers support multiple ClusterNetworks. Others only support one. This is equivalent to the cluster-cidr.
clusterNetwork[]	**object**	ClusterNetworkEntry is a subnet from which to allocate PodIPs. A network of size HostPrefix (in CIDR notation) will be allocated when nodes join the cluster. If the HostPrefix field is not used by the plugin, it can be left unset. Not all network providers support multiple ClusterNetworks
defaultNetwork	**object**	defaultNetwork is the "default" network that all pods will receive
deployKubeProxy	**boolean**	deployKubeProxy specifies whether or not a standalone kube-proxy should be deployed by the operator. Some network providers include kube-proxy or similar functionality. If unset, the plugin will attempt to select the correct value, which is false when ovn-kubernetes is used and true otherwise.
disableMultiNetwork	**boolean**	disableMultiNetwork defaults to 'false' and this setting enables the pod multi-networking capability. disableMultiNetwork when set to 'true' at cluster install time does not install the components, typically the Multus CNI and the network-attachment-definition CRD, that enable the pod multi-networking capability. Setting the parameter to 'true' might be useful when you need install third-party CNI plugins, but these plugins are not supported by Red Hat. Changing the parameter value as a postinstallation cluster task has no effect.

Property	Type	Description
disableNetworkDiagnostics	**boolean**	disableNetworkDiagnostics specifies whether or not PodNetworkConnectivityCheck CRs from a test pod to every node, apiserver and LB should be disabled or not. If unset, this property defaults to 'false' and network diagnostics is enabled. Setting this to 'true' would reduce the additional load of the pods performing the checks.
exportNetworkFlows	**object**	exportNetworkFlows enables and configures the export of network flow metadata from the pod network by using protocols NetFlow, SFlow or IPFIX. Currently only supported on OVN-Kubernetes plugin. If unset, flows will not be exported to any collector.
kubeProxyConfig	**object**	kubeProxyConfig lets us configure desired proxy configuration, if deployKubeProxy is true. If not specified, sensible defaults will be chosen by OpenShift directly.
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
managementState	**string**	managementState indicates whether and how the operator should manage the component
migration	**object**	migration enables and configures cluster network migration, for network changes that cannot be made instantly.

Property	Type	Description
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
serviceNetwork	**array (string)**	serviceNetwork is the ip address pool to use for Service IPs Currently, all existing network providers only support a single value here, but this is an array to allow for growth.
unsupportedConfigOverrides	``	unsupportedConfigOverrides overrides the final configuration that was computed by the operator. Red Hat does not support the use of this field. Misuse of this field could lead to unexpected behavior or conflict with other configuration options. Seek guidance from the Red Hat support before using this field. Use of this property blocks cluster upgrades, it must be removed before upgrading your cluster.

Property	Type	Description
useMultiNetworkPolicy	**boolean**	useMultiNetworkPolicy enables a controller which allows for MultiNetworkPolicy objects to be used on additional networks as created by Multus CNI. MultiNetworkPolicy are similar to NetworkPolicy objects, but NetworkPolicy objects only apply to the primary interface. With MultiNetworkPolicy, you can control the traffic that a pod can receive over the secondary interfaces. If unset, this property defaults to 'false' and MultiNetworkPolicy objects are ignored. If 'disableMultiNetwork' is 'true' then the value of this field is ignored.

22.1.2. .spec.additionalNetworks

Description

additionalNetworks is a list of extra networks to make available to pods when multiple networks are enabled.

Type

array

22.1.3. .spec.additionalNetworks[]

Description

AdditionalNetworkDefinition configures an extra network that is available but not created by default. Instead, pods must request them by name. type must be specified, along with exactly one "Config" that matches the type.

Type

object

Required

- **name**

Property	Type	Description
name	**string**	name is the name of the network. This will be populated in the resulting CRD This must be unique.

Property	Type	Description
namespace	**string**	namespace is the namespace of the network. This will be populated in the resulting CRD If not given the network will be created in the default namespace.
rawCNIConfig	**string**	rawCNIConfig is the raw CNI configuration json to create in the NetworkAttachmentDefinition CRD
simpleMacvlanConfig	**object**	simpleMacvlanConfig configures the macvlan interface in case of type:NetworkTypeSimpleMacvlan
type	**string**	type is the type of network The supported values are NetworkTypeRaw, NetworkTypeSimpleMacvlan

22.1.4. .spec.additionalNetworks[].simpleMacvlanConfig

Description

simpleMacvlanConfig configures the macvlan interface in case of type:NetworkTypeSimpleMacvlan

Type

object

Property	Type	Description
ipamConfig	**object**	ipamConfig configures IPAM module will be used for IP Address Management (IPAM).
master	**string**	master is the host interface to create the macvlan interface from. If not specified, it will be default route interface
mode	**string**	mode is the macvlan mode: bridge, private, vepa, passthru. The default is bridge
mtu	**integer**	mtu is the mtu to use for the macvlan interface. if unset, host's kernel will select the value.

22.1.5. .spec.additionalNetworks[].simpleMacvlanConfig.ipamConfig

Description

ipamConfig configures IPAM module will be used for IP Address Management (IPAM).

Type

object

Property	Type	Description
staticIPAMConfig	**object**	staticIPAMConfig configures the static IP address in case of type:IPAMTypeStatic
type	**string**	type is the type of IPAM module will be used for IP Address Management(IPAM). The supported values are IPAMTypeDHCP, IPAMTypeStatic

22.1.6. .spec.additionalNetworks[].simpleMacvlanConfig.ipamConfig.staticIPAMConfig

Description

staticIPAMConfig configures the static IP address in case of type:IPAMTypeStatic

Type

object

Property	Type	Description
addresses	**array**	addresses configures IP address for the interface
addresses[]	**object**	StaticIPAMAddresses provides IP address and Gateway for static IPAM addresses
dns	**object**	dns configures DNS for the interface
routes	**array**	routes configures IP routes for the interface
routes[]	**object**	StaticIPAMRoutes provides Destination/Gateway pairs for static IPAM routes

22.1.7. .spec.additionalNetworks[].simpleMacvlanConfig.ipamConfig.staticIPAMConfig

Description

addresses configures IP address for the interface

Type

array

22.1.8. .spec.additionalNetworks[].simpleMacvlanConfig.ipamConfig.staticIPAMConfig

Description

StaticIPAMAddresses provides IP address and Gateway for static IPAM addresses

Type

object

Property	Type	Description
address	**string**	address is the IP address in CIDR format
gateway	**string**	gateway is IP inside of subnet to designate as the gateway

22.1.9. .spec.additionalNetworks[].simpleMacvlanConfig.ipamConfig.staticIPAMConfig

Description

dns configures DNS for the interface

Type

object

Property	Type	Description
domain	**string**	domain configures the domainname the local domain used for short hostname lookups
nameservers	**array (string)**	nameservers points DNS servers for IP lookup
search	**array (string)**	search configures priority ordered search domains for short hostname lookups

22.1.10. .spec.additionalNetworks[].simpleMacvlanConfig.ipamConfig.staticIPAMConfi

Description

routes configures IP routes for the interface

Type

array

22.1.11. .spec.additionalNetworks[].simpleMacvlanConfig.ipamConfig.staticIPAMConfig

Description

StaticIPAMRoutes provides Destination/Gateway pairs for static IPAM routes

Type

object

Property	Type	Description
destination	**string**	destination points the IP route destination
gateway	**string**	gateway is the route's next-hop IP address If unset, a default gateway is assumed (as determined by the CNI plugin).

22.1.12. .spec.additionalRoutingCapabilities

Description

additionalRoutingCapabilities describes components and relevant configuration providing additional routing capabilities. When set, it enables such components and the usage of the routing capabilities they provide for the machine network. Upstream operators, like MetalLB operator, requiring these capabilities may rely on, or automatically set this attribute. Network plugins may leverage advanced routing capabilities acquired through the enablement of these components but may require specific configuration on their side to do so; refer to their respective documentation and configuration options.

Type

object

Required

- **providers**

Property	Type	Description
providers	**array (string)**	providers is a set of enabled components that provide additional routing capabilities. Entries on this list must be unique. The only valid value is currrently "FRR" which provides FRR routing capabilities through the deployment of FRR.

22.1.13. .spec.clusterNetwork

Description

clusterNetwork is the IP address pool to use for pod IPs. Some network providers support multiple ClusterNetworks. Others only support one. This is equivalent to the cluster-cidr.

Type

array

22.1.14. .spec.clusterNetwork[]

Description

ClusterNetworkEntry is a subnet from which to allocate PodIPs. A network of size HostPrefix (in CIDR notation) will be allocated when nodes join the cluster. If the HostPrefix field is not used by the plugin, it can be left unset. Not all network providers support multiple ClusterNetworks

Type

object

Property	Type	Description
cidr	**string**	
hostPrefix	**integer**	

22.1.15. .spec.defaultNetwork

Description

defaultNetwork is the "default" network that all pods will receive

Type

object

Property	Type	Description
openshiftSDNConfig	**object**	openshiftSDNConfig was previously used to configure the openshift-sdn plugin. DEPRECATED: OpenShift SDN is no longer supported.
ovnKubernetesConfig	**object**	ovnKubernetesConfig configures the ovn-kubernetes plugin.
type	**string**	type is the type of network All NetworkTypes are supported except for NetworkTypeRaw

22.1.16. .spec.defaultNetwork.openshiftSDNConfig

Description

openshiftSDNConfig was previously used to configure the openshift-sdn plugin. DEPRECATED: OpenShift SDN is no longer supported.

Type

object

Property	Type	Description
enableUnidling	**boolean**	enableUnidling controls whether or not the service proxy will support idling and unidling of services. By default, unidling is enabled.
mode	**string**	mode is one of "Multitenant", "Subnet", or "NetworkPolicy"
mtu	**integer**	mtu is the mtu to use for the tunnel interface. Defaults to 1450 if unset. This must be 50 bytes smaller than the machine's uplink.
useExternalOpenvswitch	**boolean**	useExternalOpenvswitch used to control whether the operator would deploy an OVS DaemonSet itself or expect someone else to start OVS. As of 4.6, OVS is always run as a system service, and this flag is ignored.
vxlanPort	**integer**	vxlanPort is the port to use for all vxlan packets. The default is 4789.

22.1.17. .spec.defaultNetwork.ovnKubernetesConfig

Description

ovnKubernetesConfig configures the ovn-kubernetes plugin.

Type

object

Property	Type	Description
egressIPConfig	**object**	egressIPConfig holds the configuration for EgressIP options.
gatewayConfig	**object**	gatewayConfig holds the configuration for node gateway options.

Property	Type	Description
genevePort	**integer**	geneve port is the UDP port to be used by geneve encapulation. Default is 6081
hybridOverlayConfig	**object**	hybridOverlayConfig configures an additional overlay network for peers that are not using OVN.
ipsecConfig	**object**	ipsecConfig enables and configures IPsec for pods on the pod network within the cluster.
ipv4	**object**	ipv4 allows users to configure IP settings for IPv4 connections. When ommitted, this means no opinions and the default configuration is used. Check individual fields within ipv4 for details of default values.
ipv6	**object**	ipv6 allows users to configure IP settings for IPv6 connections. When ommitted, this means no opinions and the default configuration is used. Check individual fields within ipv4 for details of default values.
mtu	**integer**	mtu is the MTU to use for the tunnel interface. This must be 100 bytes smaller than the uplink mtu. Default is 1400
policyAuditConfig	**object**	policyAuditConfig is the configuration for network policy audit events. If unset, reported defaults are used.

Property	Type	Description
routeAdvertisements	**string**	routeAdvertisements determines if the functionality to advertise cluster network routes through a dynamic routing protocol, such as BGP, is enabled or not. This functionality is configured through the ovn-kubernetes RouteAdvertisements CRD. Requires the 'FRR' routing capability provider to be enabled as an additional routing capability. Allowed values are "Enabled", "Disabled" and ommited. When omitted, this means the user has no opinion and the platform is left to choose reasonable defaults. These defaults are subject to change over time. The current default is "Disabled".
v4InternalSubnet	**string**	v4InternalSubnet is a v4 subnet used internally by ovn-kubernetes in case the default one is being already used by something else. It must not overlap with any other subnet being used by OpenShift or by the node network. The size of the subnet must be larger than the number of nodes. Default is 100.64.0.0/16
v6InternalSubnet	**string**	v6InternalSubnet is a v6 subnet used internally by ovn-kubernetes in case the default one is being already used by something else. It must not overlap with any other subnet being used by OpenShift or by the node network. The size of the subnet must be larger than the number of nodes. Default is fd98::/64

22.1.18. .spec.defaultNetwork.ovnKubernetesConfig.egressIPConfig

Description

egressIPConfig holds the configuration for EgressIP options.

Type

object

Property	Type	Description
reachabilityTotalTimeoutSeconds	**integer**	reachabilityTotalTimeout configures the EgressIP node reachability check total timeout in seconds. If the EgressIP node cannot be reached within this timeout, the node is declared down. Setting a large value may cause the EgressIP feature to react slowly to node changes. In particular, it may react slowly for EgressIP nodes that really have a genuine problem and are unreachable. When omitted, this means the user has no opinion and the platform is left to choose a reasonable default, which is subject to change over time. The current default is 1 second. A value of 0 disables the EgressIP node's reachability check.

22.1.19. .spec.defaultNetwork.ovnKubernetesConfig.gatewayConfig

Description

gatewayConfig holds the configuration for node gateway options.

Type

object

Property	Type	Description
ipForwarding	**string**	ipForwarding controls IP forwarding for all traffic on OVN-Kubernetes managed interfaces (such as br-ex). By default this is set to Restricted, and Kubernetes related traffic is still forwarded appropriately, but other IP traffic will not be routed by the OCP node. If there is a desire to allow the host to forward traffic across OVN-Kubernetes managed interfaces, then set this field to "Global". The supported values are "Restricted" and "Global".

Property	Type	Description
ipv4	**object**	ipv4 allows users to configure IP settings for IPv4 connections. When omitted, this means no opinion and the default configuration is used. Check individual members fields within ipv4 for details of default values.
ipv6	**object**	ipv6 allows users to configure IP settings for IPv6 connections. When omitted, this means no opinion and the default configuration is used. Check individual members fields within ipv6 for details of default values.
routingViaHost	**boolean**	routingViaHost allows pod egress traffic to exit via the ovn-k8s-mp0 management port into the host before sending it out. If this is not set, traffic will always egress directly from OVN to outside without touching the host stack. Setting this to true means hardware offload will not be supported. Default is false if GatewayConfig is specified.

22.1.20. .spec.defaultNetwork.ovnKubernetesConfig.gatewayConfig.ipv4

Description

ipv4 allows users to configure IP settings for IPv4 connections. When omitted, this means no opinion and the default configuration is used. Check individual members fields within ipv4 for details of default values.

Type

object

Property	Type	Description

Property	Type	Description
internalMasqueradeSubnet	**string**	internalMasqueradeSubnet contains the masquerade addresses in IPV4 CIDR format used internally by ovn-kubernetes to enable host to service traffic. Each host in the cluster is configured with these addresses, as well as the shared gateway bridge interface. The values can be changed after installation. The subnet chosen should not overlap with other networks specified for OVN-Kubernetes as well as other networks used on the host. Additionally the subnet must be large enough to accommodate 6 IPs (maximum prefix length /29). When omitted, this means no opinion and the platform is left to choose a reasonable default which is subject to change over time. The current default subnet is 169.254.0.0/17 The value must be in proper IPV4 CIDR format

22.1.21. .spec.defaultNetwork.ovnKubernetesConfig.gatewayConfig.ipv6

Description

ipv6 allows users to configure IP settings for IPv6 connections. When omitted, this means no opinion and the default configuration is used. Check individual members fields within ipv6 for details of default values.

Type

object

Property	Type	Description

Property	Type	Description
internalMasqueradeSubnet	**string**	internalMasqueradeSubnet contains the masquerade addresses in IPV6 CIDR format used internally by ovn-kubernetes to enable host to service traffic. Each host in the cluster is configured with these addresses, as well as the shared gateway bridge interface. The values can be changed after installation. The subnet chosen should not overlap with other networks specified for OVN-Kubernetes as well as other networks used on the host. Additionally the subnet must be large enough to accommodate 6 IPs (maximum prefix length /125). When omitted, this means no opinion and the platform is left to choose a reasonable default which is subject to change over time. The current default subnet is fd69::/112 Note that IPV6 dual addresses are not permitted

22.1.22. .spec.defaultNetwork.ovnKubernetesConfig.hybridOverlayConfig

Description

hybridOverlayConfig configures an additional overlay network for peers that are not using OVN.

Type

object

Property	Type	Description
hybridClusterNetwork	**array**	hybridClusterNetwork defines a network space given to nodes on an additional overlay network.
hybridClusterNetwork[]	**object**	ClusterNetworkEntry is a subnet from which to allocate PodIPs. A network of size HostPrefix (in CIDR notation) will be allocated when nodes join the cluster. If the HostPrefix field is not used by the plugin, it can be left unset. Not all network providers support multiple ClusterNetworks

Property	Type	Description
hybridOverlayVXLANPort	**integer**	hybridOverlayVXLANPort defines the VXLAN port number to be used by the additional overlay network. Default is 4789

22.1.23. .spec.defaultNetwork.ovnKubernetesConfig.hybridOverlayConfig.hybridClust

Description

hybridClusterNetwork defines a network space given to nodes on an additional overlay network.

Type

array

22.1.24. .spec.defaultNetwork.ovnKubernetesConfig.hybridOverlayConfig.hybridClust

Description

ClusterNetworkEntry is a subnet from which to allocate PodIPs. A network of size HostPrefix (in CIDR notation) will be allocated when nodes join the cluster. If the HostPrefix field is not used by the plugin, it can be left unset. Not all network providers support multiple ClusterNetworks

Type

object

Property	Type	Description
cidr	**string**	
hostPrefix	**integer**	

22.1.25. .spec.defaultNetwork.ovnKubernetesConfig.ipsecConfig

Description

ipsecConfig enables and configures IPsec for pods on the pod network within the cluster.

Type

object

Property	Type	Description
full	**object**	full defines configuration parameters for the IPsec **Full** mode. This is permitted only when mode is configured with **Full**, and forbidden otherwise.

Property	Type	Description
mode	**string**	mode defines the behaviour of the ipsec configuration within the platform. Valid values are **Disabled**, **External** and **Full**. When 'Disabled', ipsec will not be enabled at the node level. When 'External', ipsec is enabled on the node level but requires the user to configure the secure communication parameters. This mode is for external secure communications and the configuration can be done using the k8s-nmstate operator. When 'Full', ipsec is configured on the node level and inter-pod secure communication within the cluster is configured. Note with **Full**, if ipsec is desired for communication with external (to the cluster) entities (such as storage arrays), this is left to the user to configure.

22.1.26. .spec.defaultNetwork.ovnKubernetesConfig.ipsecConfig.full

Description

full defines configuration parameters for the IPsec **Full** mode. This is permitted only when mode is configured with **Full**, and forbidden otherwise.

Type

object

Property	Type	Description

Property	Type	Description
encapsulation	**string**	encapsulation option to configure libreswan on how inter-pod traffic across nodes are encapsulated to handle NAT traversal. When configured it uses UDP port 4500 for the encapsulation. Valid values are Always, Auto and omitted. Always means enable UDP encapsulation regardless of whether NAT is detected. Auto means enable UDP encapsulation based on the detection of NAT. When omitted, this means no opinion and the platform is left to choose a reasonable default, which is subject to change over time. The current default is Auto.

22.1.27. .spec.defaultNetwork.ovnKubernetesConfig.ipv4

Description

ipv4 allows users to configure IP settings for IPv4 connections. When ommitted, this means no opinions and the default configuration is used. Check individual fields within ipv4 for details of default values.

Type

object

Property	Type	Description
internalJoinSubnet	**string**	internalJoinSubnet is a v4 subnet used internally by ovn-kubernetes in case the default one is being already used by something else. It must not overlap with any other subnet being used by OpenShift or by the node network. The size of the subnet must be larger than the number of nodes. The current default value is 100.64.0.0/16 The subnet must be large enough to accommodate one IP per node in your cluster The value must be in proper IPV4 CIDR format

Property	Type	Description
internalTransitSwitchSubnet	**string**	internalTransitSwitchSubnet is a v4 subnet in IPV4 CIDR format used internally by OVN-Kubernetes for the distributed transit switch in the OVN Interconnect architecture that connects the cluster routers on each node together to enable east west traffic. The subnet chosen should not overlap with other networks specified for OVN-Kubernetes as well as other networks used on the host. When ommitted, this means no opinion and the platform is left to choose a reasonable default which is subject to change over time. The current default subnet is 100.88.0.0/16 The subnet must be large enough to accommodate one IP per node in your cluster The value must be in proper IPV4 CIDR format

22.1.28. .spec.defaultNetwork.ovnKubernetesConfig.ipv6

Description

ipv6 allows users to configure IP settings for IPv6 connections. When ommitted, this means no opinions and the default configuration is used. Check individual fields within ipv4 for details of default values.

Type

object

Property	Type	Description

Property	Type	Description
internalJoinSubnet	**string**	internalJoinSubnet is a v6 subnet used internally by ovn-kubernetes in case the default one is being already used by something else. It must not overlap with any other subnet being used by OpenShift or by the node network. The size of the subnet must be larger than the number of nodes. The subnet must be large enough to accommodate one IP per node in your cluster The current default value is fd98::/64 The value must be in proper IPV6 CIDR format Note that IPV6 dual addresses are not permitted
internalTransitSwitchSubnet	**string**	internalTransitSwitchSubnet is a v4 subnet in IPV4 CIDR format used internally by OVN-Kubernetes for the distributed transit switch in the OVN Interconnect architecture that connects the cluster routers on each node together to enable east west traffic. The subnet chosen should not overlap with other networks specified for OVN-Kubernetes as well as other networks used on the host. When ommitted, this means no opinion and the platform is left to choose a reasonable default which is subject to change over time. The subnet must be large enough to accommodate one IP per node in your cluster The current default subnet is fd97::/64 The value must be in proper IPV6 CIDR format Note that IPV6 dual addresses are not permitted

22.1.29. .spec.defaultNetwork.ovnKubernetesConfig.policyAuditConfig

Description

policyAuditConfig is the configuration for network policy audit events. If unset, reported defaults are used.

Type

object

Property	Type	Description
destination	**string**	destination is the location for policy log messages. Regardless of this config, persistent logs will always be dumped to the host at /var/log/ovn/ however Additionally syslog output may be configured as follows. Valid values are: - "libc" → to use the libc syslog() function of the host node's journdald process - "udp:host:port" → for sending syslog over UDP - "unix:file" → for using the UNIX domain socket directly - "null" → to discard all messages logged to syslog The default is "null"
maxFileSize	**integer**	maxFilesSize is the max size an ACL_audit log file is allowed to reach before rotation occurs Units are in MB and the Default is 50MB
maxLogFiles	**integer**	maxLogFiles specifies the maximum number of ACL_audit log files that can be present.
rateLimit	**integer**	rateLimit is the approximate maximum number of messages to generate per-second per-node. If unset the default of 20 msg/sec is used.
syslogFacility	**string**	syslogFacility the RFC5424 facility for generated messages, e.g. "kern". Default is "local0"

22.1.30. .spec.exportNetworkFlows

Description

exportNetworkFlows enables and configures the export of network flow metadata from the pod network by using protocols NetFlow, SFlow or IPFIX. Currently only supported on OVN-Kubernetes plugin. If unset, flows will not be exported to any collector.

Type

object

Property	Type	Description
ipfix	**object**	ipfix defines IPFIX configuration.
netFlow	**object**	netFlow defines the NetFlow configuration.
sFlow	**object**	sFlow defines the SFlow configuration.

22.1.31. .spec.exportNetworkFlows.ipfix

Description

ipfix defines IPFIX configuration.

Type

object

Property	Type	Description
collectors	**array (string)**	ipfixCollectors is list of strings formatted as ip:port with a maximum of ten items

22.1.32. .spec.exportNetworkFlows.netFlow

Description

netFlow defines the NetFlow configuration.

Type

object

Property	Type	Description
collectors	**array (string)**	netFlow defines the NetFlow collectors that will consume the flow data exported from OVS. It is a list of strings formatted as ip:port with a maximum of ten items

22.1.33. .spec.exportNetworkFlows.sFlow

Description

sFlow defines the SFlow configuration.

Type

object

Property	Type	Description
collectors	**array (string)**	sFlowCollectors is list of strings formatted as ip:port with a maximum of ten items

22.1.34. .spec.kubeProxyConfig

Description

kubeProxyConfig lets us configure desired proxy configuration, if deployKubeProxy is true. If not specified, sensible defaults will be chosen by OpenShift directly.

Type

object

Property	Type	Description
bindAddress	**string**	The address to "bind" on Defaults to 0.0.0.0
iptablesSyncPeriod	**string**	An internal kube-proxy parameter. In older releases of OCP, this sometimes needed to be adjusted in large clusters for performance reasons, but this is no longer necessary, and there is no reason to change this from the default value. Default: 30s
proxyArguments	**object**	Any additional arguments to pass to the kubeproxy process
proxyArguments{}	**array (string)**	ProxyArgumentList is a list of arguments to pass to the kubeproxy process

22.1.35. .spec.kubeProxyConfig.proxyArguments

Description

Any additional arguments to pass to the kubeproxy process

Type

object

22.1.36. .spec.migration

Description

migration enables and configures cluster network migration, for network changes that cannot be made instantly.

Type

object

Property	Type	Description
features	**object**	features was previously used to configure which network plugin features would be migrated in a network type migration. DEPRECATED: network type migration is no longer supported, and setting this to a non-empty value will result in the network operator rejecting the configuration.
mode	**string**	mode indicates the mode of network type migration. DEPRECATED: network type migration is no longer supported, and setting this to a non-empty value will result in the network operator rejecting the configuration.
mtu	**object**	mtu contains the MTU migration configuration. Set this to allow changing the MTU values for the default network. If unset, the operation of changing the MTU for the default network will be rejected.
networkType	**string**	networkType was previously used when changing the default network type. DEPRECATED: network type migration is no longer supported, and setting this to a non-empty value will result in the network operator rejecting the configuration.

22.1.37. .spec.migration.features

Description

features was previously used to configure which network plugin features would be migrated in a network type migration. DEPRECATED: network type migration is no longer supported, and setting this to a non-empty value will result in the network operator rejecting the configuration.

Type

object

Property	Type	Description
egressFirewall	**boolean**	egressFirewall specified whether or not the Egress Firewall configuration was migrated. DEPRECATED: network type migration is no longer supported.
egressIP	**boolean**	egressIP specified whether or not the Egress IP configuration was migrated. DEPRECATED: network type migration is no longer supported.
multicast	**boolean**	multicast specified whether or not the multicast configuration was migrated. DEPRECATED: network type migration is no longer supported.

22.1.38. .spec.migration.mtu

Description

mtu contains the MTU migration configuration. Set this to allow changing the MTU values for the default network. If unset, the operation of changing the MTU for the default network will be rejected.

Type

object

Property	Type	Description
machine	**object**	machine contains MTU migration configuration for the machine's uplink. Needs to be migrated along with the default network MTU unless the current uplink MTU already accommodates the default network MTU.
network	**object**	network contains information about MTU migration for the default network. Migrations are only allowed to MTU values lower than the machine's uplink MTU by the minimum appropriate offset.

22.1.39. .spec.migration.mtu.machine

Description

machine contains MTU migration configuration for the machine's uplink. Needs to be migrated along with the default network MTU unless the current uplink MTU already accommodates the default network MTU.

Type

object

Property	Type	Description
from	**integer**	from is the MTU to migrate from.
to	**integer**	to is the MTU to migrate to.

22.1.40. .spec.migration.mtu.network

Description

network contains information about MTU migration for the default network. Migrations are only allowed to MTU values lower than the machine's uplink MTU by the minimum appropriate offset.

Type

object

Property	Type	Description
from	**integer**	from is the MTU to migrate from.
to	**integer**	to is the MTU to migrate to.

22.1.41. .status

Description

NetworkStatus is detailed operator status, which is distilled up to the Network clusteroperator object.

Type

object

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status
conditions[]	**object**	OperatorCondition is just the standard condition fields.
generations	**array**	generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Property	Type	Description
generations[]	**object**	GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.
latestAvailableRevision	**integer**	latestAvailableRevision is the deploymentID of the most recent deployment
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with
readyReplicas	**integer**	readyReplicas indicates how many replicas are ready and at the desired state
version	**string**	version is the level this availability applies to

22.1.42. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

22.1.43. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	
reason	**string**	
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

22.1.44. .status.generations

Description

generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Type

array

22.1.45. .status.generations[]

Description

GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.

Type

object

Required

- **group**
- **name**
- **namespace**
- **resource**

Property	Type	Description
group	**string**	group is the group of the thing you're tracking
hash	**string**	hash is an optional field set for resources without generation that are content sensitive like secrets and configmaps
lastGeneration	**integer**	lastGeneration is the last generation of the workload controller involved
name	**string**	name is the name of the thing you're tracking
namespace	**string**	namespace is where the thing you're tracking is
resource	**string**	resource is the resource type of the thing you're tracking

22.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/networks**
  - **DELETE**: delete collection of Network
  - **GET**: list objects of kind Network
  - **POST**: create a Network
- **/apis/operator.openshift.io/v1/networks/{name}**
  - **DELETE**: delete a Network
  - **GET**: read the specified Network
  - **PATCH**: partially update the specified Network
  - **PUT**: replace the specified Network
- **/apis/operator.openshift.io/v1/networks/{name}/status**
  - **GET**: read status of the specified Network
  - **PATCH**: partially update status of the specified Network

- **PUT**: replace status of the specified Network

22.2.1. /apis/operator.openshift.io/v1/networks

HTTP method

DELETE

Description

delete collection of Network

Table 22.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind Network

Table 22.2. HTTP responses

HTTP code	Reponse body
200 - OK	**NetworkList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a Network

Table 22.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 22.4. Body parameters

Parameter	Type	Description
body	**Network** schema	

Table 22.5. HTTP responses

HTTP code	Reponse body
200 - OK	**Network** schema
201 - Created	**Network** schema
202 - Accepted	**Network** schema
401 - Unauthorized	Empty

22.2.2. /apis/operator.openshift.io/v1/networks/{name}

Table 22.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the Network

HTTP method

DELETE

Description

delete a Network

Table 22.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 22.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified Network

Table 22.9. HTTP responses

HTTP code	Reponse body
200 - OK	**Network** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified Network

Table 22.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 22.11. HTTP responses

HTTP code	Reponse body
200 - OK	**Network** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified Network

Table 22.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 22.13. Body parameters

Parameter	Type	Description
body	**Network** schema	

Table 22.14. HTTP responses

HTTP code	Reponse body
200 - OK	**Network** schema
201 - Created	**Network** schema
401 - Unauthorized	Empty

22.2.3. /apis/operator.openshift.io/v1/networks/{name}/status

Table 22.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the Network

HTTP method

GET

Description

read status of the specified Network

Table 22.16. HTTP responses

HTTP code	Reponse body
200 - OK	**Network** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified Network

Table 22.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 22.18. HTTP responses

HTTP code	Reponse body
200 - OK	**Network** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified Network

Table 22.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 22.20. Body parameters

Parameter	Type	Description
body	**Network** schema	

Table 22.21. HTTP responses

HTTP code	Reponse body
200 - OK	**Network** schema
201 - Created	**Network** schema
401 - Unauthorized	Empty

CHAPTER 23. OPENSHIFTAPISERVER [OPERATOR.OPENSHIFT.IO/V1]

Description

OpenShiftAPIServer provides information to configure an operator to manage openshift-apiserver. Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

23.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	spec is the specification of the desired behavior of the OpenShift API Server.
status	**object**	status defines the observed status of the OpenShift API Server.

23.1.1. .spec

Description

spec is the specification of the desired behavior of the OpenShift API Server.

Type

object

Property	Type	Description
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
managementState	**string**	managementState indicates whether and how the operator should manage the component
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator

Property	Type	Description
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
unsupportedConfigOverrides	``	unsupportedConfigOverrides overrides the final configuration that was computed by the operator. Red Hat does not support the use of this field. Misuse of this field could lead to unexpected behavior or conflict with other configuration options. Seek guidance from the Red Hat support before using this field. Use of this property blocks cluster upgrades, it must be removed before upgrading your cluster.

23.1.2. .status

Description

status defines the observed status of the OpenShift API Server.

Type

object

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status
conditions[]	**object**	OperatorCondition is just the standard condition fields.
generations	**array**	generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Property	Type	Description
generations[]	**object**	GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.
latestAvailableRevision	**integer**	latestAvailableRevision is the deploymentID of the most recent deployment
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with
readyReplicas	**integer**	readyReplicas indicates how many replicas are ready and at the desired state
version	**string**	version is the level this availability applies to

23.1.3. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

23.1.4. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	
reason	**string**	
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

23.1.5. .status.generations

Description

generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Type

array

23.1.6. .status.generations[]

Description

GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.

Type

object

Required

- **group**
- **name**
- **namespace**
- **resource**

Property	Type	Description
group	**string**	group is the group of the thing you're tracking
hash	**string**	hash is an optional field set for resources without generation that are content sensitive like secrets and configmaps
lastGeneration	**integer**	lastGeneration is the last generation of the workload controller involved
name	**string**	name is the name of the thing you're tracking
namespace	**string**	namespace is where the thing you're tracking is
resource	**string**	resource is the resource type of the thing you're tracking

23.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/openshiftapiservers**
  - **DELETE**: delete collection of OpenShiftAPIServer
  - **GET**: list objects of kind OpenShiftAPIServer
  - **POST**: create an OpenShiftAPIServer
- **/apis/operator.openshift.io/v1/openshiftapiservers/{name}**
  - **DELETE**: delete an OpenShiftAPIServer
  - **GET**: read the specified OpenShiftAPIServer
  - **PATCH**: partially update the specified OpenShiftAPIServer
  - **PUT**: replace the specified OpenShiftAPIServer
- **/apis/operator.openshift.io/v1/openshiftapiservers/{name}/status**
  - **GET**: read status of the specified OpenShiftAPIServer
  - **PATCH**: partially update status of the specified OpenShiftAPIServer

- **PUT**: replace status of the specified OpenShiftAPIServer

23.2.1. /apis/operator.openshift.io/v1/openshiftapiservers

HTTP method

DELETE

Description

delete collection of OpenShiftAPIServer

Table 23.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind OpenShiftAPIServer

Table 23.2. HTTP responses

HTTP code	Reponse body
200 - OK	**OpenShiftAPIServerList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create an OpenShiftAPIServer

Table 23.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 23.4. Body parameters

Parameter	Type	Description
body	**OpenShiftAPIServer** schema	

Table 23.5. HTTP responses

HTTP code	Reponse body
200 - OK	**OpenShiftAPIServer** schema
201 - Created	**OpenShiftAPIServer** schema
202 - Accepted	**OpenShiftAPIServer** schema
401 - Unauthorized	Empty

23.2.2. /apis/operator.openshift.io/v1/openshiftapiservers/{name}

Table 23.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the OpenShiftAPIServer

HTTP method

DELETE

Description

delete an OpenShiftAPIServer

Table 23.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 23.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified OpenShiftAPIServer

Table 23.9. HTTP responses

HTTP code	Reponse body
200 - OK	**OpenShiftAPIServer** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified OpenShiftAPIServer

Table 23.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 23.11. HTTP responses

HTTP code	Reponse body
200 - OK	**OpenShiftAPIServer** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified OpenShiftAPIServer

Table 23.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 23.13. Body parameters

Parameter	Type	Description
body	**OpenShiftAPIServer** schema	

Table 23.14. HTTP responses

HTTP code	Reponse body
200 - OK	**OpenShiftAPIServer** schema
201 - Created	**OpenShiftAPIServer** schema
401 - Unauthorized	Empty

23.2.3. /apis/operator.openshift.io/v1/openshiftapiservers/{name}/status

Table 23.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the OpenShiftAPIServer

HTTP method

GET

Description

read status of the specified OpenShiftAPIServer

Table 23.16. HTTP responses

HTTP code	Reponse body
200 - OK	**OpenShiftAPIServer** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified OpenShiftAPIServer

Table 23.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 23.18. HTTP responses

HTTP code	Reponse body
200 - OK	**OpenShiftAPIServer** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified OpenShiftAPIServer

Table 23.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 23.20. Body parameters

Parameter	Type	Description
body	**OpenShiftAPIServer** schema	

Table 23.21. HTTP responses

HTTP code	Reponse body
200 - OK	**OpenShiftAPIServer** schema
201 - Created	**OpenShiftAPIServer** schema
401 - Unauthorized	Empty

CHAPTER 24. OPENSHIFTCONTROLLERMANAGER [OPERATOR.OPENSHIFT.IO/V1]

Description

OpenShiftControllerManager provides information to configure an operator to manage openshift-controller-manager.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

24.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	

Property	Type	Description
status	**object**	

24.1.1. .spec

Description

Type

object

Property	Type	Description
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
managementState	**string**	managementState indicates whether and how the operator should manage the component
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".

Property	Type	Description
unsupportedConfigOverrides	``	unsupportedConfigOverrides overrides the final configuration that was computed by the operator. Red Hat does not support the use of this field. Misuse of this field could lead to unexpected behavior or conflict with other configuration options. Seek guidance from the Red Hat support before using this field. Use of this property blocks cluster upgrades, it must be removed before upgrading your cluster.

24.1.2. .status

Description

Type

object

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status
conditions[]	**object**	OperatorCondition is just the standard condition fields.
generations	**array**	generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.
generations[]	**object**	GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.
latestAvailableRevision	**integer**	latestAvailableRevision is the deploymentID of the most recent deployment
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with

Property	Type	Description
readyReplicas	**integer**	readyReplicas indicates how many replicas are ready and at the desired state
version	**string**	version is the level this availability applies to

24.1.3. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

24.1.4. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	
reason	**string**	

Property	Type	Description
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

24.1.5. .status.generations

Description

generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Type

array

24.1.6. .status.generations[]

Description

GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.

Type

object

Required

- **group**
- **name**
- **namespace**
- **resource**

Property	Type	Description
group	**string**	group is the group of the thing you're tracking
hash	**string**	hash is an optional field set for resources without generation that are content sensitive like secrets and configmaps
lastGeneration	**integer**	lastGeneration is the last generation of the workload controller involved

Property	Type	Description
name	**string**	name is the name of the thing you're tracking
namespace	**string**	namespace is where the thing you're tracking is
resource	**string**	resource is the resource type of the thing you're tracking

24.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/openshiftcontrollermanagers**
  - **DELETE**: delete collection of OpenShiftControllerManager
  - **GET**: list objects of kind OpenShiftControllerManager
  - **POST**: create an OpenShiftControllerManager
- **/apis/operator.openshift.io/v1/openshiftcontrollermanagers/{name}**
  - **DELETE**: delete an OpenShiftControllerManager
  - **GET**: read the specified OpenShiftControllerManager
  - **PATCH**: partially update the specified OpenShiftControllerManager
  - **PUT**: replace the specified OpenShiftControllerManager
- **/apis/operator.openshift.io/v1/openshiftcontrollermanagers/{name}/status**
  - **GET**: read status of the specified OpenShiftControllerManager
  - **PATCH**: partially update status of the specified OpenShiftControllerManager
  - **PUT**: replace status of the specified OpenShiftControllerManager

24.2.1. /apis/operator.openshift.io/v1/openshiftcontrollermanagers

HTTP method

DELETE

Description

delete collection of OpenShiftControllerManager

Table 24.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind OpenShiftControllerManager

Table 24.2. HTTP responses

HTTP code	Reponse body
200 - OK	**OpenShiftControllerManagerList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create an OpenShiftControllerManager

Table 24.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 24.4. Body parameters

Parameter	Type	Description
body	**OpenShiftControllerManager** schema	

Table 24.5. HTTP responses

HTTP code	Reponse body
200 - OK	**OpenShiftControllerManager** schema
201 - Created	**OpenShiftControllerManager** schema
202 - Accepted	**OpenShiftControllerManager** schema
401 - Unauthorized	Empty

24.2.2. /apis/operator.openshift.io/v1/openshiftcontrollermanagers/{name}

Table 24.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the OpenShiftControllerManager

HTTP method

DELETE

Description

delete an OpenShiftControllerManager

Table 24.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 24.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified OpenShiftControllerManager

Table 24.9. HTTP responses

HTTP code	Reponse body
200 - OK	**OpenShiftControllerManager** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified OpenShiftControllerManager

Table 24.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 24.11. HTTP responses

HTTP code	Reponse body
200 - OK	**OpenShiftControllerManager** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified OpenShiftControllerManager

Table 24.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 24.13. Body parameters

Parameter	Type	Description
body	**OpenShiftControllerManager** schema	

Table 24.14. HTTP responses

HTTP code	Reponse body
200 - OK	**OpenShiftControllerManager** schema
201 - Created	**OpenShiftControllerManager** schema
401 - Unauthorized	Empty

24.2.3. /apis/operator.openshift.io/v1/openshiftcontrollermanagers/{name}/status

Table 24.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the OpenShiftControllerManager

HTTP method

GET

Description

read status of the specified OpenShiftControllerManager

Table 24.16. HTTP responses

HTTP code	Reponse body
200 - OK	**OpenShiftControllerManager** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified OpenShiftControllerManager

Table 24.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 24.18. HTTP responses

HTTP code	Reponse body
200 - OK	**OpenShiftControllerManager** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified OpenShiftControllerManager

Table 24.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 24.20. Body parameters

Parameter	Type	Description
body	**OpenShiftControllerManager** schema	

Table 24.21. HTTP responses

HTTP code	Reponse body
200 - OK	**OpenShiftControllerManager** schema
201 - Created	**OpenShiftControllerManager** schema
401 - Unauthorized	Empty

CHAPTER 25. OPERATORPKI [NETWORK.OPERATOR.OPENSHIFT.IO/V1]

Description

OperatorPKI is a simple certificate authority. It is not intended for external use - rather, it is internal to the network operator. The CNO creates a CA and a certificate signed by that CA. The certificate has both ClientAuth and ServerAuth extended usages enabled.

> More specifically, given an OperatorPKI with <name>, the CNO will manage:

- A Secret called <name>-ca with two data keys:
- tls.key - the private key
- tls.crt - the CA certificate
- A ConfigMap called <name>-ca with a single data key:
- cabundle.crt - the CA certificate(s)
- A Secret called <name>-cert with two data keys:
- tls.key - the private key
- tls.crt - the certificate, signed by the CA

The CA certificate will have a validity of 10 years, rotated after 9. The target certificate will have a validity of 6 months, rotated after 3

The CA certificate will have a CommonName of "<namespace>_<name>-ca@<timestamp>", where <timestamp> is the last rotation time.

Type

object

Required

- **spec**

25.1. SPECIFICATION

Property	Type	Description

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	OperatorPKISpec is the PKI configuration.
status	**object**	OperatorPKIStatus is not implemented.

25.1.1. .spec

Description

OperatorPKISpec is the PKI configuration.

Type

object

Required

- **targetCert**

Property	Type	Description
targetCert	**object**	targetCert configures the certificate signed by the CA. It will have both ClientAuth and ServerAuth enabled

25.1.2. .spec.targetCert

Description

targetCert configures the certificate signed by the CA. It will have both ClientAuth and ServerAuth enabled

Type

object

Required

- **commonName**

Property	Type	Description
commonName	**string**	commonName is the value in the certificate's CN

25.1.3. .status

Description

OperatorPKIStatus is not implemented.

Type

object

25.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/network.operator.openshift.io/v1/operatorpkis**
  - **GET**: list objects of kind OperatorPKI
- **/apis/network.operator.openshift.io/v1/namespaces/{namespace}/operatorpkis**
  - **DELETE**: delete collection of OperatorPKI
  - **GET**: list objects of kind OperatorPKI
  - **POST**: create an OperatorPKI
- **/apis/network.operator.openshift.io/v1/namespaces/{namespace}/operatorpkis/{name}**

- **DELETE**: delete an OperatorPKI
- **GET**: read the specified OperatorPKI
- **PATCH**: partially update the specified OperatorPKI
- **PUT**: replace the specified OperatorPKI

25.2.1. /apis/network.operator.openshift.io/v1/operatorpkis

HTTP method

GET

Description

list objects of kind OperatorPKI

Table 25.1. HTTP responses

HTTP code	Reponse body
200 - OK	**OperatorPKIList** schema
401 - Unauthorized	Empty

25.2.2. /apis/network.operator.openshift.io/v1/namespaces/{namespace}/operatorpk

HTTP method

DELETE

Description

delete collection of OperatorPKI

Table 25.2. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind OperatorPKI

Table 25.3. HTTP responses

HTTP code	Reponse body
200 - OK	**OperatorPKIList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create an OperatorPKI

Table 25.4. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 25.5. Body parameters

Parameter	Type	Description
body	**OperatorPKI** schema	

Table 25.6. HTTP responses

HTTP code	Reponse body
200 - OK	**OperatorPKI** schema
201 - Created	**OperatorPKI** schema
202 - Accepted	**OperatorPKI** schema
401 - Unauthorized	Empty

25.2.3. /apis/network.operator.openshift.io/v1/namespaces/{namespace}/operatorpk

Table 25.7. Global path parameters

Parameter	Type	Description
name	**string**	name of the OperatorPKI

HTTP method

DELETE

Description

delete an OperatorPKI

Table 25.8. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 25.9. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified OperatorPKI

Table 25.10. HTTP responses

HTTP code	Reponse body
200 - OK	**OperatorPKI** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified OperatorPKI

Table 25.11. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 25.12. HTTP responses

HTTP code	Reponse body
200 - OK	**OperatorPKI** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified OperatorPKI

Table 25.13. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 25.14. Body parameters

Parameter	Type	Description
body	**OperatorPKI** schema	

Table 25.15. HTTP responses

HTTP code	Reponse body
200 - OK	**OperatorPKI** schema
201 - Created	**OperatorPKI** schema
401 - Unauthorized	Empty

CHAPTER 26. SERVICECA [OPERATOR.OPENSHIFT.IO/V1]

Description

ServiceCA provides information to configure an operator to manage the service cert controllers Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

26.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	spec holds user settable values for configuration

Property	Type	Description
status	**object**	status holds observed values from the cluster. They may not be overridden.

26.1.1. .spec

Description

spec holds user settable values for configuration

Type

object

Property	Type	Description
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
managementState	**string**	managementState indicates whether and how the operator should manage the component
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".

Property	Type	Description
unsupportedConfigOverrides	``	unsupportedConfigOverrides overrides the final configuration that was computed by the operator. Red Hat does not support the use of this field. Misuse of this field could lead to unexpected behavior or conflict with other configuration options. Seek guidance from the Red Hat support before using this field. Use of this property blocks cluster upgrades, it must be removed before upgrading your cluster.

26.1.2. .status

Description

status holds observed values from the cluster. They may not be overridden.

Type

object

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status
conditions[]	**object**	OperatorCondition is just the standard condition fields.
generations	**array**	generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.
generations[]	**object**	GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.
latestAvailableRevision	**integer**	latestAvailableRevision is the deploymentID of the most recent deployment
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with

Property	Type	Description
readyReplicas	**integer**	readyReplicas indicates how many replicas are ready and at the desired state
version	**string**	version is the level this availability applies to

26.1.3. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

26.1.4. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	
reason	**string**	
status	**string**	status of the condition, one of True, False, Unknown.

Property	Type	Description
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

26.1.5. .status.generations

Description

generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Type

array

26.1.6. .status.generations[]

Description

GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.

Type

object

Required

- **group**
- **name**
- **namespace**
- **resource**

Property	Type	Description
group	**string**	group is the group of the thing you're tracking
hash	**string**	hash is an optional field set for resources without generation that are content sensitive like secrets and configmaps
lastGeneration	**integer**	lastGeneration is the last generation of the workload controller involved
name	**string**	name is the name of the thing you're tracking

Property	Type	Description
namespace	**string**	namespace is where the thing you're tracking is
resource	**string**	resource is the resource type of the thing you're tracking

26.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/servicecas**
  - **DELETE**: delete collection of ServiceCA
  - **GET**: list objects of kind ServiceCA
  - **POST**: create a ServiceCA
- **/apis/operator.openshift.io/v1/servicecas/{name}**
  - **DELETE**: delete a ServiceCA
  - **GET**: read the specified ServiceCA
  - **PATCH**: partially update the specified ServiceCA
  - **PUT**: replace the specified ServiceCA
- **/apis/operator.openshift.io/v1/servicecas/{name}/status**
  - **GET**: read status of the specified ServiceCA
  - **PATCH**: partially update status of the specified ServiceCA
  - **PUT**: replace status of the specified ServiceCA

26.2.1. /apis/operator.openshift.io/v1/servicecas

HTTP method

DELETE

Description

delete collection of ServiceCA

Table 26.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema

HTTP code	Reponse body
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind ServiceCA

Table 26.2. HTTP responses

HTTP code	Reponse body
200 - OK	**ServiceCAList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a ServiceCA

Table 26.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 26.4. Body parameters

Parameter	Type	Description
body	**ServiceCA** schema	

Table 26.5. HTTP responses

HTTP code	Reponse body
200 - OK	**ServiceCA** schema
201 - Created	**ServiceCA** schema
202 - Accepted	**ServiceCA** schema
401 - Unauthorized	Empty

26.2.2. /apis/operator.openshift.io/v1/servicecas/{name}

Table 26.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the ServiceCA

HTTP method

DELETE

Description

delete a ServiceCA

Table 26.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 26.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified ServiceCA

Table 26.9. HTTP responses

HTTP code	Reponse body
200 - OK	**ServiceCA** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified ServiceCA

Table 26.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 26.11. HTTP responses

HTTP code	Reponse body
200 - OK	**ServiceCA** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified ServiceCA

Table 26.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 26.13. Body parameters

Parameter	Type	Description
body	**ServiceCA** schema	

Table 26.14. HTTP responses

HTTP code	Reponse body
200 - OK	**ServiceCA** schema
201 - Created	**ServiceCA** schema
401 - Unauthorized	Empty

26.2.3. /apis/operator.openshift.io/v1/servicecas/{name}/status

Table 26.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the ServiceCA

HTTP method

GET

Description

read status of the specified ServiceCA

Table 26.16. HTTP responses

HTTP code	Reponse body
200 - OK	**ServiceCA** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified ServiceCA

Table 26.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 26.18. HTTP responses

HTTP code	Reponse body
200 - OK	**ServiceCA** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified ServiceCA

Table 26.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 26.20. Body parameters

Parameter	Type	Description
body	**ServiceCA** schema	

Table 26.21. HTTP responses

HTTP code	Reponse body
200 - OK	**ServiceCA** schema
201 - Created	**ServiceCA** schema
401 - Unauthorized	Empty

CHAPTER 27. STORAGE [OPERATOR.OPENSHIFT.IO/V1]

Description

Storage provides a means to configure an operator to manage the cluster storage operator. **cluster** is the canonical name.
Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

Required

- **spec**

27.1. SPECIFICATION

Property	Type	Description
apiVersion	**string**	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	**string**	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	spec holds user settable values for configuration

Property	Type	Description
status	**object**	status holds observed values from the cluster. They may not be overridden.

27.1.1. .spec

Description

spec holds user settable values for configuration

Type

object

Property	Type	Description
logLevel	**string**	logLevel is an intent based logging for an overall component. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for their operands. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".
managementState	**string**	managementState indicates whether and how the operator should manage the component
observedConfig	``	observedConfig holds a sparse config that controller has observed from the cluster state. It exists in spec because it is an input to the level for the operator
operatorLogLevel	**string**	operatorLogLevel is an intent based logging for the operator itself. It does not give fine grained control, but it is a simple way to manage coarse grained logging choices that operators have to interpret for themselves. Valid values are: "Normal", "Debug", "Trace", "TraceAll". Defaults to "Normal".

Property	Type	Description
unsupportedConfigOverrides	``	unsupportedConfigOverrides overrides the final configuration that was computed by the operator. Red Hat does not support the use of this field. Misuse of this field could lead to unexpected behavior or conflict with other configuration options. Seek guidance from the Red Hat support before using this field. Use of this property blocks cluster upgrades, it must be removed before upgrading your cluster.
vsphereStorageDriver	**string**	vsphereStorageDriver indicates the storage driver to use on VSphere clusters. Once this field is set to CSIWithMigrationDriver, it can not be changed. If this is empty, the platform will choose a good default, which may change over time without notice. The current default is CSIWithMigrationDriver and may not be changed. DEPRECATED: This field will be removed in a future release.

27.1.2. .status

Description

status holds observed values from the cluster. They may not be overridden.

Type

object

Property	Type	Description
conditions	**array**	conditions is a list of conditions and their status
conditions[]	**object**	OperatorCondition is just the standard condition fields.

Property	Type	Description
generations	**array**	generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.
generations[]	**object**	GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.
latestAvailableRevision	**integer**	latestAvailableRevision is the deploymentID of the most recent deployment
observedGeneration	**integer**	observedGeneration is the last generation change you've dealt with
readyReplicas	**integer**	readyReplicas indicates how many replicas are ready and at the desired state
version	**string**	version is the level this availability applies to

27.1.3. .status.conditions

Description

conditions is a list of conditions and their status

Type

array

27.1.4. .status.conditions[]

Description

OperatorCondition is just the standard condition fields.

Type

object

Required

- **lastTransitionTime**
- **status**
- **type**

Property	Type	Description
lastTransitionTime	**string**	lastTransitionTime is the last time the condition transitioned from one status to another. This should be when the underlying condition changed. If that is not known, then using the time when the API field changed is acceptable.
message	**string**	
reason	**string**	
status	**string**	status of the condition, one of True, False, Unknown.
type	**string**	type of condition in CamelCase or in foo.example.com/CamelCase.

27.1.5. .status.generations

Description

generations are used to determine when an item needs to be reconciled or has changed in a way that needs a reaction.

Type

array

27.1.6. .status.generations[]

Description

GenerationStatus keeps track of the generation for a given resource so that decisions about forced updates can be made.

Type

object

Required

- **group**
- **name**
- **namespace**
- **resource**

Property	Type	Description
group	**string**	group is the group of the thing you're tracking
hash	**string**	hash is an optional field set for resources without generation that are content sensitive like secrets and configmaps
lastGeneration	**integer**	lastGeneration is the last generation of the workload controller involved
name	**string**	name is the name of the thing you're tracking
namespace	**string**	namespace is where the thing you're tracking is
resource	**string**	resource is the resource type of the thing you're tracking

27.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/operator.openshift.io/v1/storages**
  - **DELETE**: delete collection of Storage
  - **GET**: list objects of kind Storage
  - **POST**: create a Storage
- **/apis/operator.openshift.io/v1/storages/{name}**
  - **DELETE**: delete a Storage
  - **GET**: read the specified Storage
  - **PATCH**: partially update the specified Storage
  - **PUT**: replace the specified Storage
- **/apis/operator.openshift.io/v1/storages/{name}/status**
  - **GET**: read status of the specified Storage
  - **PATCH**: partially update status of the specified Storage

- **PUT**: replace status of the specified Storage

27.2.1. /apis/operator.openshift.io/v1/storages

HTTP method

DELETE

Description

delete collection of Storage

Table 27.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind Storage

Table 27.2. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a Storage

Table 27.3. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 27.4. Body parameters

Parameter	Type	Description
body	**Storage** schema	

Table 27.5. HTTP responses

HTTP code	Reponse body
200 - OK	**Storage** schema
201 - Created	**Storage** schema
202 - Accepted	**Storage** schema
401 - Unauthorized	Empty

27.2.2. /apis/operator.openshift.io/v1/storages/{name}

Table 27.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the Storage

HTTP method

DELETE

Description

delete a Storage

Table 27.7. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Table 27.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified Storage

Table 27.9. HTTP responses

HTTP code	Reponse body
200 - OK	**Storage** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified Storage

Table 27.10. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 27.11. HTTP responses

HTTP code	Reponse body
200 - OK	**Storage** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified Storage

Table 27.12. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 27.13. Body parameters

Parameter	Type	Description
body	**Storage** schema	

Table 27.14. HTTP responses

HTTP code	Reponse body
200 - OK	**Storage** schema
201 - Created	**Storage** schema
401 - Unauthorized	Empty

27.2.3. /apis/operator.openshift.io/v1/storages/{name}/status

Table 27.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the Storage

HTTP method

GET

Description

read status of the specified Storage

Table 27.16. HTTP responses

HTTP code	Reponse body
200 - OK	**Storage** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified Storage

Table 27.17. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 27.18. HTTP responses

HTTP code	Reponse body
200 - OK	**Storage** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified Storage

Table 27.19. Query parameters

Parameter	Type	Description
dryRun	**string**	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	**string**	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Table 27.20. Body parameters

Parameter	Type	Description
body	**Storage** schema	

Table 27.21. HTTP responses

HTTP code	Reponse body
200 - OK	**Storage** schema
201 - Created	**Storage** schema
401 - Unauthorized	Empty